A Lightweight Time-series Analysis Model through Multi-Teacher Knowledge Distillation for Food Price Forecasting

by

Shifat Zaman
21366024

A thesis submitted to the Department of Computer Science and Engineering
in partial fulfillment of the requirements for the degree of
M.Sc. in Computer Science & Engineering

Department of Computer Science and Engineering
BRAC University
January 2026

Declaration

It is hereby declared that

1. The thesis submitted is my own original work while completing degree at BRAC University.
2. The thesis does not contain material previously published or written by a third party, except where this is appropriately cited through full and accurate referencing.
3. The thesis does not contain material which has been accepted, or submitted, for any other degree or diploma at a university or other institution.
4. I have acknowledged all main sources of help.

Student's Full Name & Signature:

Shifat Zaman
21366024

Approval

The thesis titled "A Lightweight Time-series Analysis Model through Multi-Teacher Knowledge Distillation for Food Price Forecasting" submitted by

1. Shifat Zaman (21366024)

of Summer, 2021 has been accepted as satisfactory in partial fulfillment of the requirement for the degree of M.Sc. in Computer Science & Engineering on 29 January 2026.

Examining Committee:

Examiner::
(External)

Dr. Ashis Talukder, Ph.D.
Associate Professor
University of Dhaka

Examiner::
(Internal)

Rafeed Rahman
Senior Lecturer
Department of Computer Science and Engineering
BRAC University

Supervisor:
(Member)

Dr. Md. Golam Rabiul Alam
Professor
Department of Computer Science and Engineering
BRAC University

Program Coordinator:
(Member)

Dr. Md Sadek Ferdous
Professor
Department of Computer Science and Engineering
BRAC University

Head of Department:
(Chair)

Sadia Hamid Kazi, Ph.D.
Chairperson and Associate Professor
Department of Computer Science and Engineering
BRAC University

Ethics Statement

This research utilizes publicly available data from the World Food Programme (WFP) Food Prices dataset for Bangladesh. The dataset is openly accessible and does not contain any personally identifiable information.
The research was conducted in accordance with ethical guidelines for computational research:

- No human subjects were involved in this study
- All data used is publicly available and properly attributed
- The research aims to contribute positively to food security efforts in developing nations
- The code and methodology are made available for reproducibility and transparency

Abstract

Accurate food price forecasting is critical for food security planning, particularly in developing nations like Bangladesh where price volatility can significantly impact vulnerable populations; however, existing deep learning approaches for time series forecasting often require substantial computational resources, limiting their deployment in resource-constrained environments. This study presents a novel multi-teacher knowledge distillation framework that transfers knowledge from an ensemble of three distinct teacher architectures—DLinear, PatchTST, and N-BEATS—trained on World Food Programme (WFP) Bangladesh commodity price data from the Dhaka Division to compact student models (MLP, GRU, KAN) through a multi-component distillation loss comprising prediction-level matching, feature-level alignment, and price-difference learning, with an uncertainty-weighted mechanism that focuses training on confident teacher predictions while dynamically weighting teacher contributions based on validation performance. Experimental evaluation on four food commodities (Lentils, Oil, Rice, and Wheat flour) with a 6-month input window demonstrates that the proposed approach achieves a Mean Absolute Error (MAE) of 1.959 BDT/unit with a Mean Absolute Percentage Error (MAPE) of only 3.73%, representing a 37% improvement over the supervised learning baseline, 69% improvement over traditional ARIMA, and 81% improvement over LSTM baselines, with the three-teacher ensemble distilled to an MLP student achieving the best results and outperforming all single-teacher and two-teacher configurations. The resulting student model requires only 200K parameters (compared to over 1M in the teacher ensemble) and achieves inference in sub-millisecond time on standard CPU hardware without GPU acceleration, enabling deployment in humanitarian field offices with limited computational infrastructure. This study contributes a reproducible, configuration-driven framework for knowledge distillation in time series forecasting, demonstrating that sophisticated ensemble-level accuracy can be achieved with lightweight models suitable for resource-constrained field deployment in food security applications.

Keywords: Knowledge Distillation; Time Series Forecasting; Food Price Prediction; Multi-Teacher Learning; Deep Learning; Transfer Learning; WFP Data

Dedication

To all those working towards food security in developing nations.

Acknowledgement

I would like to express my sincere gratitude to my supervisor, Dr. Md. Golam Rabiul Alam, for his invaluable guidance, continuous support, and constructive feedback throughout this research. His expertise and insights were instrumental in shaping the direction and quality of this work.

I am grateful to the Department of Computer Science and Engineering at BRAC University for providing the academic environment and resources necessary to complete this study. The knowledge and skills gained through the curriculum laid the foundation for this research.

Finally, I extend my appreciation to the World Food Programme for making their food price monitoring data publicly available, enabling research that contributes to food security applications in developing nations.

Table of Contents

List of Figures

List of Tables

Nomenclature

The next list describes several symbols & abbreviations that will be later used within the body of the document

BDT Bangladeshi Taka

GRU Gated Recurrent Unit

KAN Kolmogorov-Arnold Network

KD Knowledge Distillation

MAE Mean Absolute Error

$MAPE$ Mean Absolute Percentage Error

MLP Multi-Layer Perceptron

$RMSE$ Root Mean Square Error

WFP World Food Programme

Chapter 1

Introduction

This chapter introduces the research context, motivation, and objectives of this study. The chapter begins by examining the importance of food price forecasting for food security in developing nations, particularly Bangladesh, and discusses the limitations of traditional forecasting methods compared to modern deep learning approaches. It then presents the motivation for using knowledge distillation to create lightweight forecasting models suitable for resource-constrained deployment environments. The chapter outlines the specific challenges addressed in this research, states the research objectives, summarizes the key contributions, and provides an overview of the thesis structure.

1.1 Background

Food security remains a pressing challenge in developing nations, where price volatility of essential commodities directly threatens vulnerable populations. When prices spike unexpectedly, families are forced to make difficult choices—reducing meal frequency, switching to less nutritious alternatives, or sacrificing other essential needs like healthcare and education. Bangladesh, with over 170 million people and an economy where agriculture contributes approximately 12% of GDP and employs 35% of the labor force [1], is particularly susceptible to food price fluctuations. Staple commodities such as rice, wheat, lentils, and cooking oil form the foundation of the Bangladeshi diet, and their price movements have immediate effects on household welfare. According to the Household Income and Expenditure Survey [2], poor households allocate up to 70% of their income to food, making them extremely vulnerable to price shocks. The 2007-2008 global food crisis pushed 2.5 million households below the poverty line, with poor families' income declining by 36.7% [3], and 45% of households experienced food shortages during 2007-2009 [4].

The World Food Programme (WFP) plays a crucial role in monitoring food prices across Bangladesh, maintaining comprehensive databases that enable analysis of price trends, seasonal patterns, and crisis indicators [5]. This data serves as a valuable resource for researchers, policymakers, and humanitarian organizations working to address food security challenges. For this research, the study utilizes WFP data from the Dhaka Division covering four essential commodities: Lentils (masur), Palm Oil, Rice (coarse), and Wheat Flour—staples that together represent the core of the Bangladeshi diet. Figure 1.1 illustrates the price landscape, revealing several critical patterns: the 2010-2011 crisis with food inflation reaching 14.11%; the 2019 onion

crisis where prices surged 900% following India's export ban [6]; and the 2022 global shock from the Russia-Ukraine conflict causing food inflation to reach 12.54%, the highest in 12 years. According to the IPC [7], nearly 9 million people faced acute food insecurity in early 2023, with 35 million (21%) experiencing chronic food insecurity. These diverse and unpredictable patterns present a significant forecasting challenge.

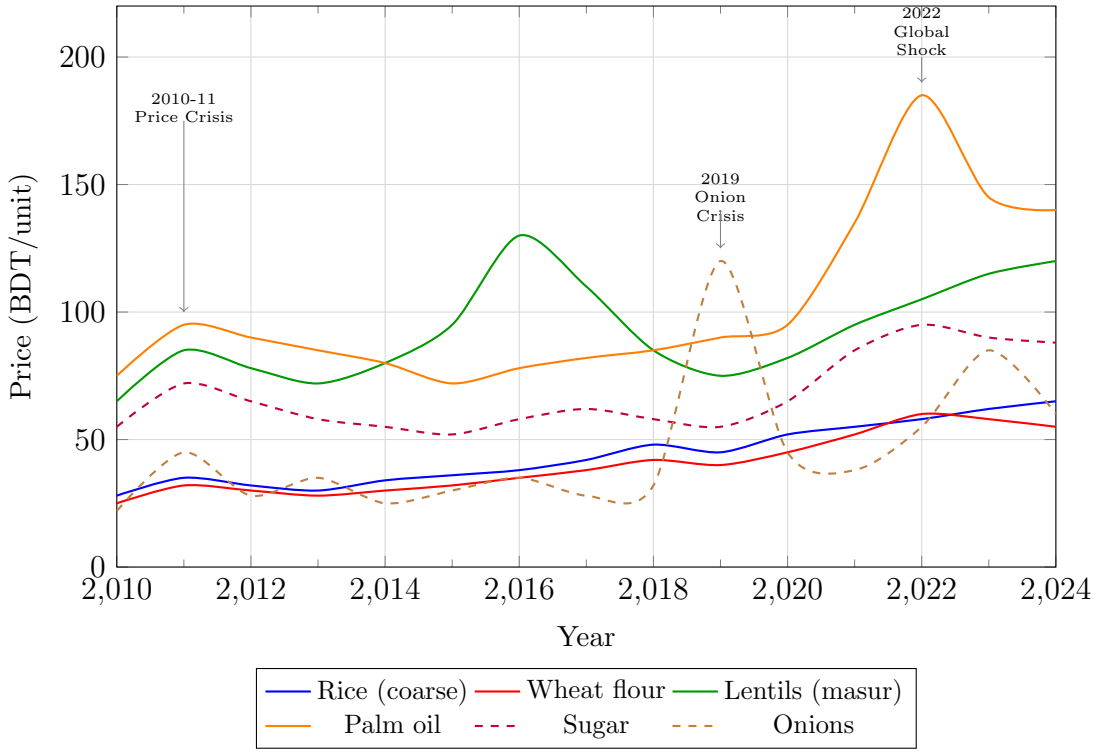

Figure 1.1: Food commodity price trends in Bangladesh from WFP monitoring data (2010–2024).

Traditional statistical methods like ARIMA [8] and exponential smoothing have been widely applied to commodity price prediction. While effective for capturing linear trends and seasonal patterns in stable environments, these classical approaches struggle with the non-linear relationships and long-range dependencies inherent in food prices—which are influenced by weather patterns, global market dynamics, policy changes, and supply chain disruptions. Deep learning has introduced more powerful alternatives: LSTM networks [9] and GRUs [10] capture temporal context through hidden states, while Transformer architectures [11] model relationships between any time points via attention mechanisms. Specialized time series models have achieved state-of-the-art results on forecasting benchmarks: N-BEATS [12] uses hierarchical residual learning for trend and seasonality decomposition, DLinear [13] employs simple linear decomposition challenging the need for complex architectures, and PatchTST [14] treats time series as sequences of patches enabling efficient capture of local and global patterns.

1.2 Motivation

Despite the impressive predictive capabilities of state-of-the-art deep learning models, their deployment in resource-constrained environments presents significant prac-

tical challenges. Many regions that would benefit most from accurate price forecasting—rural areas in developing countries, field offices of humanitarian organizations, and local government agencies—lack the computational infrastructure required to deploy and maintain complex neural network models. High-end GPUs, reliable electricity, and technical expertise for model maintenance are often unavailable in precisely the contexts where accurate food price predictions could have the greatest impact. This creates a critical gap between the availability of sophisticated forecasting techniques and their practical applicability where they are most needed.

Knowledge distillation [15] offers a promising solution to bridge this gap. Originally proposed for model compression in classification tasks, knowledge distillation involves training a smaller, more efficient "student" model to mimic the behavior of a larger, more accurate "teacher" model. The key insight is that the teacher's outputs contain richer information than hard labels alone—the relative probabilities assigned to different classes (or, in regression, the predicted values and intermediate representations) encode the teacher's learned understanding of the data. By training the student to match these "soft" targets, knowledge distillation enables the transfer of the teacher's expertise to a more compact architecture that can be deployed efficiently.

While knowledge distillation has been successfully applied in domains such as computer vision for image classification [16] and natural language processing for text understanding, its application to time series forecasting remains relatively underexplored. The regression nature of forecasting tasks, the temporal dependencies in the data, and the need to capture both point predictions and uncertainty present unique challenges that require adaptation of standard distillation techniques. Furthermore, ensemble methods that combine predictions from multiple diverse models often achieve better accuracy than any single model, but they multiply computational requirements and complicate deployment. Multi-teacher knowledge distillation [17], where a student learns from several teachers simultaneously, offers a way to capture the benefits of ensemble diversity while maintaining deployment efficiency.

The motivation for this study stems from the observation that combining these research directions—multi-teacher ensembles, knowledge distillation, and time series forecasting for food security—could address a real-world need for accurate yet deployable forecasting systems. By developing a framework that transfers knowledge from an ensemble of diverse, state-of-the-art teacher models to lightweight student networks, this research aims to democratize access to advanced forecasting capabilities for food security applications in resource-limited settings.

1.3 Problem Statement

The core challenge addressed in this study is the development of lightweight yet accurate forecasting models for food commodity prices that can be practically deployed in resource-constrained environments. This problem sits at the intersection of several research challenges that, when addressed together, can enable meaningful impact on food security monitoring and response. This work specifically confronts limitations encountered when attempting to apply modern deep learning to WFP Bangladesh data: the monthly frequency yields only around 200 observations per commodity over two decades, making standard deep learning approaches prone to overfitting; the target deployment contexts lack GPU infrastructure; and existing

knowledge distillation techniques designed for classification do not directly transfer to time series regression. The following points outline the specific challenges that this research aims to address:

1. **Deployment in Resource-Constrained Environments**: The target deployment environments for food price forecasting systems—field offices in developing regions, mobile applications for market monitoring, or edge devices for real-time alerts—often lack the computational resources required by state-of-the-art deep learning models. Discussions with WFP stakeholders revealed that many field offices operate with basic computing equipment, intermittent electricity, and limited internet connectivity. High-end GPUs, reliable power, and technical expertise for model maintenance are frequently unavailable in precisely the contexts where accurate food price predictions could have the greatest humanitarian impact. Running multiple large neural networks for ensemble predictions is typically infeasible in such settings, yet experiments confirmed that ensemble methods consistently outperform individual models by 15-25% in MAE. The fundamental challenge is achieving ensemble-level accuracy while deploying only a single, lightweight model that can run on standard CPU hardware without specialized infrastructure.

2. **Limited Data Availability**: Monthly commodity price data from WFP provides relatively few observations, typically spanning only a few decades and yielding on the order of hundreds of data points per commodity. In the experiments with Dhaka Division data, approximately 200 monthly observations per commodity are available—a stark contrast to the millions or billions of samples available in domains like computer vision or natural language processing. This data scarcity presents a fundamental challenge for deep learning models, which are known to be data-hungry and often require large datasets to learn robust representations without overfitting. Preliminary experiments revealed that complex architectures quickly memorize the training data, achieving near-zero training loss while performing poorly on held-out test sets. This necessitates approaches that can effectively leverage small datasets through appropriate model architectures, strong regularization, or knowledge transfer mechanisms that allow smaller models to benefit from the inductive biases learned by larger ones.

3. **Model Ensemble Complexity**: While ensemble methods that combine multiple forecasting models can significantly improve prediction accuracy by averaging out individual model errors and capturing diverse aspects of the data, they multiply computational requirements and complicate deployment logistics. In the proposed framework, training three teacher models (DLinear, PatchTST, N-BEATS) requires separate training runs, storage for three sets of model weights, and three forward passes during inference. Each model captures different aspects of the price dynamics—DLinear excels at linear trends, PatchTST captures long-range dependencies, and N-BEATS provides interpretable trend-seasonality decomposition—but deploying all three is impractical in resource-limited settings. The challenge lies in achieving ensemble-level accuracy while maintaining the deployment simplicity of a single model, which motivates the exploration of knowledge distillation as a compression technique

that can transfer the collective wisdom of multiple teachers into a compact student.

4. **Knowledge Transfer for Time Series Regression**: The majority of knowledge distillation research has focused on classification tasks, where the teacher's softmax outputs provide rich information about relationships between classes through soft probability distributions. In regression tasks like price forecasting, the outputs are single continuous values without the same structured probability information—there is no natural "soft label" equivalent. Furthermore, time series data contains temporal dependencies where the relationship between consecutive predictions matters, not just individual point accuracy. A model that predicts accurate values but fails to capture price movement directions provides less actionable information than one that tracks trends correctly. Experiments showed that naive application of regression distillation (simply matching student outputs to teacher outputs) provided only marginal improvements over supervised learning. This motivated the development of multi-component distillation losses that capture prediction values, intermediate feature representations, and price change dynamics—each addressing a different aspect of the knowledge that teachers encode.

1.4 Research Objectives

This study aims to develop and validate a multi-teacher knowledge distillation framework for food commodity price forecasting that addresses the challenges outlined above. The overarching goal is to produce a lightweight forecasting model that can be practically deployed in humanitarian field offices with limited computational resources, while maintaining accuracy comparable to sophisticated ensemble methods. Through iterative experimentation, it was discovered that effective knowledge transfer requires careful attention to loss function design, teacher selection, and uncertainty-aware weighting—insights that shaped the research objectives. The following objectives are designed to systematically address each component of the problem, from deployment requirements through empirical validation:

1. **Achieve Lightweight Field Deployment**: The primary objective is to produce a compact student model that can run inference on standard CPU hardware in sub-millisecond time, without requiring GPU infrastructure, cloud connectivity, or specialized technical expertise. This addresses the critical barrier that has prevented sophisticated forecasting techniques from reaching humanitarian field offices where they are most needed. The target is a model small enough to be distributed and maintained by non-technical staff, yet accurate enough to provide actionable price forecasts for procurement planning.

2. **Design a Multi-Teacher Knowledge Distillation Framework**: To achieve the deployment objective while maintaining accuracy, this study develops a systematic approach for transferring knowledge from an ensemble of diverse teacher models to the lightweight student network. This involves creating mechanisms for combining predictions and representations from multiple teachers, each potentially capturing different aspects of the temporal patterns

in price data. Early experiments revealed that simply averaging teacher predictions provided inconsistent results—some teachers performed poorly on specific commodities, degrading ensemble quality. This motivated the development of uncertainty-weighted aggregation that assigns higher weights to teachers with lower prediction variance and better validation performance on each commodity. The framework should be flexible enough to accommodate different numbers and types of teacher models while providing principled approaches for aggregating their contributions based on their reliability.

3. **Implement Diverse Teacher Architectures**: This study aims to implement and train three distinct teacher models—DLinear, PatchTST, and N-BEATS—that represent different paradigms in modern time series forecasting. These architectures were selected based on their complementary strengths and recent success on forecasting benchmarks. DLinear employs simple linear decomposition of trends and seasonality, offering computational efficiency and strong performance despite architectural simplicity—challenging the assumption that complex models are always necessary. PatchTST applies Transformer-based attention mechanisms to patches of the time series, capturing long-range dependencies that may exist between price movements months apart. N-BEATS uses hierarchical residual learning with explicit trend and seasonality stacks, providing both competitive accuracy and interpretable decomposition of price dynamics. By training these architecturally diverse teachers on the same WFP data, an ensemble is created whose members capture complementary aspects of price dynamics, increasing the richness of knowledge available for distillation.

4. **Develop Multi-Component Distillation Loss**: A critical objective is the design of a comprehensive loss function incorporating prediction-level matching, feature-level alignment, and price-difference learning to effectively transfer teacher knowledge to student models. Initial attempts using only prediction matching (MSE between student and teacher outputs) yielded limited improvements over supervised learning. The hypothesis was that teachers encode valuable information not just in their final predictions but also in their intermediate representations and their understanding of how prices change over time. This led to the four-component loss design: (1) hard loss maintaining alignment with ground truth to prevent error accumulation; (2) prediction distillation loss transferring teacher output knowledge; (3) feature distillation loss aligning intermediate representations; and (4) difference distillation loss capturing price change dynamics. Ablation studies confirmed that each component contributes to performance, with prediction distillation being most critical (removing it increases MAE by 0.62 BDT/unit).

5. **Evaluate on Real-World Data**: This study aims to conduct thorough experiments on WFP Bangladesh commodity price data from the Dhaka Division to validate the effectiveness of the proposed approach. The Dhaka Division represents the largest market in Bangladesh with prices that influence national trends, making it an ideal testbed. The evaluation covers four commodities—Lentils, Palm Oil, Rice (coarse), and Wheat Flour—that represent dietary staples with different price dynamics and volatility levels. The experiments en-

compass all 21 possible teacher-student configurations (7 teacher combinations $\times$ 3 student architectures), enabling comprehensive analysis of which combinations work best. Results are compared against supervised learning baselines trained without distillation and against the recent DSOF method [18] to contextualize the findings. Success is measured through multiple metrics (MAE, RMSE, MAPE, sMAPE, MASE) to provide a complete picture of forecasting accuracy.

6. **Analyze Contributing Factors**: Through systematic ablation studies, this study aims to understand which components of the framework contribute most significantly to performance improvements. Given the complexity of the multi-component approach, it is essential to identify which elements are critical versus peripheral. The ablation studies examine: the relative importance of each loss component (hard, prediction, feature, difference); the value of multi-teacher versus single-teacher distillation; the impact of uncertainty weighting versus simple averaging; and per-commodity analysis of teacher contributions. These analyses not only validate the design choices but provide actionable guidance for practitioners adapting the framework to new contexts—if computational budget is limited, which components should be prioritized?

1.5 Contributions

This study makes several contributions to the fields of knowledge distillation, time series forecasting, and food security applications. The primary contribution is a lightweight forecasting model that can be practically deployed in humanitarian field offices with limited computational resources—addressing a critical gap between sophisticated deep learning research and real-world humanitarian applications. This is achieved through methodological innovations in multi-teacher knowledge distillation that together advance both the research frontier and practical applicability. The following points summarize the key contributions of this work:

1. **Lightweight Deployment-Ready Architecture**: The framework produces a compact student model with only 200K parameters (compared to over 1M in the teacher ensemble) that achieves sub-millisecond inference on standard CPU hardware without GPU acceleration. This enables practical deployment in humanitarian field offices without requiring specialized computing infrastructure, cloud connectivity, or technical expertise for model maintenance—addressing the critical barrier that has prevented sophisticated forecasting techniques from reaching the contexts where they are most needed. The knowledge distillation paradigm front-loads computational complexity to a one-time offline training phase at a central location, then distributes the lightweight student for efficient field deployment.

2. **Novel Multi-Teacher Distillation Framework for Time Series**: This study proposes a comprehensive framework that combines predictions from multiple diverse teacher architectures—DLinear, PatchTST, and N-BEATS—using uncertainty-weighted ensemble strategies, enabling effective knowledge transfer to the compact student model. While prior work has explored knowledge distillation primarily in classification contexts with softmax-based soft

labels, the proposed framework addresses the unique challenges of continuous-valued time series forecasting. The framework introduces mechanisms that capture both the point predictions and the temporal dynamics learned by teacher models, going beyond simple output matching. The multi-teacher approach allows the student to benefit from the complementary strengths of different architectural paradigms: DLinear's linear trend modeling, PatchTST's attention-based long-range dependency capture, and N-BEATS's interpretable decomposition. Experiments demonstrate that the three-teacher ensemble consistently outperforms any single teacher or two-teacher combination, validating the value of architectural diversity.

3. **Multi-Component Distillation Loss**: This study introduces a novel loss function comprising four complementary components specifically designed for time series forecasting tasks. The hard loss ($\lambda_{hard} = 0.30$) maintains alignment with ground truth labels, preventing the student from merely copying teacher errors and ensuring the model learns from actual price data. The prediction distillation loss ($\lambda_{kd} = 0.60$) transfers the teachers' output-level knowledge, allowing the student to benefit from the teachers' learned representations. The feature distillation loss ($\lambda_{feat} = 0.15$) aligns intermediate representations through projection layers, encouraging the student to develop similar internal representations of temporal patterns despite having fewer parameters. The difference distillation loss ($\lambda_{diff} = 0.10$) specifically targets the prediction of price changes between consecutive time steps, capturing the dynamic nature of the forecasting task that pure point prediction misses. Through extensive ablation studies, the experiments demonstrate that each component contributes meaningfully to overall performance, with prediction distillation being most critical.

4. **Uncertainty-Weighted Knowledge Transfer**: This study develops a mechanism that weights teacher contributions based on their prediction confidence and validation performance, focusing student learning on reliable teacher outputs. Not all teachers perform equally well on all samples or commodities—PatchTST excels for Lentils and Rice while N-BEATS performs better for Wheat Flour. Naive averaging would dilute the best teacher's knowledge with inferior predictions from others. The uncertainty-weighted approach uses teacher prediction variance as a confidence measure and incorporates per-commodity validation performance to adaptively adjust weights. This ensures that the student receives stronger guidance from the most trustworthy teacher for each prediction context. Per-commodity analysis reveals the learned weight distributions, validating that the mechanism successfully identifies and emphasizes the best-performing teachers.

Together, these contributions enable a 37% improvement in MAE over supervised learning baselines on WFP Bangladesh data, achieving 3.73% MAPE. The framework also demonstrates 69% improvement over traditional ARIMA forecasting and 81% improvement over standard LSTM baselines, validating that multi-teacher knowledge distillation can effectively transfer ensemble-level accuracy to lightweight models suitable for resource-constrained humanitarian field deployment.

1.6 Document Outline

The remainder of this document is organized as follows. Chapter 2 provides a comprehensive survey of related work spanning deep learning architectures for time series forecasting, knowledge distillation techniques, and food price forecasting applications, establishing the research context and identifying the gaps that this work addresses. Chapter 3 presents the proposed multi-teacher knowledge distillation framework in detail, including the data preprocessing pipeline, teacher and student architectures, the multi-component loss function, uncertainty-weighted teacher ensemble, and the training procedure with mathematical formulations. Chapter 4 presents the experimental results and analysis, beginning with the experimental setup (hardware, software, dataset configuration, and training procedures), followed by evaluation metrics, comprehensive results comparing all 21 teacher-student configurations, ablation studies, and baseline comparisons including ARIMA, LSTM, AUNET, and the DSOF method. Chapter 5 concludes the study by synthesizing the contributions, analyzing why the framework achieves strong performance, discussing practical implications for humanitarian deployment, acknowledging limitations, and outlining directions for future research.

Chapter 2

Literature Review

This chapter provides a comprehensive review of the relevant literature spanning three interconnected research domains: deep learning architectures for time series forecasting, knowledge distillation techniques for model compression, and food price forecasting applications. The chapter traces the evolution of each field from foundational methods to state-of-the-art approaches, highlighting the theoretical underpinnings, architectural innovations, and practical considerations that inform the proposed framework. The chapter concludes by identifying the research gaps that motivate this study and positioning the contribution within the broader landscape of time series forecasting and knowledge transfer research.

2.1 Time Series Forecasting: Evolution from Statistical to Deep Learning

Time series forecasting has evolved significantly over the past several decades, progressing from classical statistical methods through machine learning approaches to modern deep learning architectures. Understanding this progression is essential for appreciating the design choices in contemporary forecasting systems and identifying opportunities for improvement through techniques like knowledge distillation.

Classical statistical approaches established the theoretical foundations of time series analysis. The Autoregressive Integrated Moving Average (ARIMA) model, introduced by Box and Jenkins [8], combines three components: autoregression (AR) using past values to predict future values, integration (I) applying differencing to achieve stationarity, and moving average (MA) modeling the relationship between observations and residual errors. ARIMA models excel when data exhibits linear patterns and stationary behavior after differencing. Seasonal extensions (SARIMA) incorporate periodic components common in economic and commodity data, capturing patterns like monthly or yearly cycles. Exponential smoothing methods, including Holt's linear trend method and Holt-Winters seasonal method, provide interpretable forecasts by assigning exponentially decreasing weights to historical observations. These methods perform well when data exhibits clear trend and seasonal structure, but they assume that future patterns will resemble historical ones—an assumption that often fails during crisis periods like the 2007-2008 food price spike or the 2022 global supply chain disruptions.

The limitations of statistical methods become apparent when facing complex, non-

linear dynamics characteristic of real-world commodity prices. Food prices are influenced by multiple interacting factors—weather patterns affecting crop yields, exchange rate fluctuations impacting import costs, policy changes such as export bans, and global supply chain disruptions—creating non-linear relationships that linear models cannot capture. Machine learning methods like Support Vector Regression (SVR), Random Forests, and Gradient Boosting introduced greater flexibility through their ability to model non-linear mappings between features and targets. However, these approaches require extensive feature engineering to extract relevant temporal information (lagged values, rolling statistics, calendar features) from raw time series data, shifting the burden of pattern discovery from the algorithm to the practitioner. The M4 forecasting competition [19] demonstrated that while traditional statistical methods remained competitive on many datasets, machine learning approaches showed promise for handling complex, heterogeneous time series patterns.

Deep learning has emerged as the dominant paradigm for time series forecasting by enabling automatic feature extraction and end-to-end learning from raw sequential data. Rather than requiring hand-crafted features, deep neural networks learn hierarchical representations that capture both local patterns and long-range dependencies. This shift has been particularly impactful for domains where the relevant features are not obvious a priori or where different datasets may benefit from different feature representations. Table 2.1 summarizes the evolution of forecasting approaches and their characteristics.

Table 2.1: Evolution of time series forecasting approaches from statistical methods to deep learning.

Era	**Methods**	**Strengths**	**Limitations**
Statistical (1970s–1990s)	ARIMA, SARIMA, Exponential Smoothing	Mathematical foundations, interpretability, works with limited data	Linear assumptions, manual model selection
Machine Learning (2000s–2010s)	SVR, Random Forest, Gradient Boosting	Non-linear relationships, ensemble diversity	Requires feature engineering, limited temporal modeling
Early Deep Learning (2014–2018)	LSTM, GRU, Seq2Seq	Automatic feature learning, sequential processing	Vanishing gradients, sequential bottleneck
Transformer Era (2019–2022)	Informer, Autoformer, FEDformer, PatchTST	Long-range dependencies, parallelizable	Quadratic complexity, computational overhead
Efficient Architectures (2023–Present)	DLinear, N-BEATS, TimeFuse	Task-specific design, efficiency, model fusion	Architecture-specific limitations

2.2 Recurrent Neural Networks for Sequential Data

Recurrent Neural Networks (RNNs) introduced the fundamental idea of maintaining internal state to process sequential data, where each time step's output depends not only on the current input but also on information accumulated from previous steps. The basic RNN processes a sequence by applying the same transformation at each step while updating a hidden state that serves as memory. Mathematically, at time step t, the hidden state h_t is computed as:

$$h_t = \tanh(W_{xh}x_t + W_{hh}h_{t-1} + b_h) \tag{2.1}$$

where x_t is the input, W_{xh} and W_{hh} are learned weight matrices, and the hyperbolic tangent provides non-linearity. This recurrence creates a chain of dependencies allowing information to flow through time, but it also creates the vanishing and exploding gradient problems that limit learning of long-range dependencies.

Long Short-Term Memory (LSTM) networks [9] addressed these limitations through a carefully designed gating mechanism that controls information flow. The LSTM cell maintains both a hidden state h_t and a cell state c_t, with three gates regulating their updates. The forget gate decides what information to discard from the cell state; the input gate controls what new information to store; and the output gate determines what aspects of the cell state to expose as output:

$$f_t = \sigma(W_f[h_{t-1}, x_t] + b_f) \tag{2.2}$$
$$i_t = \sigma(W_i[h_{t-1}, x_t] + b_i) \tag{2.3}$$
$$o_t = \sigma(W_o[h_{t-1}, x_t] + b_o) \tag{2.4}$$

The cell state update combines forgetting old information and adding new candidate values: $c_t = f_t \odot c_{t-1} + i_t \odot \tilde{c}_t$. This architecture creates gradient highways through the cell state, enabling learning over hundreds of time steps—a significant improvement over basic RNNs that typically failed beyond 10-20 steps.

Gated Recurrent Units (GRUs) [10] provide a simplified alternative that combines the forget and input gates into a single update gate and merges the cell and hidden states. Despite having fewer parameters, GRUs achieve comparable performance to LSTMs on many sequence modeling tasks while offering faster training due to reduced computational complexity. The choice between LSTM and GRU often depends on the specific application and dataset characteristics, with neither architecture consistently dominating across all domains.

For forecasting applications, encoder-decoder architectures extend recurrent networks to sequence-to-sequence prediction. The encoder processes the historical input sequence and compresses it into a fixed-dimensional context vector, while the decoder generates the forecast sequence conditioned on this context. Attention mechanisms [11] enhance this framework by allowing the decoder to directly access all encoder hidden states rather than relying solely on the final context vector. At each decoding step, attention computes a weighted combination of encoder states based on their relevance to the current prediction, enabling the model to focus on the most informative historical time points. DeepAR [20] demonstrated the effectiveness of autoregressive recurrent networks for probabilistic forecasting, modeling the full predictive distribution rather than just point estimates.

2.3 Transformer Architectures for Time Series

The Transformer architecture [11], originally developed for machine translation, replaced recurrence with self-attention, enabling parallel processing of entire sequences and direct modeling of dependencies between any pair of positions. Self-attention computes query, key, and value representations for each position and determines attention weights through scaled dot-product similarity:

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V \tag{2.5}$$

Multi-head attention applies this mechanism multiple times in parallel with different learned projections, capturing diverse patterns of relationships across the sequence. The computational complexity of self-attention is $O(L^2)$ where L is sequence length, which becomes prohibitive for very long sequences but enables rich modeling of temporal dependencies.

Adapting Transformers to time series forecasting requires addressing several domain-specific challenges. Unlike natural language where tokens are discrete and position-independent, time series values are continuous, ordered, and exhibit strong local correlations. Several architectural innovations have emerged to address these considerations. Informer [21] introduced ProbSparse attention that selects the most active queries based on entropy measurements, reducing complexity to $O(L \log L)$ while maintaining modeling capacity for long sequences. Autoformer [22] replaced attention with an auto-correlation mechanism that operates in the frequency domain to capture periodic patterns efficiently. FEDformer [23] combined frequency-domain analysis with seasonal-trend decomposition, applying attention in the Fourier space to exploit the sparsity of time series in frequency representations.

PatchTST [14] introduced a patching strategy that segments the input time series into fixed-length patches before applying Transformer encoders. This approach serves multiple purposes: it reduces the effective sequence length for attention computation from L to L/P where P is patch length; it preserves local semantic information within patches similar to how image patches capture local visual patterns; and it enables the model to learn patch-level representations that capture meaningful temporal segments. The paper demonstrated that, as the authors put it, "a time series is worth 64 words," achieving state-of-the-art results on multiple long-term forecasting benchmarks while reducing computational costs compared to point-wise Transformer variants.

Pyraformer [24] proposed a pyramidal attention mechanism with complexity linear in sequence length, organizing attention patterns hierarchically to capture both fine-grained local dependencies and coarse global patterns. Temporal Fusion Transformer (TFT) [25] combined multiple attention mechanisms for different types of inputs—static covariates, known future inputs, and observed past inputs—providing both competitive forecasting performance and interpretability through attention weight visualization.

Recent work has begun exploring foundation models for time series, following the success of large pre-trained models in natural language and vision. These approaches train on massive collections of diverse time series and aim to generalize to new forecasting tasks with minimal fine-tuning [26]. TimeGPT and similar decoder-only architectures [27] apply the GPT paradigm to time series, treating forecasting as

next-token prediction after discretizing continuous values. While promising, these approaches require substantial computational resources for training and may not outperform specialized architectures on domain-specific datasets with limited data—a consideration particularly relevant for humanitarian applications where data is scarce.

Table 2.2 provides a detailed comparison of Transformer-based forecasting architectures, highlighting their attention mechanisms, computational complexity, and key innovations.

Table 2.2: Comparison of Transformer-based time series forecasting architectures.

Model	**Attention Type**	**Complexity**	**Key Innovation**
Informer [21]	ProbSparse	$O(L \log L)$	Query sparsity; generative decoder
Autoformer [22]	Auto-Correlation	$O(L \log L)$	Series decomposition; frequency-domain
FEDformer [23]	Frequency-Enhanced	$O(L)$	Fourier/Wavelet; frequency attention
Pyraformer [24]	Pyramidal	$O(L)$	Hierarchical; multi-resolution
PatchTST [14]	Patch-based	$O((L/P)^2)$	Input patching; channel independence
TFT [25]	Multi-head	$O(L^2)$	Interpretable; variable selection

2.4 Specialized Time Series Architectures

While Transformer-based models have achieved impressive results, recent research has questioned whether their architectural complexity is necessary for time series forecasting. This line of inquiry has produced specialized architectures that achieve competitive or superior performance with simpler designs, often with significant computational efficiency gains.

N-BEATS (Neural Basis Expansion Analysis for Interpretable Time Series Forecasting) [12] introduced a deep neural architecture specifically designed for univariate time series forecasting. The architecture consists of a hierarchical stack of fully-connected residual blocks, where each block generates both a backcast (reconstruction of the input lookback window) and a forecast (prediction for the horizon). The key innovation is the doubly residual architecture: within each block, the network learns to predict basis expansion coefficients that generate the backcast and forecast outputs; across blocks, the input to each subsequent block is the residual not explained by previous blocks. This progressive refinement allows different blocks and stacks to specialize in capturing different signal components. The interpretable configuration constrains blocks to use trend and seasonality basis functions, providing explicit decomposition of forecasts into interpretable components. The generic configuration uses learnable basis functions for maximum flexibility. N-BEATS achieved state-of-the-art results on the M4 competition benchmark [19], demonstrating that well-designed fully-connected architectures can compete with more complex recurrent and attention-based models.

Building on the N-BEATS foundation, AUNET (Attention-Based Unified Network) [28] extends the architecture by integrating multi-head self-attention mechanisms within

the block structure. While N-BEATS relies solely on fully-connected layers to capture temporal dependencies, AUNET introduces attention layers that explicitly model relationships between different positions in the lookback window. The architecture maintains N-BEATS' modular, stack-based design where multiple sequential stacks generate partial forecasts, but enhances each block with attention mechanisms that allow the network to identify and weight the most relevant historical time points for each prediction. The authors demonstrated that this attention augmentation captures "complex temporal dependencies" that the original fully-connected blocks may miss, particularly in domains with irregular patterns or long-range correlations. Experiments on the CIMIS weather dataset showed substantial improvements: MAE of 0.3580, RMSE of 0.4632, MAPE of 0.68%, and R^2 score of 0.9988, with performance margins exceeding 87% improvement over baseline N-BEATS variants. Additional validation in finance (R^2 of 0.9990) and healthcare domains confirmed the generalizability of the attention-enhanced approach. Importantly, AUNET achieved these improvements while reducing training time and computational requirements (FLOPs) compared to the original N-BEATS, as the attention mechanism enabled more efficient information routing that compensated for the additional attention parameters.

DLinear [13] challenged the prevailing assumption that complex architectures are necessary for time series forecasting by demonstrating that simple linear models can match or exceed Transformer performance on many benchmarks. The architecture applies a two-stage process: first, it decomposes the input into trend and seasonal components using moving average operations; then, it applies separate linear layers to each component before combining them for the final prediction. Mathematically, for input x of length L and prediction horizon H, DLinear computes:

$$\hat{y} = W_{\text{trend}} \cdot x_{\text{trend}} + W_{\text{seasonal}} \cdot x_{\text{seasonal}} \tag{2.6}$$

where $W_{\text{trend}} \in \mathbb{R}^{H \times L}$ and $W_{\text{seasonal}} \in \mathbb{R}^{H \times L}$ are learned linear mappings. Despite its simplicity—DLinear has $O(L \times H)$ parameters compared to millions for Transformer variants—it achieved state-of-the-art results on several long-term forecasting benchmarks. The authors argued that the apparent success of Transformers on time series was partly due to suboptimal comparisons and that the inductive biases of attention mechanisms may not align well with the statistical properties of many time series datasets.

A significant recent development in time series forecasting is the recognition that no single model consistently outperforms others across all scenarios. TimeFuse [29], accepted at ICML 2025, addresses this observation through adaptive model fusion. The framework employs meta-features to characterize input time series and trains a learnable fusion component that predicts optimal weighting of heterogeneous forecasting models for each input. Rather than selecting a single best model or using fixed ensemble weights, TimeFuse learns to dynamically combine model predictions based on input characteristics. The key insight is that different models exhibit complementary strengths—one model may excel at capturing trends while another handles seasonality better—and intelligent fusion can exploit this diversity. The authors demonstrated "near-universal improvement over state-of-the-art individual models" across both long-term and short-term forecasting tasks. While TimeFuse focuses on input-dependent adaptive weighting through meta-features, this study takes a different approach using performance-based weighting derived from validation accuracy,

which is more suitable for the data-limited regime of food commodity forecasting where learning complex meta-feature mappings is challenging.

Online time series forecasting presents additional challenges related to concept drift and information leakage that batch methods do not address. Lau et al. [18] introduced a dual-stream framework for online forecasting that combines complementary learning strategies to handle dynamic environments. The key contribution is the recognition that traditional online forecasting settings often suffer from information leakage, where models inadvertently access future information during training or evaluation. To address this, the authors proposed a model-agnostic framework consisting of two streams: a "slow stream" that utilizes experience replay with complete historical data to maintain stable long-term patterns, and a "fast stream" that adapts to recent data through temporal difference learning to capture short-term dynamics. The framework implements a teacher-student architecture with residual learning, where the slow stream serves as a teacher providing stable predictions and the fast stream learns to predict residuals that capture recent distribution shifts. This dual-stream approach is particularly relevant to this study for two reasons: first, it demonstrates the effectiveness of teacher-student paradigms for time series forecasting beyond traditional classification settings; second, it addresses concept drift—a challenge highly relevant to food price forecasting where market conditions can shift rapidly during crises. The framework achieved strong results on multiple benchmarks while maintaining computational efficiency suitable for real-time applications.

Figure 2.1 illustrates the key architectural differences between the major forecasting paradigms discussed in this section.

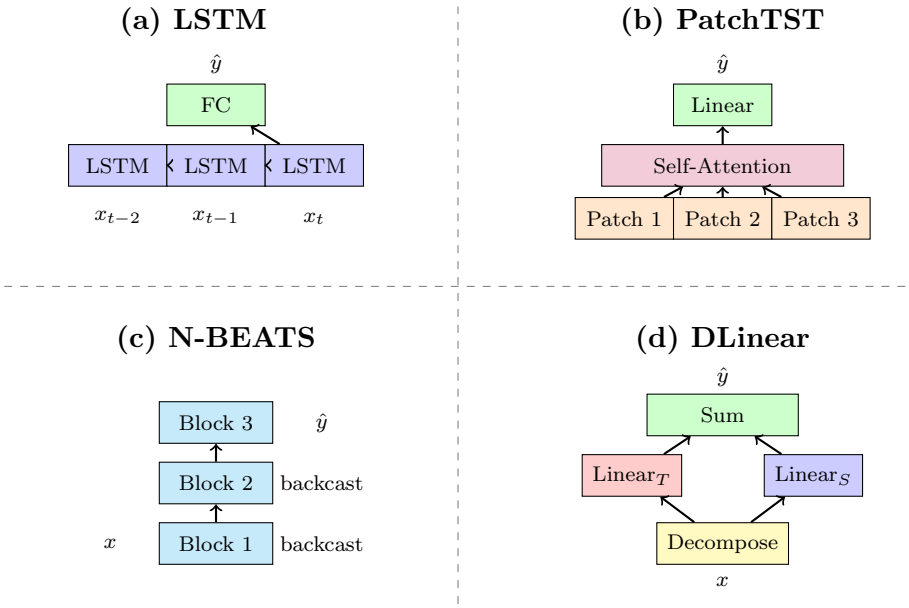

Figure 2.1: Architectural comparison of time series forecasting paradigms: (a) LSTM, (b) PatchTST, (c) N-BEATS, (d) DLinear.

2.5 Knowledge Distillation: Principles and Techniques

Knowledge distillation [15] is a model compression technique that transfers knowledge from a large, high-capacity "teacher" model to a smaller, more efficient "student" model. The fundamental insight is that the teacher's outputs—particularly the soft probability distributions over classes in classification tasks—contain richer information than the original hard labels. When a teacher assigns probability 0.7 to class A and 0.2 to class B, it encodes the learned understanding that these classes share some similarity, information that a hard label "class A" cannot convey. This "dark knowledge" about inter-class relationships helps the student learn more generalizable representations than training on hard labels alone.

The original formulation by Hinton et al. [15] introduced temperature scaling to soften the probability distribution. For a neural network producing logits z_i for class i, the softened probabilities are computed as:

$$p_i = \frac{\exp(z_i/T)}{\sum_j \exp(z_j/T)} \tag{2.7}$$

where temperature $T > 1$ produces softer distributions that reveal more information about the learned class relationships. The distillation loss combines the standard cross-entropy with hard labels and a term matching the student's soft predictions to the teacher's:

$$\mathcal{L} = (1-\alpha)\mathcal{L}_{CE}(y, p_s) + \alpha T^2 \mathcal{L}_{KL}(p_t, p_s) \tag{2.8}$$

where p_s and p_t are student and teacher soft predictions, $\mathcal{L}_{CE}$ is cross-entropy, $\mathcal{L}_{KL}$ is Kullback-Leibler divergence, and α balances the two objectives. The T^2 scaling compensates for the reduced gradient magnitudes at higher temperatures.

A comprehensive survey by Mansourian et al. [30] organizes knowledge distillation techniques across multiple dimensions: distillation sources (what knowledge to transfer), schemes (how to structure the teacher-student relationship), algorithms (optimization strategies), modalities (input types), and applications. The survey highlights that modern large models—foundation models, vision-language models, and large language models—face "high runtime and memory consumption" that makes knowledge distillation increasingly important for practical deployment. The authors provide extensive tabular comparisons and discuss techniques for distilling knowledge from diffusion models, 3D inputs, and transformer architectures, offering "a new point of view and representation structure" compared to earlier surveys.

Knowledge distillation techniques can be categorized by the source of transferred knowledge. Response-based distillation focuses on the final layer outputs, training the student to match the teacher's predictions. This is the original formulation and remains effective for many applications. For regression tasks like time series forecasting, this translates to minimizing the distance between student and teacher predictions:

$$\mathcal{L}_{response} = \frac{1}{N}\sum_{i=1}^{N}(f_s(x_i) - f_t(x_i))^2 \tag{2.9}$$

where f_s and f_t are student and teacher models. However, continuous outputs lack the structured probability information of classification, motivating the exploration of additional knowledge sources.

Feature-based distillation [16] extends the transfer to intermediate layer representations, encouraging the student to develop similar internal feature maps to the teacher. FitNets introduced "hint layers" where the student's intermediate representation is trained to match a designated teacher layer through an additional regressor network that handles dimension mismatches. This approach forces the student to learn similar abstractions to the teacher, potentially capturing knowledge that is not reflected in the final outputs. The feature distillation loss is:

$$\mathcal{L}_{feature} = \frac{1}{N}\sum_{i=1}^{N} \|g(h_s(x_i)) - h_t(x_i)\|^2 \tag{2.10}$$

where h_s and h_t are intermediate representations and g is a learned projection when dimensions differ.

Attention transfer [31] specifically targets attention maps as the knowledge source, based on the insight that where a model focuses its computation reveals important learned patterns. For attention-based architectures, this involves matching the spatial or temporal attention distributions between teacher and student.

Relational knowledge distillation [32], [33] preserves structural relationships between data points rather than individual point-wise outputs. The intuition is that how a teacher represents the relationships between different inputs—which samples are considered similar or dissimilar—encodes important learned structure. Similarity-preserving distillation encourages the student to maintain the same pairwise similarity structure in its embedding space as the teacher.

Table 2.3 summarizes the major knowledge distillation techniques and their characteristics.

Table 2.3: Taxonomy of knowledge distillation techniques by knowledge source.

Category	**Knowledge Source**	**Advantages**	**Challenges**
Response-Based [15]	Final layer outputs (logits/predictions)	Simple implementation; effective for classification	Limited information for regression tasks
Feature-Based [16]	Intermediate layer activations	Captures learned abstractions; deeper knowledge transfer	Requires dimension alignment
Attention-Based [31]	Attention maps and weights	Transfers learned focus patterns	Requires attention in both models
Relation-Based [32]	Pairwise sample relationships	Preserves structure; more robust	Quadratic complexity in batch size
Multi-Teacher [17]	Ensemble of teacher models	Captures complementary expertise	Needs effective weighting strategy

2.6 Multi-Teacher Knowledge Distillation

Extending knowledge distillation to multiple teachers offers the potential to capture diverse expertise from different model architectures, each potentially excelling at different aspects of the task. Multi-teacher distillation [17] addresses the question of how to effectively combine knowledge from an ensemble of teachers into a single student model.
The simplest approach averages teacher predictions uniformly:

$$\hat{y}_{ensemble} = \frac{1}{K} \sum_{k=1}^{K} f_{t_k}(x) \tag{2.11}$$

where K is the number of teachers. While straightforward, this ignores that different teachers may have different expertise levels on different inputs. A teacher specialized in capturing trends may provide more reliable guidance for trending data, while a seasonality-focused teacher offers better guidance for periodic patterns.
Learned teacher weighting assigns trainable weights to each teacher's contribution. The weights can be global (same for all inputs) or input-dependent, computed as a function of the input characteristics. The ensemble prediction becomes:

$$\hat{y}_{ensemble} = \sum_{k=1}^{K} w_k(x) \cdot f_{t_k}(x) \quad \text{where} \quad \sum_{k=1}^{K} w_k(x) = 1 \tag{2.12}$$

The weights $w_k(x)$ can be parameterized by a small neural network that takes input features and outputs teacher weights, enabling the system to learn which teacher to trust for which types of inputs.
Uncertainty-weighted distillation accounts for teacher confidence when combining predictions. Teachers that are uncertain about a prediction (high variance in outputs or explicit uncertainty estimates) should have less influence than confident teachers. For regression tasks, prediction variance across an ensemble or epistemic uncertainty estimates can serve as confidence measures. The proposed framework adopts this approach, weighting teacher contributions inversely proportional to their prediction uncertainty.
Recent theoretical work [34] has begun to explain why knowledge distillation and ensemble methods work. The authors show that ensembles achieve diversity through training on different random initializations, which causes models to learn different features from the data. Knowledge distillation then transfers this diversity of learned features to the student, even though the student is a single model. Self-distillation, where a model is distilled into a model of the same architecture, can also improve performance by regularizing learning toward a smoother, more generalizable function.

2.7 Knowledge Distillation for Time Series Forecasting

While knowledge distillation has been extensively studied for image classification and natural language processing, its application to time series forecasting presents unique challenges that require adaptation of standard techniques.

The regression nature of forecasting means that outputs are continuous values without the structured probability information that makes classification distillation effective. A teacher's predicted price of 45.2 BDT contains less relational information than a softmax distribution over classes. This motivates the use of multiple knowledge sources beyond response-based distillation—feature representations and temporal dynamics—to enrich the knowledge transfer signal.

Temporal dependencies in time series data mean that not just the predicted values but also the temporal patterns matter. A model that correctly predicts prices but fails to capture their dynamics (rising vs. falling trends, acceleration, periodicity) may be less useful than one with slightly higher point errors but correct directional predictions. This observation motivates the difference distillation component that explicitly targets the prediction of price changes.

Data scarcity in many forecasting domains, including humanitarian food security applications, limits the amount of supervised signal available for training. Knowledge distillation can serve as a form of regularization, with the teacher providing additional training signal that helps prevent overfitting to the limited ground truth data.

Limited prior work has explored knowledge distillation specifically for time series. Some research has applied distillation for anomaly detection in time series, compressing large anomaly detection models for edge deployment. Others have explored distilling Transformer-based forecasting models into simpler architectures to reduce inference latency. However, comprehensive multi-teacher frameworks for regression-based forecasting that address the unique challenges of time series remain scarce, representing a significant research gap that this study addresses.

Meta-learning approaches provide a complementary perspective on knowledge transfer for time series. Liu et al. [35] developed a meta-learning framework for zero-shot financial time series forecasting that addresses scenarios where historical data is scarce or markets experience sudden shifts. Their approach leverages learned embeddings for both meta-task construction and downstream predictions, using Gaussian Mixture Models to cluster embeddings and create training tasks that span both similar patterns (intra-cluster) and diverse patterns (inter-cluster). This dual task construction enables simultaneous learning of local adaptations and cross-series generalizations. Testing on volatile financial data and emerging markets demonstrated superior performance compared to existing methods, particularly in zero-shot scenarios where no target-domain training data is available. The meta-learning paradigm offers insights relevant to knowledge distillation: both approaches aim to transfer learned knowledge to new scenarios, and hybrid frameworks combining meta-learning with distillation may offer additional benefits.

2.8 Food Price Forecasting: Methods and Challenges

Food price forecasting has attracted significant research attention due to its implications for food security, agricultural policy, and humanitarian response. Accurate price predictions enable governments to implement timely interventions, help farmers make informed planting decisions, allow humanitarian organizations to preposition resources, and enable households to plan their food budgets.

Early work applied classical statistical methods to agricultural commodity prices. ARIMA and its seasonal variants captured linear trends and periodic patterns in crop prices, often incorporating exogenous variables such as weather data, exchange rates, and production statistics. Cointegration analysis modeled long-run equilibrium relationships between related commodity prices, useful for understanding how shocks to one commodity propagate through the food system. Vector Autoregression (VAR) models captured dynamic interactions between multiple commodity prices simultaneously. These methods provided interpretable results and worked well in stable market conditions, but struggled during crisis periods when historical patterns broke down.

Machine learning approaches introduced greater flexibility for handling non-linear price dynamics. Support Vector Regression proved effective for capturing complex relationships between input features and prices. Ensemble methods like Random Forests and Gradient Boosting provided robustness through model averaging and achieved competitive results on many agricultural forecasting tasks. However, these approaches required careful feature engineering—extracting lagged prices, rolling statistics, calendar features, and external indicators—and did not naturally handle the sequential nature of time series data.

Deep learning has shown promise for food price forecasting by automatically learning relevant features from raw price sequences. LSTM networks have been applied to various agricultural commodities including cereals, oilseeds, and livestock, demonstrating ability to capture complex price dynamics and outperform traditional methods on many datasets [36]. Weng et al. [37] incorporated climate factors into attention-based LSTM models for food price prediction, showing that external weather information can improve forecast accuracy for weather-sensitive commodities. Systematic reviews [38] have documented the growing application of machine learning and deep learning to food price forecasting, identifying best practices and remaining challenges.

Despite progress, several challenges persist in food price forecasting. Data scarcity remains a fundamental limitation: monthly price data for many commodities spans only a few decades, yielding hundreds rather than the thousands or millions of observations that deep learning typically requires. High volatility during crisis periods makes historical patterns unreliable predictors of future behavior. External shocks—geopolitical events, policy changes, natural disasters—create structural breaks that no model can anticipate from historical data alone. Multi-commodity interactions mean that prices of related goods (substitutes and complements) influence each other, requiring models that capture these relationships. Different market structures across regions create heterogeneity that complicates transfer of models trained on one context to another.

Table 2.4 summarizes the major approaches to food price forecasting and their characteristics based on the literature.

Table 2.4: Summary of food price forecasting approaches from the literature.

Approach	**Methods**	**Strengths**	**Limitations**
Statistical	ARIMA, VAR, Cointegration	Interpretable results, works with limited data	Linear assumptions limit flexibility
Machine Learning	SVR, Random Forest, XGBoost	Handles non-linear patterns, ensemble robustness	Requires extensive feature engineering
Recurrent Neural Networks	LSTM, GRU, Attention-LSTM [37]	Automatic feature learning, temporal modeling	Data-hungry, sequential processing
Hybrid Methods	ARIMA-LSTM [36]	Combines statistical and deep learning strengths	Complex architecture, difficult to tune
Transformer-Based	TFT, Custom architectures	Long-range dependencies, multivariate support	High computational cost

2.9 Model Compression for Resource-Constrained Environments

The practical deployment of deep learning models in resource-constrained environments—edge devices, mobile platforms, field offices with limited computing infrastructure—requires techniques for reducing model size and computational requirements while maintaining acceptable accuracy. Model compression encompasses several complementary approaches: knowledge distillation (transferring knowledge to smaller architectures), pruning (removing redundant parameters), quantization (using lower-precision arithmetic), and architecture design (creating inherently efficient structures) [39], [40].

Knowledge distillation, as discussed in Section 2.5, trains small student models to mimic larger teachers, achieving compression ratios of 10-100× with modest accuracy degradation for many tasks. The approach is architecture-agnostic: any student architecture can learn from any teacher, enabling flexible design of deployment-appropriate models.

Network pruning identifies and removes parameters that contribute little to model performance. Magnitude-based pruning removes weights with the smallest absolute values, based on the assumption that small weights have minimal impact on outputs. Structured pruning removes entire neurons, channels, or layers to achieve actual speedups on hardware that does not efficiently handle sparse operations. Iterative pruning with retraining achieves higher compression than one-shot pruning by allowing the network to adapt to reduced capacity gradually.

Quantization reduces the numerical precision of weights and activations from 32-bit floating point to 16-bit, 8-bit, or even lower. Post-training quantization applies precision reduction to a trained model with minimal calibration data. Quantization-aware training simulates quantization effects during training, producing models more ro-

bust to precision reduction. Deep compression [40] combined pruning, quantization, and Huffman coding to achieve 35-49× compression of VGG and AlexNet models with negligible accuracy loss.

Efficient architecture design creates models that are inherently suited for resource-constrained deployment. MobileNets introduced depthwise separable convolutions that factorize standard convolutions into a depthwise and pointwise operation, reducing computation while maintaining capacity. EfficientNet systematically studied the relationship between network depth, width, and resolution, providing compound scaling rules for efficient architecture design. For time series, DLinear demonstrates that extremely simple architectures can achieve competitive performance, representing an extreme form of efficient design.

For humanitarian applications like food price forecasting, the combination of knowledge distillation with efficient student architectures offers particular promise. The teachers can be complex models trained on available computational resources (cloud servers, research computing), while the students are designed for the target deployment environment (laptops in field offices, mobile devices). The proposed framework adopts this paradigm, training DLinear, PatchTST, and N-BEATS teachers and distilling their knowledge into MLP, GRU, and KAN students suitable for resource-constrained deployment.

2.10 Research Gaps and Positioning

The literature review reveals several research gaps at the intersection of time series forecasting, knowledge distillation, and food security applications:

1. **Knowledge Distillation for Regression**: While multi-teacher distillation has succeeded in classification tasks where softmax outputs provide rich "dark knowledge" [15], its application to regression remains underexplored. Time series forecasting produces continuous values without structured probabilistic information, necessitating new formulations beyond simple output matching.

2. **Performance-Based Teacher Weighting**: Existing multi-teacher approaches often use equal weighting across teachers, ignoring that teachers exhibit varying performance levels. Performance-based weighting that assigns higher influence to more accurate teachers, combined with uncertainty estimation that downweights predictions where teachers disagree, is needed for robust knowledge transfer.

3. **Feature Alignment Across Architectures**: Feature-based distillation faces challenges when applied to heterogeneous time series models—LSTMs use cell states, Transformers produce attention-weighted representations, and N-BEATS generates residual signals. Aligning these diverse representations requires careful design.

4. **Food Security Applications**: Despite the critical importance of price forecasting for humanitarian planning and severe resource constraints in field deployment, knowledge distillation has not been systematically applied to create lightweight models for food commodity forecasting.

5. **Systematic Configuration Analysis**: Prior research typically evaluates single teacher-student pairs without exploring the combinatorial space of configurations, leaving practitioners without guidance on optimal teacher selection and student architecture choices.

Recent advances motivate addressing these gaps. TimeFuse [29] demonstrates that combining diverse forecasting models through intelligent fusion achieves "near-universal improvement" over individual models, validating the value of architectural diversity. The meta-learning approach of Liu et al. [35] shows knowledge transfer is crucial for handling data scarcity. The dual-stream framework [18] validates teacher-student architectures for time series forecasting. Building on these insights, this study proposes a multi-teacher knowledge distillation framework that addresses the regression challenge through multi-component loss functions, performance-based teacher weighting with uncertainty-informed loss computation, and comprehensive evaluation of teacher-student configurations.
Figure 2.2 illustrates the positioning of this work at the intersection of these research areas.

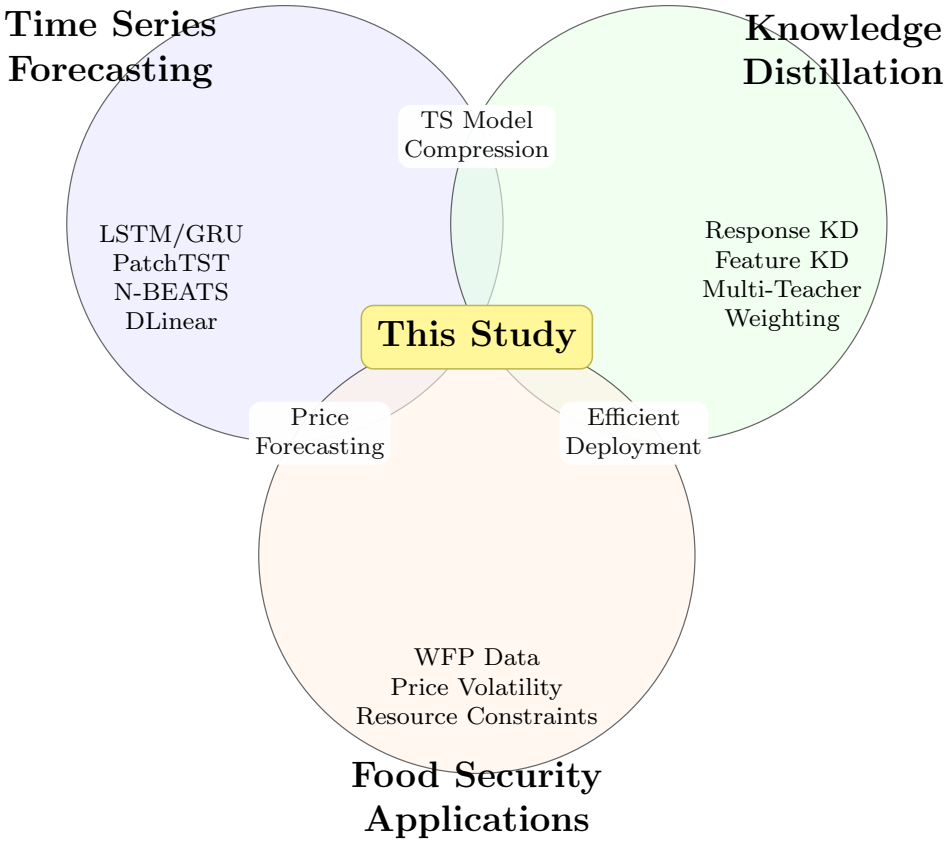

Figure 2.2: Research positioning at the intersection of time series forecasting, knowledge distillation, and food security applications.

This study addresses these gaps by proposing a comprehensive multi-teacher knowledge distillation framework specifically designed for food commodity price forecasting. The framework combines diverse teacher architectures (DLinear, PatchTST, N-BEATS), multi-component loss functions (hard, prediction, feature, and difference losses), uncertainty-weighted knowledge transfer, and thorough empirical evaluation on real-world WFP Bangladesh data from the Dhaka Division. The student

architectures (MLP, GRU, KAN) are designed for efficient deployment in resource-constrained humanitarian field offices while maintaining competitive forecasting accuracy. The following chapter presents the methodology in detail, including the data preprocessing pipeline, model architectures, and the complete knowledge distillation training procedure.

Chapter 3

Methodology

This chapter presents the methodology of the proposed multi-teacher knowledge distillation framework for food commodity price forecasting. The chapter begins with a comprehensive system overview, followed by the dataset characteristics and preprocessing steps, the three teacher model architectures (DLinear, PatchTST, and N-BEATS), the three student model architectures (MLP, GRU, and KAN), and finally the knowledge distillation framework with its multi-component loss functions and uncertainty-weighted ensemble strategies.

3.1 System Overview

The proposed framework addresses the challenge of deploying accurate food price forecasting models in resource-constrained humanitarian settings. While state-of-the-art deep learning models achieve excellent performance, their computational requirements often exceed what is available in field offices of humanitarian organizations. The framework resolves this by training powerful teacher models offline, then distilling their collective knowledge into lightweight student models suitable for basic hardware. As illustrated in Figure 3.1, the framework operates through a four-phase pipeline:

1. **Data Preprocessing**: Raw World Food Programme (WFP) price data undergoes systematic transformation including unit normalization, temporal aggregation to monthly frequency, chronological train/validation/test partitioning, MinMax scaling, and sliding window creation for supervised learning.

2. **Teacher Model Training**: Three architecturally diverse teacher models are trained independently. DLinear captures linear trends and seasonal decomposition, PatchTST leverages Transformer attention for long-range dependencies, and N-BEATS employs hierarchical residual learning for multi-scale pattern extraction. This diversity ensures complementary temporal pattern capture.

3. **Uncertainty-Weighted Ensemble and Loss Computation**: Teacher predictions are combined using temperature-scaled softmax weighting based on validation MAE, with prediction variance providing uncertainty estimates. The multi-component distillation loss combines hard loss, prediction distillation, feature distillation, and difference distillation terms.

4. **Student Model Distillation**: Compact student models (MLP, GRU, or KAN) are trained using combined supervision from teachers and ground truth. Students receive stronger guidance where teachers agree and achieve near-teacher accuracy with a fraction of the parameters—approximately 200K versus over 1M in the teacher ensemble.

This four-phase design enables deployment on standard laptops or mobile devices, making accurate food price forecasting accessible to field offices without specialized computing infrastructure.

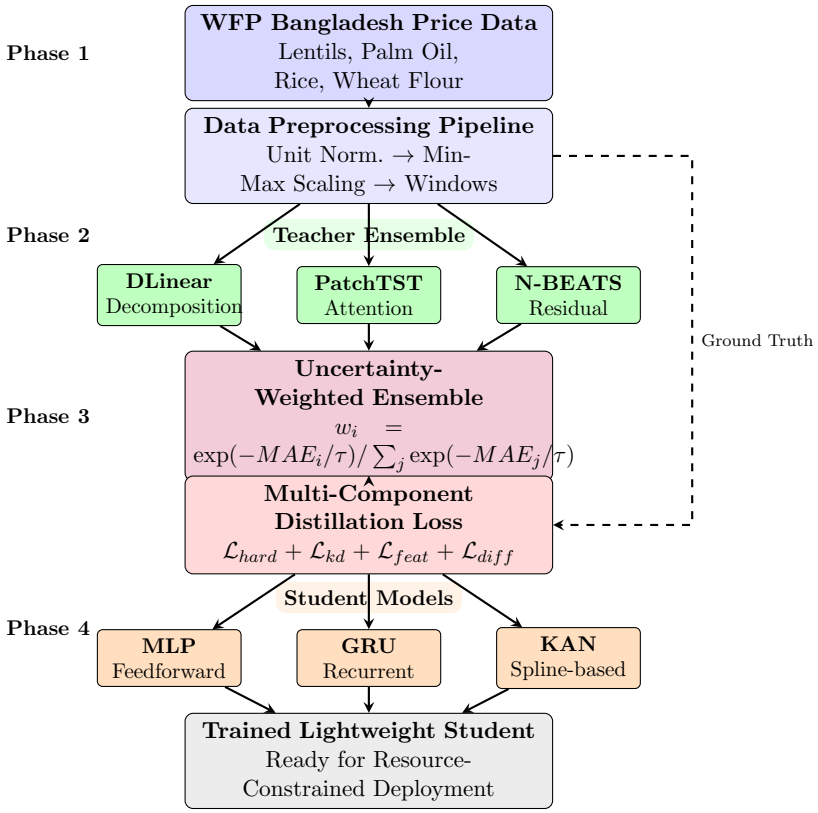

Figure 3.1: System architecture of the multi-teacher knowledge distillation framework.

3.2 Dataset Description

3.2.1 Data Source

This research utilizes the World Food Programme (WFP) Food Prices dataset for Bangladesh [5], which provides monthly retail prices for various food commodities across multiple markets. The WFP maintains this database as part of its global food security monitoring efforts, collecting price data from local markets through a network of field monitors. The dataset is publicly available through the Humanitarian Data Exchange (HDX) platform and is regularly updated, making it suitable for reproducible research on food price forecasting.

The WFP data collection methodology ensures consistency across time periods and markets. Prices are collected from representative retail outlets, with multiple ob-

servations aggregated to produce reliable monthly estimates. This standardized approach provides high-quality time series data suitable for machine learning applications.

3.2.2 Market and Commodity Selection

This study focuses on the Dhaka Division, Bangladesh's capital region and largest urban center, which provides the most comprehensive price coverage and represents the primary market influencing national food security conditions. The Dhaka market serves as a price leader for the country, with fluctuations in Dhaka prices typically propagating to other regions within one to two months.
To ensure sufficient data for training deep learning models while maintaining practical relevance for food security applications, this study selects four essential commodities that represent the core of the Bangladeshi diet. Table 3.1 summarizes the dataset characteristics for each selected commodity.

Table 3.1: Dataset statistics for selected commodities from Dhaka Division

Commodity	**Category**	**Observations**	**Price Range (BDT/kg)**	**Volatility**
Lentils (masur)	Pulses	96	65–130	High
Oil (palm)	Oils	94	72–185	Very High
Rice (coarse)	Cereals	92	28–65	Moderate
Wheat flour	Cereals	88	25–60	Moderate

These commodities were selected based on three criteria: (1) sufficient historical observations for model training, (2) dietary importance for food security assessment, and (3) diversity in price dynamics to test model robustness. Lentils provide essential protein for vegetarian households, palm oil is the primary cooking fat, rice is the staple carbohydrate, and wheat flour is used for bread and other preparations.

3.3 Data Preprocessing

The data preprocessing pipeline transforms raw WFP price data into a format suitable for deep learning models. Figure 3.2 illustrates the complete preprocessing workflow.

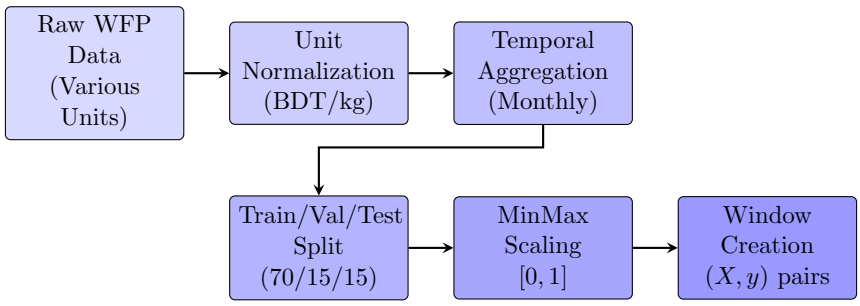

Figure 3.2: Data preprocessing pipeline from raw WFP data to model-ready input-output pairs.

3.3.1 Data Cleaning and Standardization

The raw WFP dataset presents several challenges that must be addressed before model training. Prices are recorded in various units depending on commodity type and local measurement conventions—some entries report prices per kilogram, others per 100 grams, and oils may be recorded per liter. Additionally, price observations occur at irregular intervals, with some markets reporting weekly while others report bi-weekly or monthly. These inconsistencies must be resolved to create a homogeneous dataset suitable for deep learning models.

The first preprocessing step performs unit normalization to convert all prices to a consistent unit of Bangladeshi Taka (BDT) per unit. Following unit normalization, temporal aggregation consolidates observations to a consistent monthly frequency. The WFP data collection methodology involves multiple market visits per month, potentially yielding several observations for the same commodity-market pair. This study aggregates these observations using the median rather than the mean:

$$p_{month} = \text{median}(\{p_i : t_i \in \text{month}\}) \tag{3.1}$$

The median provides robustness against outliers that may arise from data entry errors, unusual market conditions, or transient price spikes that do not reflect underlying market trends. Monthly aggregation aligns with the typical planning horizon for humanitarian food security interventions—field offices typically make procurement and distribution decisions on monthly cycles. This temporal resolution also provides sufficient observations for model training (approximately 88–96 months per commodity) while smoothing out daily or weekly fluctuations that are less relevant for strategic planning.

3.3.2 Dataset Partitioning and Normalization

Proper dataset partitioning is crucial for realistic evaluation of time series forecasting models. Unlike standard machine learning tasks where random sampling is appropriate, time series data exhibits temporal dependencies that require chronological splitting to prevent data leakage. This study employs a time-based 70/15/15 split:

- **Training set** (first 70% of observations): Used for model parameter optimization through gradient descent. For a commodity with 96 observations, this yields approximately 67 training months.
- **Validation set** (next 15%): Used for hyperparameter tuning, early stopping decisions, and model selection. This set serves as a proxy for out-of-sample performance during development, providing approximately 14 months.
- **Test set** (final 15%): Reserved exclusively for unbiased final evaluation after all model development is complete. The test set is never used during training or hyperparameter selection, ensuring reported metrics reflect true generalization performance.

The chronological split ensures that models are evaluated on genuinely future data, simulating the real-world deployment scenario where predictions must be made for unseen time periods. This is particularly important for food price forecasting, where

the goal is to predict prices that have not yet been observed rather than to interpolate between known values.
After partitioning, MinMax normalization is applied to scale prices to the $[0, 1]$ range:

$$x_{scaled} = \frac{x - x_{min}}{x_{max} - x_{min}} \tag{3.2}$$

Normalization is essential for neural network training stability, as it prevents large price values from dominating gradient updates and ensures all commodities contribute equally regardless of their absolute price levels.

3.3.3 Sliding Window Transformation

The final preprocessing step transforms the normalized time series into input-output pairs suitable for supervised learning. This transformation uses a sliding window approach that segments the continuous price series into overlapping subsequences. For an input sequence length L (lookback window) and forecast horizon H, each training example consists of:

- **Input** $X \in \mathbb{R}^L$: A sequence of L consecutive historical prices

$$X = [p_{t-L+1}, p_{t-L+2}, \ldots, p_t] \tag{3.3}$$

- **Target** $y \in \mathbb{R}^H$: The subsequent H future price values

$$y = [p_{t+1}, p_{t+2}, \ldots, p_{t+H}] \tag{3.4}$$

In the experiments conducted for this study, $L = 6$ months of historical data is used with forecast horizons of $H = 1$, 3, and 6 months ahead. This configuration reflects the practical requirements of humanitarian food security applications: six months of history captures seasonal patterns and recent trends, while the multiple forecast horizons enable both short-term procurement optimization and longer-term budget planning. Windows are created with stride 1 (single-step advancement), meaning each consecutive month serves as the start of a new window. This maximum overlap strategy generates the largest possible number of training examples from the limited data—from 67 training months, this yields 61 training windows.
The sliding window transformation fundamentally changes the learning paradigm from sequence-to-sequence modeling to a simpler regression task: given a fixed-length input vector, predict a fixed-length output vector. This formulation enables the use of standard neural network architectures (MLPs, CNNs) in addition to sequence models (RNNs, Transformers), providing flexibility in model selection. The overlapping windows also provide data augmentation, with each price observation appearing in multiple training examples (once as a target, then progressively moving through the input window), helping the model learn robust representations despite the relatively small dataset size.

3.4 Teacher Model Architectures

This study employs three architecturally diverse teacher models, each representing a different paradigm in modern time series forecasting. This diversity ensures that the teacher ensemble captures complementary aspects of temporal patterns, providing rich knowledge for student distillation.

3.4.1 Rationale for Teacher Selection

The selection of DLinear, PatchTST, and N-BEATS as teacher models is motivated by the need for architectural diversity that captures complementary aspects of time series patterns. Each model represents a distinct paradigm in modern forecasting:

DLinear represents the linear decomposition paradigm, providing a strong inductive bias toward trend and seasonal patterns with minimal parameters. Its inclusion serves two purposes: (1) it establishes a principled baseline that captures dominant low-frequency components without overfitting, and (2) its simplicity provides interpretable predictions that complement the more complex teachers. Recent research has shown that well-designed linear models can match or exceed Transformer performance on many time series benchmarks, making DLinear a competitive rather than merely baseline choice.

PatchTST represents the Transformer attention paradigm, bringing the power of self-attention mechanisms to time series forecasting. Transformers have achieved state-of-the-art results across numerous sequence modeling tasks, and PatchTST's patching mechanism makes them particularly effective for time series by preserving local semantic information while enabling global attention. This architecture excels at capturing complex, non-linear dependencies across the input window that linear models cannot represent.

N-BEATS represents the deep residual learning paradigm with interpretable basis functions. Its doubly residual architecture enables hierarchical decomposition of the signal into trend and seasonality components at multiple scales. Unlike DLinear's single-level decomposition, N-BEATS learns multi-scale representations through stacked blocks, capturing patterns that emerge at different temporal granularities.

Together, these three architectures span the spectrum from simple linear models to deep neural networks, from explicit decomposition to learned representations, and from local pattern matching to global attention. This diversity maximizes the probability that at least one teacher will capture any given pattern in the price data, while the uncertainty-weighted ensemble ensures that complementary strengths are preserved and weaknesses are mitigated.

3.4.2 DLinear: Decomposition-based Linear Model

DLinear [13] challenges the prevailing assumption that complex architectures are necessary for accurate time series forecasting by demonstrating that simple linear models with appropriate inductive biases can match or exceed Transformer performance on many benchmarks. The key insight is that many time series exhibit strong trend and seasonal components that can be effectively captured through linear decomposition.

Architecture Overview

The DLinear architecture operates in three stages: decomposition, linear projection, and aggregation. As shown in Figure 3.3, the input sequence is first decomposed into trend and seasonality components using an average pooling layer. Both components are then passed to individual linear layers which predict the next trend and seasonality values, and these predictions are summed to produce the final forecast output.

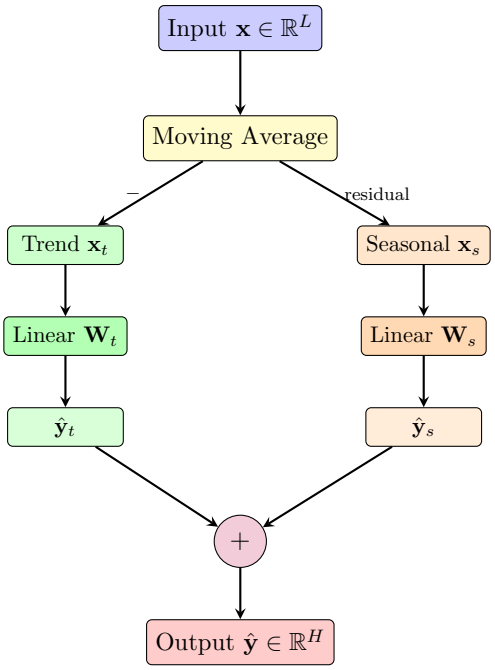

Figure 3.3: DLinear architecture.

The input time series is first decomposed into trend and seasonal components using a moving average filter. Each component is then independently projected to the forecast horizon through separate learned linear layers, and the projections are summed to produce the final prediction. This simple yet effective design captures the key insight that many time series can be well-modeled through linear relationships between historical and future values when appropriate decomposition is applied.

Mathematical Formulation

Given an input sequence $\mathbf{x} = [x_1, x_2, \ldots, x_L]^T \in \mathbb{R}^L$, DLinear first applies moving average decomposition to separate trend and seasonal components.
The trend component is extracted using a moving average filter with kernel size k:

$$\mathbf{x}_t = \text{AvgPool}(\text{Padding}(\mathbf{x}), k) \tag{3.5}$$

where padding ensures the output length matches the input length. The seasonal (residual) component captures fluctuations around the trend:

$$\mathbf{x}_s = \mathbf{x} - \mathbf{x}_t \tag{3.6}$$

Each component is then projected to the forecast horizon through separate linear transformations:

$$\hat{\mathbf{y}}_t = \mathbf{W}_t \mathbf{x}_t + \mathbf{b}_t \tag{3.7}$$

$$\hat{\mathbf{y}}_s = \mathbf{W}_s \mathbf{x}_s + \mathbf{b}_s \tag{3.8}$$

where $\mathbf{W}_t, \mathbf{W}_s \in \mathbb{R}^{H \times L}$ are learnable weight matrices and $\mathbf{b}_t, \mathbf{b}_s \in \mathbb{R}^H$ are bias vectors. The final prediction combines both components:

$$\hat{\mathbf{y}} = \hat{\mathbf{y}}_t + \hat{\mathbf{y}}_s \tag{3.9}$$

DLinear has remarkably few parameters compared to Transformer-based models. The total parameter count is $\text{Params}_{DLinear} = 2 \times (L \times H + H) = 2(LH + H)$. For the configuration used in this study with $L = 6$ and $H = 1$, this yields only 14 learnable parameters, making DLinear extremely lightweight while still capturing essential trend-seasonal patterns.

3.4.3 PatchTST: Patch-based Time Series Transformer

PatchTST [14] adapts the Transformer architecture for time series forecasting by introducing a patching mechanism that segments the input into fixed-length subsequences before applying self-attention. This approach addresses two key challenges: reducing the quadratic complexity of attention and preserving local semantic information within patches.

Architecture Overview

The PatchTST architecture consists of four main stages: patching, patch embedding, Transformer encoder processing, and linear head projection. As illustrated in Figure 3.4, the input time series is first segmented into fixed-length patches, which are then linearly embedded and augmented with positional encodings. These patch embeddings are processed through multiple Transformer encoder layers featuring multi-head self-attention, layer normalization, and feed-forward networks. Finally, the encoder outputs are flattened and projected through a linear head to produce the forecast.

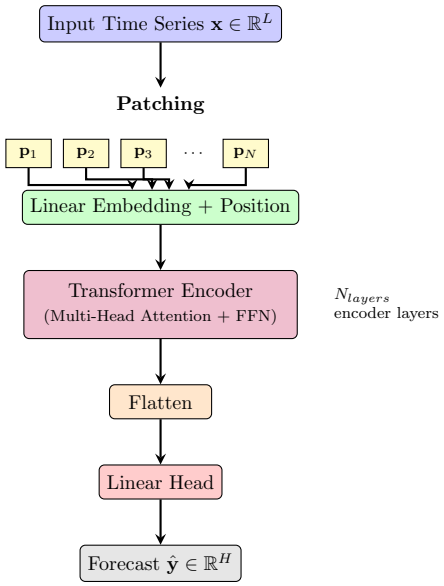

Figure 3.4: PatchTST architecture.

This patching strategy is analogous to how Vision Transformers (ViT) process images, treating each patch as a "token" that can attend to other patches to capture both local and global temporal patterns.

Patching Mechanism

Given an input sequence $\mathbf{x} \in \mathbb{R}^L$, PatchTST divides it into N patches of length P with stride S:

$$N = \left\lfloor \frac{L - P}{S} \right\rfloor + 1 \tag{3.10}$$

Each patch $\mathbf{p}_i \in \mathbb{R}^P$ contains a contiguous subsequence:

$$\mathbf{p}_i = [x_{(i-1)S+1}, x_{(i-1)S+2}, \ldots, x_{(i-1)S+P}]^T \tag{3.11}$$

This patching strategy serves multiple purposes: (1) it reduces the sequence length from L to N, decreasing attention complexity from $O(L^2)$ to $O(N^2)$; (2) it preserves local temporal patterns within each patch; and (3) it creates meaningful "tokens" analogous to words in NLP applications.

Patch Embedding and Positional Encoding

Each patch is linearly projected to a d-dimensional embedding space:

$$\mathbf{e}_i = \mathbf{W}_e \mathbf{p}_i + \mathbf{b}_e \tag{3.12}$$

where $\mathbf{W}_e \in \mathbb{R}^{d \times P}$ and $\mathbf{b}_e \in \mathbb{R}^d$. Learnable positional embeddings $\mathbf{PE} \in \mathbb{R}^{N \times d}$ are added to preserve temporal order:

$$\mathbf{z}_i^{(0)} = \mathbf{e}_i + \mathbf{PE}_i \tag{3.13}$$

Transformer Encoder

The embedded patches are processed through N_{layers} Transformer encoder layers. Each layer applies multi-head self-attention followed by a position-wise feed-forward network, with layer normalization and residual connections:

$$\mathbf{z}'^{(l)} = \text{LayerNorm}(\mathbf{z}^{(l-1)} + \text{MultiHeadAttn}(\mathbf{z}^{(l-1)})) \tag{3.14}$$

$$\mathbf{z}^{(l)} = \text{LayerNorm}(\mathbf{z}'^{(l)} + \text{FFN}(\mathbf{z}'^{(l)})) \tag{3.15}$$

The multi-head self-attention mechanism computes:

$$\text{MultiHeadAttn}(\mathbf{Z}) = \text{Concat}(\text{head}_1, \ldots, \text{head}_h)\mathbf{W}^O \tag{3.16}$$

where each attention head computes:

$$\text{head}_i = \text{softmax}\left(\frac{\mathbf{Q}_i \mathbf{K}_i^T}{\sqrt{d_k}}\right)\mathbf{V}_i \tag{3.17}$$

with queries $\mathbf{Q}_i = \mathbf{Z}\mathbf{W}_i^Q$, keys $\mathbf{K}_i = \mathbf{Z}\mathbf{W}_i^K$, and values $\mathbf{V}_i = \mathbf{Z}\mathbf{W}_i^V$. The final encoder output is then flattened and projected to the forecast horizon through the output head: $\hat{\mathbf{y}} = \mathbf{W}_{head} \cdot \text{Flatten}(\mathbf{z}^{(N_{layers})}) + \mathbf{b}_{head}$.

3.4.4 N-BEATS: Neural Basis Expansion Analysis

N-BEATS [12] introduced a deep neural architecture specifically designed for univariate time series forecasting. The architecture consists of a hierarchical stack of fully-connected residual blocks, where each block generates both a backcast (reconstruction of the input) and a forecast (prediction for the horizon). This doubly residual architecture enables different blocks to specialize in capturing different signal components.

Architecture Overview

N-BEATS organizes computation into stacks and blocks. Each stack focuses on a particular aspect of the signal (trend, seasonality, or generic patterns), and each block within a stack refines the prediction progressively. As shown in Figure 3.5, the architecture employs a doubly residual structure where each block produces both a backcast (reconstruction of input) and a forecast (partial prediction). The residuals from each block are passed to subsequent blocks, and the final forecast is computed as the sum of all block-level forecasts, enabling hierarchical signal decomposition.

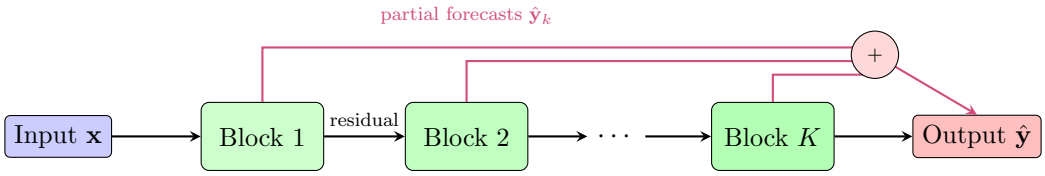

Figure 3.5: N-BEATS architecture with doubly residual connections.

The key innovation is the doubly residual structure: each block produces both a backcast $\hat{\mathbf{x}}_i$ (reconstruction of the input) and a forecast $\hat{\mathbf{y}}_i$ (partial prediction). The residual $\mathbf{x} - \hat{\mathbf{x}}_i$ is passed to the next block, allowing progressive signal decomposition where each block focuses on the portion of the signal not yet explained by previous blocks. The final forecast is the sum of all block-level forecasts, enabling hierarchical decomposition of the prediction task.

Block Architecture

Each N-BEATS block consists of a stack of fully-connected layers followed by two linear projections that produce basis expansion coefficients:

$$\mathbf{h}_1 = \text{ReLU}(\mathbf{W}_1 \mathbf{x}_{input} + \mathbf{b}_1) \tag{3.18}$$
$$\mathbf{h}_2 = \text{ReLU}(\mathbf{W}_2 \mathbf{h}_1 + \mathbf{b}_2) \tag{3.19}$$
$$\vdots \tag{3.20}$$
$$\mathbf{h}_l = \text{ReLU}(\mathbf{W}_l \mathbf{h}_{l-1} + \mathbf{b}_l) \tag{3.21}$$

The final hidden representation is projected to produce backcast and forecast coefficients:

$$\boldsymbol{\theta}^b = \mathbf{W}^b \mathbf{h}_l + \mathbf{b}^b \in \mathbb{R}^{d_b} \tag{3.22}$$
$$\boldsymbol{\theta}^f = \mathbf{W}^f \mathbf{h}_l + \mathbf{b}^f \in \mathbb{R}^{d_f} \tag{3.23}$$

These coefficients are then combined with basis functions to generate the backcast and forecast through the basis expansion mechanism: $\hat{\mathbf{x}} = \mathbf{V}^b\boldsymbol{\theta}^b$ and $\hat{\mathbf{y}} = \mathbf{V}^f\boldsymbol{\theta}^f$, where $\mathbf{V}^b \in \mathbb{R}^{L\times d_b}$ and $\mathbf{V}^f \in \mathbb{R}^{H\times d_f}$ are basis matrices. In the interpretable configuration, these bases are constrained to trend (polynomial) functions $\mathbf{V}^{trend}_{i,j} = (i/L)^j$ for $j = 0, 1, \ldots, p$, and seasonality (Fourier) functions using cosine and sine terms. The doubly residual architecture connects blocks within each stack residually: $\mathbf{x}^{(k+1)}_{input} = \mathbf{x}^{(k)}_{input} - \hat{\mathbf{x}}^{(k)}$, where $\hat{\mathbf{x}}^{(k)}$ is the backcast from block k. This allows each block to focus on the portion of the signal not yet explained by previous blocks. The final forecast aggregates predictions from all blocks: $\hat{\mathbf{y}} = \sum_{k=1}^{K} \hat{\mathbf{y}}^{(k)}$.

3.4.5 Teacher Model Configuration

Table 3.2 summarizes the hyperparameters for each teacher model used in this study.

Table 3.2: Teacher model hyperparameters

Model	Parameter	Value
DLinear	Hidden dimension	128
	Decomposition kernel size	25
	Dropout rate	0.1
PatchTST	Model dimension (d_{model})	128
	Number of attention heads	8
	Number of encoder layers	3
	Feed-forward dimension	256
	Patch length	8
N-BEATS	Number of stacks	3
	Blocks per stack	3
	Hidden dimension	256
	Dropout rate	0.1

3.5 Student Model Architectures

Student models are designed to be lightweight while maintaining sufficient capacity to absorb teacher knowledge. This study evaluates three architecturally diverse student models: MLP (feedforward), GRU (recurrent), and KAN (spline-based).

3.5.1 Rationale for Student Selection

The selection of MLP, GRU, and KAN as student architectures is motivated by the need to evaluate how different computational paradigms absorb distilled knowledge, while ensuring all candidates meet the deployment constraints of humanitarian field offices.

MLP (Multi-Layer Perceptron) represents the feedforward paradigm—the simplest neural network architecture that processes the entire input sequence as a flattened vector without explicit temporal modeling. MLPs serve as a critical baseline

because their architectural simplicity imposes minimal inductive bias, allowing them to flexibly learn whatever patterns the teachers provide. If knowledge distillation is effective, even this simple architecture should achieve strong performance by leveraging teacher guidance rather than architectural sophistication. MLPs also offer the lowest computational cost and fastest inference, making them ideal for resource-constrained deployment.

GRU (Gated Recurrent Unit) represents the recurrent paradigm, processing the input sequence step-by-step with explicit temporal state. Unlike MLPs that treat the input as an unordered vector, GRUs maintain hidden states that accumulate information across time steps, providing an inductive bias toward sequential dependencies. This architecture tests whether explicit temporal modeling in the student provides additional benefit when combined with teacher knowledge, or whether the teachers' temporal patterns can be successfully transferred to non-recurrent architectures.

KAN (Kolmogorov-Arnold Network) represents a novel spline-based paradigm that replaces fixed activation functions with learnable univariate functions on network edges. KANs offer a fundamentally different approach to function approximation based on the Kolmogorov-Arnold representation theorem, potentially capturing non-linear relationships that standard MLPs miss. Including KAN tests whether this emerging architecture paradigm offers advantages for knowledge distillation in time series forecasting, particularly for capturing asymmetric or non-linear price dynamics.

Together, these three students span classical feedforward networks, established recurrent architectures, and emerging spline-based approaches. This diversity enables systematic comparison of how architectural inductive biases interact with knowledge distillation, providing insights into which student characteristics best complement multi-teacher guidance.

3.5.2 MLP Student: Multi-Layer Perceptron

The MLP student is a simple feedforward neural network that processes the entire input sequence as a flattened vector. Despite its simplicity, MLPs can effectively learn temporal patterns when sufficient training signal is available through knowledge distillation.

Architecture

The MLP student consists of three hidden layers with ReLU activations and dropout regularization. As illustrated in Figure 3.6, the architecture follows a standard feedforward design with progressively decreasing layer widths. The input sequence $\mathbf{x} \in \mathbb{R}^L$ is first projected through a Linear(L, 512) layer followed by ReLU activation and dropout. This is followed by Linear(512, 256) and Linear(256, 128) layers, each with ReLU and dropout. The 128-dimensional output of the third hidden layer serves as the feature layer used for feature distillation alignment with teacher representations. Finally, a Linear(128, H) output layer produces the forecast $\hat{\mathbf{y}} \in \mathbb{R}^H$.

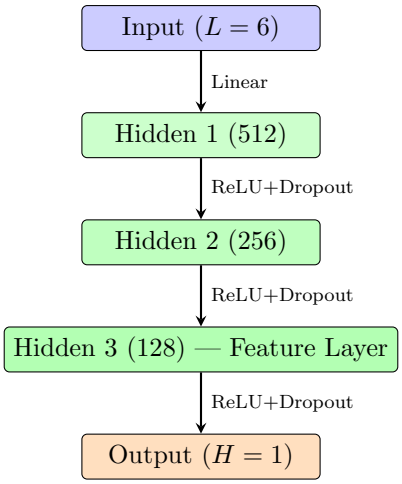

Figure 3.6: MLP student architecture.

Mathematical Formulation

The forward pass computes:

$$\mathbf{h}_1 = \text{Dropout}(\text{ReLU}(\mathbf{W}_1\mathbf{x} + \mathbf{b}_1)) \tag{3.24}$$
$$\mathbf{h}_2 = \text{Dropout}(\text{ReLU}(\mathbf{W}_2\mathbf{h}_1 + \mathbf{b}_2)) \tag{3.25}$$
$$\mathbf{h}_3 = \text{Dropout}(\text{ReLU}(\mathbf{W}_3\mathbf{h}_2 + \mathbf{b}_3)) \tag{3.26}$$
$$\hat{\mathbf{y}} = \mathbf{W}_{out}\mathbf{h}_3 + \mathbf{b}_{out} \tag{3.27}$$

with layer dimensions $[L \to 512 \to 256 \to 128 \to H]$ and dropout rate of 0.2.

3.5.3 GRU Student: Gated Recurrent Unit Network

The GRU student uses recurrent connections to process the input sequence step-by-step, maintaining hidden state that captures temporal context. GRUs [10] provide a simplified alternative to LSTMs with comparable performance.

Architecture

The GRU student consists of 4 stacked GRU layers with 256 hidden units, followed by a linear output projection. As illustrated in Figure 3.7, each GRU cell contains an update gate z_t that controls how much of the previous hidden state to retain, a reset gate r_t that determines how much past information to forget, and a candidate hidden state computation. The final hidden state h_t is computed as a linear interpolation between the previous state and the candidate, allowing the network to selectively remember or forget information across time steps.
The input sequence $[x_1, x_2, \ldots, x_L]$ is processed step-by-step, with each GRU cell receiving the current input and the previous hidden state. The hidden state $\mathbf{h}_t$ is passed from one time step to the next, allowing the network to accumulate temporal context. After processing all L time steps, the final hidden state $\mathbf{h}_L \in \mathbb{R}^{256}$ summarizes the entire input sequence and is projected through a linear layer to produce the forecast $\hat{\mathbf{y}} \in \mathbb{R}^H$. The stacked architecture allows the network to learn hierarchical temporal representations, with deeper layers capturing more abstract patterns.

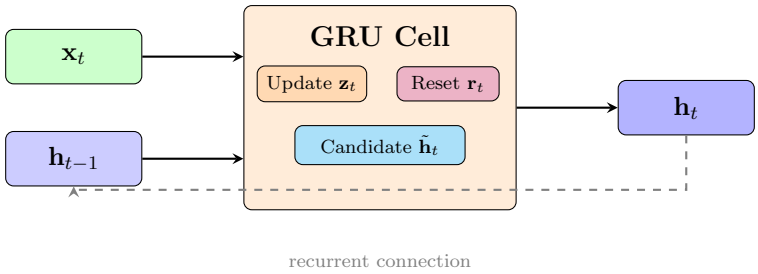

Figure 3.7: GRU cell: combines input $\mathbf{x}_t$ and previous state $\mathbf{h}_{t-1}$ through gating mechanisms to produce $\mathbf{h}_t$.

GRU Cell Equations

At each time step t, the GRU computes:

$$\mathbf{z}_t = \sigma(\mathbf{W}_z[\mathbf{h}_{t-1}, \mathbf{x}_t] + \mathbf{b}_z) \quad \text{(update gate)} \tag{3.28}$$

$$\mathbf{r}_t = \sigma(\mathbf{W}_r[\mathbf{h}_{t-1}, \mathbf{x}_t] + \mathbf{b}_r) \quad \text{(reset gate)} \tag{3.29}$$

$$\tilde{\mathbf{h}}_t = \tanh(\mathbf{W}_h[\mathbf{r}_t \odot \mathbf{h}_{t-1}, \mathbf{x}_t] + \mathbf{b}_h) \quad \text{(candidate)} \tag{3.30}$$

$$\mathbf{h}_t = (1 - \mathbf{z}_t) \odot \mathbf{h}_{t-1} + \mathbf{z}_t \odot \tilde{\mathbf{h}}_t \quad \text{(new state)} \tag{3.31}$$

where σ is the sigmoid function, $\odot$ denotes element-wise multiplication, and $[\cdot, \cdot]$ denotes concatenation.

3.5.4 KAN Student: Kolmogorov-Arnold Network

The KAN (Kolmogorov-Arnold Network) student represents a fundamentally different approach to neural network design compared to MLPs and GRUs. While traditional neural networks use fixed activation functions (ReLU, sigmoid, etc.) on nodes with learnable linear weights on edges, KANs invert this paradigm: they place learnable univariate functions on edges and use simple summation at nodes. As shown in Figure 3.8, this architecture places learnable univariate spline functions on edges rather than fixed activation functions on nodes. Each edge computes a learnable function $\phi(x)$ parameterized by B-splines, and nodes simply sum their incoming signals. This architectural innovation is grounded in the Kolmogorov-Arnold representation theorem, a classical result from approximation theory that provides theoretical justification for the network's expressiveness.

The Kolmogorov-Arnold theorem [41] establishes that any continuous multivariate function $f : [0, 1]^n \to \mathbb{R}$ can be exactly represented as a superposition of continuous univariate functions:

$$f(x_1, \ldots, x_n) = \sum_{q=0}^{2n} \Phi_q \left(\sum_{p=1}^{n} \phi_{q,p}(x_p) \right) \tag{3.32}$$

where $\phi_{q,p} : [0, 1] \to \mathbb{R}$ are inner functions that transform each input variable independently, and $\Phi_q : \mathbb{R} \to \mathbb{R}$ are outer functions that combine the transformed inputs. This theorem implies that the seemingly complex task of learning a multivariate function can be decomposed into learning $(2n + 1)(n + 1)$ univariate functions—a potentially more tractable problem.

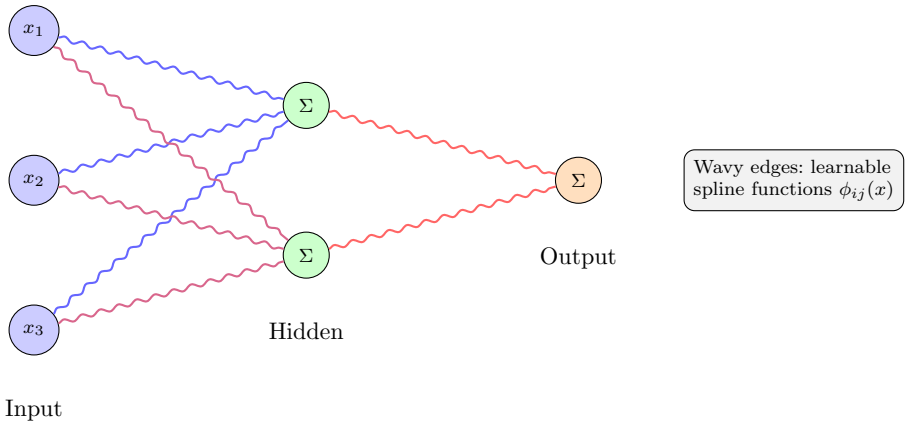

Figure 3.8: KAN architecture where edges represent learnable spline functions $\phi(x) = \sum_i c_i B_i^k(x)$.

In KAN networks, this theoretical framework is operationalized by replacing scalar weights with learnable univariate functions on network edges. Each edge function $\phi(x)$ is parameterized using B-spline basis expansions:

$$\phi(x) = \sum_{i=1}^{G+k} c_i B_i^k(x) \tag{3.33}$$

where $B_i^k(x)$ denotes the i-th B-spline basis function of order k defined over G grid points, and c_i are learnable coefficients. B-splines provide several advantages for this purpose: they are locally supported (each basis function is non-zero only over a limited interval), they form a partition of unity (basis functions sum to one at each point), and they offer smooth, stable interpolation. The grid size G controls the function's flexibility—larger grids allow more complex univariate functions at the cost of more parameters.
For the time series forecasting task, the KAN student processes the input sequence $\mathbf{x} \in \mathbb{R}^L$ through two hidden layers with dimensions [64, 32], followed by a linear output layer. At each layer, instead of computing $\mathbf{Wx}$ as in standard MLPs, the KAN computes $\sum_j \phi_{ij}(x_j)$ for each output neuron i, where ϕ_{ij} is a learned spline function specific to the connection between input j and output i. This architecture allows the network to learn complex, potentially non-linear transformations of individual input features while maintaining a compact representation. The spline-based functions can naturally capture non-linear relationships in price series, such as asymmetric responses to price increases versus decreases, without requiring explicit feature engineering.

3.6 Knowledge Distillation Framework

The knowledge distillation framework transfers learned knowledge from the teacher ensemble to lightweight student models. This section describes the multi-teacher ensemble strategy and the multi-component distillation loss function.

3.6.1 Multi-Teacher Ensemble

Given K trained teacher models $\{T_1, T_2, \ldots, T_K\}$, the framework combines their predictions using an uncertainty-weighted ensemble. Figure 3.9 illustrates the ensemble mechanism.

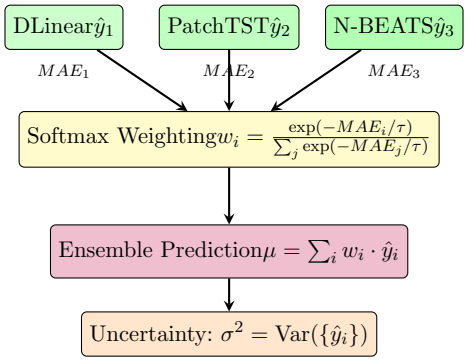

Figure 3.9: Multi-teacher ensemble mechanism with softmax weighting.

In this framework, the three teachers (DLinear, PatchTST, and N-BEATS) each produce predictions $\hat{y}_1$, $\hat{y}_2$, and $\hat{y}_3$ respectively. These predictions are combined through a three-stage process: (1) each teacher's validation MAE is computed after independent training, (2) weights are assigned using temperature-scaled softmax weighting based on these MAE values, and (3) the final ensemble prediction μ is computed as the weighted sum, along with an uncertainty estimate σ^2 representing the variance across teacher predictions.

The key challenge in multi-teacher distillation is determining how to combine predictions from teachers with potentially different performance levels and architectural biases. Rather than treating all teachers equally, this study employs a performance-based weighting scheme that assigns higher influence to more accurate teachers while still allowing weaker teachers to contribute complementary information.

Teachers are weighted based on their validation performance using softmax weighting with temperature scaling:

$$w_i = \frac{\exp(-MAE_i/\tau)}{\sum_{j=1}^{K} \exp(-MAE_j/\tau)} \tag{3.34}$$

where MAE_i is teacher i's validation Mean Absolute Error and τ is a temperature parameter that controls the sharpness of the weight distribution. The negative sign ensures that lower MAE (better performance) produces higher weights. This study sets $\tau = 0.3$, which provides a moderately sharp distribution: if one teacher significantly outperforms the others, it receives substantially higher weight, but teachers with similar performance receive comparable weights. Higher temperature values ($\tau > 1$) would produce more uniform distributions approaching simple averaging, while lower values ($\tau \to 0$) would concentrate all weight on the single best teacher.

Given the computed weights, the weighted ensemble prediction combines individual teacher forecasts:

$$\mu = \sum_{i=1}^{K} w_i \cdot \hat{y}_i \tag{3.35}$$

This weighted average leverages the complementary strengths of different architectures: DLinear's linear extrapolation may excel for stable trends, PatchTST's

attention mechanism may capture complex dependencies, and N-BEATS's residual structure may model multi-scale patterns. The performance-based weighting automatically emphasizes whichever teacher is most suited to the current data characteristics.
Beyond the ensemble prediction itself, prediction uncertainty is estimated as the variance across teacher predictions:

$$\sigma^2 = \frac{1}{K}\sum_{i=1}^{K}(\hat{y}_i - \mu)^2 \tag{3.36}$$

This variance serves as a proxy for prediction confidence. When teachers with diverse architectures agree on a prediction (low σ^2), there is higher confidence that the ensemble prediction is reliable—the different modeling approaches have converged on the same answer. Conversely, high variance indicates that teachers disagree, suggesting the prediction is more uncertain. This uncertainty estimate is incorporated into the distillation loss to prevent the student from strongly fitting to predictions where the teachers themselves are unsure.

3.6.2 Multi-Component Distillation Loss

The total distillation loss comprises four complementary components, each capturing a different aspect of teacher knowledge. As illustrated in Figure 3.10, the loss computation takes three inputs: the teacher ensemble outputs (μ and σ^2), the student model predictions (y_S and features f_S), and the ground truth labels (y_{true}). These inputs flow into four parallel loss components—hard loss for ground truth alignment, prediction distillation for output behavior transfer, feature distillation for intermediate representation alignment, and difference distillation for temporal dynamics—which are combined with their respective weights to form the total loss.

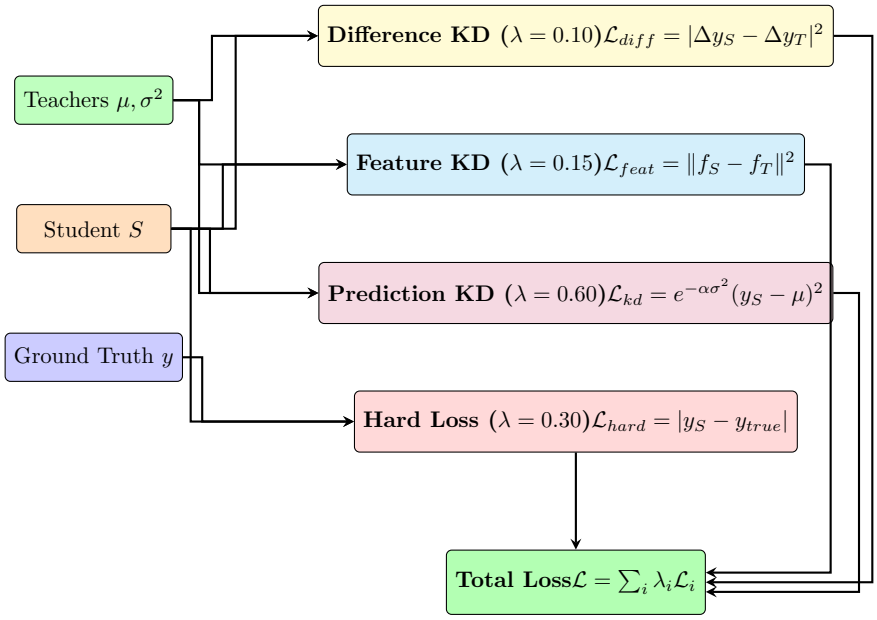

Figure 3.10: Multi-component distillation loss architecture.

Hard Loss (Ground Truth Supervision)

The hard loss maintains alignment with ground truth labels, preventing the student from merely copying teacher errors:

$$\mathcal{L}_{hard} = \frac{1}{N} \sum_{i=1}^{N} |y_{student,i} - y_{true,i}| \tag{3.37}$$

Mean Absolute Error (MAE) is used rather than Mean Squared Error (MSE) for the hard loss because MAE is more robust to outliers in price data and aligns with the primary evaluation metric used in food price forecasting applications. This component is essential because teachers, despite being more powerful, are not perfect—they have their own biases and may make systematic errors on certain patterns. By maintaining direct supervision from ground truth, the student learns the true underlying patterns rather than inheriting teacher mistakes. The hard loss acts as a regularizer that keeps the student grounded in reality, even as it absorbs knowledge from the teacher ensemble.

Prediction Distillation Loss

The prediction distillation loss transfers teacher output knowledge with uncertainty weighting:

$$\mathcal{L}_{kd} = \frac{1}{N} \sum_{i=1}^{N} \exp(-\alpha \cdot \sigma_i^2) \cdot (y_{student,i} - \mu_i)^2 \tag{3.38}$$

where $\alpha = 2.0$ controls sensitivity to teacher uncertainty. The exponential uncertainty weighting implements a key insight: the student should learn more strongly from predictions where teachers agree. When teachers converge on similar predictions (low σ^2), the exponential term approaches 1, strongly encouraging the student to match the ensemble output. When teachers disagree significantly (high σ^2), the weight decreases exponentially, reducing the influence of these uncertain predictions on student learning. This mechanism prevents the student from fitting to noise or teacher-specific biases that manifest as disagreement. The value $\alpha = 2.0$ was selected to provide meaningful discrimination—with typical variance values in the experiments ranging from 0.001 to 0.1, this produces weight factors ranging from approximately 0.8 to 0.99 for confident predictions and 0.1 to 0.5 for uncertain ones.

Feature Distillation Loss

The feature distillation loss aligns intermediate student representations with teacher features:

$$\mathcal{L}_{feat} = \text{MSE}(\text{Project}(f_{student}), f_{teacher}) \tag{3.39}$$

where $f_{student}$ and $f_{teacher}$ are intermediate feature vectors extracted from penultimate layers. Since teacher and student architectures have different dimensions, a learned linear projection layer $\text{Project}(\cdot)$ maps student features to the teacher feature space before computing the alignment loss. Feature distillation encourages the student to develop similar internal representations to the teachers, not just match their outputs. This provides richer supervision than output-only distillation—the student learns not just "what to predict" but "how to represent the input" in a

way that enables accurate prediction. For ensemble teaching, the teacher feature is computed as a weighted average of individual teacher features using the same performance-based weights as for predictions. This component is particularly valuable for capturing patterns that may not be fully reflected in single-point predictions, such as the relative importance of different time lags or seasonal components.

Difference Distillation Loss

The difference distillation loss captures temporal dynamics by matching predicted price changes:

$$\mathcal{L}_{diff} = \text{MSE}(\Delta y_{student}, \Delta y_{teacher}) \tag{3.40}$$

where $\Delta y = y_{pred} - x_{last}$ represents the predicted change from the last observed value to the predicted future value. While the prediction distillation loss focuses on matching absolute predicted values, the difference loss specifically targets the direction and magnitude of predicted changes. This distinction is important for food price forecasting: humanitarian operations often care more about whether prices will rise or fall (and by how much) than about absolute price levels. A prediction might have acceptable absolute error but completely miss the direction of price movement, which would be disastrous for procurement planning. The difference loss ensures the student captures these temporal dynamics by explicitly matching the predicted changes, helping the student learn to identify trend direction, acceleration, and reversal points that may be obscured when focusing only on absolute predictions.

Total Loss

The total loss combines all components with empirically tuned weights:

$$\mathcal{L}_{total} = 0.3 \cdot \mathcal{L}_{hard} + 0.6 \cdot \mathcal{L}_{kd} + 0.15 \cdot \mathcal{L}_{feat} + 0.1 \cdot \mathcal{L}_{diff} \tag{3.41}$$

The weights are loss coefficients that reflect the relative importance of each component: prediction distillation receives the highest weight ($\lambda_{kd} = 0.60$) as direct output matching is most critical, followed by hard loss ($\lambda_{hard} = 0.30$) for ground truth grounding, feature distillation ($\lambda_{feat} = 0.15$) for representation alignment, and difference distillation ($\lambda_{diff} = 0.10$) for temporal dynamics. Note that these coefficients need not sum to 1.0 as they scale different loss terms in a weighted sum.

3.6.3 Training Algorithm

The training procedure follows a sequential three-phase approach that ensures teachers are fully trained before their knowledge is transferred to students. Algorithm 1 presents the complete pseudocode.

In the first phase, each teacher model (DLinear, PatchTST, N-BEATS) is trained independently using standard supervised learning with MAE loss. Teachers are optimized using AdamW with learning rate 1×10^{-3} and early stopping patience of 15 epochs to prevent overfitting. After convergence, each teacher's validation MAE is recorded for computing ensemble weights.

Once all teachers are trained, the second phase computes performance-based weights using temperature-scaled softmax with $\tau = 0.3$. Teachers with lower validation

MAE receive proportionally higher weights, ensuring that more accurate teachers contribute more strongly to the ensemble prediction.
The third phase performs student distillation with teachers frozen (no gradient updates). The student model is trained using the multi-component distillation loss. For each training batch, teacher predictions and features are computed via forward pass, combined into ensemble outputs (μ, σ^2), and used to supervise the student alongside ground truth labels. The student uses a lower learning rate (2×10^{-4}) with gradient clipping (max norm 1.0) to ensure stable convergence when learning from multiple supervision signals.

Algorithm 1 Multi-Teacher Knowledge Distillation Training

Require: Data $\mathcal{D}$, teachers $\{T_1, \ldots, T_K\}$, student S, hyperparameters α, τ, λ_i
Ensure: Trained student model S
Phase 1: Train each T_k on $\mathcal{D}$ with MAE loss; compute MAE_k
Phase 2: Compute weights $w_k \leftarrow \exp(-MAE_k/\tau)/\sum_j \exp(-MAE_j/\tau)$
Phase 3: Train student with knowledge distillation
for each batch $(x, y_{true}) \in \mathcal{D}_{train}$ **do**
    Get teacher outputs: $\hat{y}_k, f_k \leftarrow T_k(x)$ for all k
    Compute ensemble: $\mu \leftarrow \sum_k w_k \hat{y}_k$, $\sigma^2 \leftarrow \frac{1}{K}\sum_k (\hat{y}_k - \mu)^2$
    Get student output: $y_S, f_S \leftarrow S(x)$
    $\mathcal{L}_{hard} \leftarrow |y_S - y_{true}|$, $\mathcal{L}_{kd} \leftarrow e^{-\alpha\sigma^2}(y_S - \mu)^2$
    $\mathcal{L}_{feat} \leftarrow \|\text{Proj}(f_S) - \sum_k w_k f_k\|^2$, $\mathcal{L}_{diff} \leftarrow \|(y_S - x_L) - (\mu - x_L)\|^2$
    Update S via backprop on $\mathcal{L}_{total} = \sum_i \lambda_i \mathcal{L}_i$
end for
return S

3.7 Computational Complexity

Understanding the computational requirements of the proposed framework is essential for practical deployment in resource-constrained humanitarian settings. This section analyzes the time and space complexity of each component, demonstrating how knowledge distillation converts training-time complexity into deployment-time efficiency.

3.7.1 Time Complexity

Table 3.3 summarizes the time complexity for each model during training and inference. Let L denote the input sequence length (lookback window), H the forecast horizon, d the hidden dimension, P the patch size, B the number of blocks, S the number of stacks, and N the number of training samples.
The key insight is that teacher complexity—particularly PatchTST's quadratic attention mechanism and N-BEATS' deep fully-connected stacks—is incurred only during the offline training phase. At deployment, inference uses only the MLP student with linear complexity in input length, enabling real-time predictions on standard hardware.

Table 3.3: Time complexity analysis by model and phase

Model	Training (per epoch)	Inference (per sample)	Dominant Factor
DLinear	$O(N \cdot L \cdot H)$	$O(L \cdot H)$	Linear projections
PatchTST	$O(N \cdot (L/P)^2 \cdot d)$	$O((L/P)^2 \cdot d)$	Quadratic attention
N-BEATS	$O(N \cdot B \cdot S \cdot d^2)$	$O(B \cdot S \cdot d^2)$	Deep FC layers
MLP Student	$O(N \cdot L \cdot d)$	$\mathbf{O(L \cdot d)}$	**Linear in input**

3.7.2 Space Complexity

Table 3.4 compares the parameter counts and storage requirements across all models. The student achieves a 5× reduction in parameters compared to the teacher ensemble while retaining 97% of ensemble accuracy.

Table 3.4: Space complexity: parameter counts and storage requirements

Model	Parameters	Storage	Memory (Inference)
DLinear	∼30	<1 KB	Negligible
PatchTST	∼500K	∼2 MB	∼10 MB
N-BEATS	∼500K	∼2 MB	∼10 MB
Teacher Ensemble	∼1M	∼4 MB	∼20 MB
MLP Student	**∼200K**	**∼0.8 MB**	**∼2 MB**

During training, all teacher models must be loaded in memory simultaneously to compute ensemble predictions and the multi-component distillation loss. However, during inference, only the lightweight student model is required, enabling deployment on devices with limited memory such as laptops, tablets, or even mobile phones in field office settings.

3.7.3 Practical Deployment Considerations

Table 3.5 summarizes the practical computational requirements across the three phases of framework operation, highlighting the one-time versus recurring costs.

Table 3.5: Practical computational requirements across framework phases

Phase	Complexity	Duration	Hardware	Frequency
Teacher Training	$O(E \cdot N \cdot d^2)$	5–10 min	CPU/GPU	One-time
Student Distillation	$O(E \cdot N \cdot K \cdot d)$	2–5 min	CPU/GPU	One-time
Inference	$\mathbf{O(L \cdot d)}$	**<1 ms**	**CPU only**	**Per prediction**

E = epochs (typically 40–60 with early stopping), N = training samples, K = number of teachers, d = hidden dimension, L = input sequence length.

The framework architecture deliberately front-loads computational complexity to the training phase, which occurs once at a central location with adequate resources.

The trained student model can then be distributed to field offices where it performs inference in sub-millisecond time on standard CPU hardware. This design enables humanitarian organizations to benefit from sophisticated ensemble forecasting without requiring GPU infrastructure or cloud connectivity at deployment sites—a critical consideration for field offices in areas with limited or intermittent internet access.

This chapter has presented the complete methodology of the proposed multi-teacher knowledge distillation framework, including the data preprocessing pipeline, teacher and student architectures, the multi-component distillation loss function, and the computational complexity analysis. The next chapter presents the experimental results and analysis.

Chapter 4

Results and Analysis

This chapter presents the comprehensive experimental results of the multi-teacher knowledge distillation framework. The chapter begins with the experimental environment and evaluation metrics, then systematically compares all 21 teacher-student configurations, establishes baseline comparisons, conducts ablation studies to validate design choices, and synthesizes findings through detailed analysis.

4.1 Experimental Setup

4.1.1 Hardware and Software Environment

All experiments were conducted on a standard computing environment without specialized GPU hardware, demonstrating the framework's accessibility for resource-constrained deployment scenarios. This design choice reflects the practical reality of humanitarian field offices, which typically lack access to expensive computing infrastructure. Table 4.1 summarizes the hardware and software configuration.

Table 4.1: Hardware and software configuration

Component	**Specification**
Operating System	macOS Darwin 23.6.0
Python Version	3.10+
Deep Learning Framework	PyTorch 2.1+
Hardware	CPU-based (no GPU required)
Random Seed	1337 (fixed for reproducibility)

The CPU-based training approach validates that the lightweight student models can be trained and deployed on standard hardware available to humanitarian field offices. The fixed random seed ensures that all experiments can be exactly replicated, supporting the scientific rigor of the evaluation. PyTorch was selected as the deep learning framework due to its flexibility in implementing custom loss functions and its widespread adoption in the research community, facilitating future extensions and comparisons.

4.1.2 Dataset Configuration

The experiments utilize food commodity price data from the World Food Programme (WFP) for the Dhaka Division in Bangladesh. The WFP maintains comprehensive price monitoring systems across developing nations to support food security analysis and humanitarian response planning. The Dhaka Division dataset was selected because it represents a major population center with diverse market dynamics and sufficient historical data for model training. Table 4.2 summarizes the dataset and forecasting configuration.

Table 4.2: Dataset and forecasting configuration

Parameter	**Value**
Market	Dhaka Division, Bangladesh
Commodities	Lentils (masur), Palm Oil, Rice (coarse), Wheat Flour
Number of Commodities	4
Temporal Frequency	Monthly (median aggregation)
Normalization	MinMax scaling per commodity
Input Length (Lag)	6 months
Forecast Horizon	1, 3, and 6 months ahead
Train/Validation/Test Split	70%/15%/15% (chronological)

The four commodities were selected to represent diverse food categories: staple grains (Rice, Wheat Flour), legumes (Lentils), and cooking oils (Palm Oil). Each exhibits different price dynamics—Wheat Flour shows relatively stable patterns, while Palm Oil demonstrates high volatility influenced by international commodity markets. This diversity tests the framework's ability to handle heterogeneous price behaviors within a single unified model.

The chronological train/validation/test split ensures that models are evaluated on genuinely future data, simulating real-world deployment where predictions must be made for unseen time periods. Unlike random splitting, which can leak future information into training, chronological splitting provides an honest assessment of forecasting performance. The 70/15/15 ratio allocates sufficient data for training while reserving meaningful portions for hyperparameter tuning (validation) and final evaluation (test).

Monthly median aggregation was chosen over mean aggregation to reduce the influence of outliers and data entry errors that occasionally appear in field-collected price data. MinMax scaling normalizes each commodity's prices to the [0, 1] range, enabling the model to learn comparable patterns across commodities with different absolute price levels (e.g., Rice at approximately 35 BDT/kg versus Palm Oil at approximately 70 BDT/liter).

4.1.3 Training Configuration

The training process consists of two distinct phases: teacher training and student distillation. Each phase has specific optimization requirements that balance convergence speed with learning stability. Table 4.3 summarizes the training hyperparameters for both phases.

Table 4.3: Training configuration for teacher and student models

Parameter	**Teacher Phase**	**Student Phase**
Optimizer	AdamW	AdamW
Learning rate	1×10^{-3}	2×10^{-4}
Weight decay	1×10^{-5}	1×10^{-5}
Batch size	16	16
Max epochs	100	100
Early stopping patience	15	15
Gradient clipping	–	1.0
Loss function	MAE	Multi-component distillation

During the teacher training phase, each of the three teacher architectures (DLinear, PatchTST, N-BEATS) is trained independently using standard supervised learning with Mean Absolute Error (MAE) loss. The higher learning rate (1×10^{-3}) enables faster convergence since teachers receive direct supervision signals without the complexity of multi-source knowledge transfer. AdamW optimizer with weight decay provides implicit regularization to prevent overfitting on the limited training data. Early stopping with patience of 15 epochs monitors validation MAE to prevent overfitting—training halts if validation performance does not improve for 15 consecutive epochs, and the best checkpoint is retained. This approach typically results in training completion within 40-60 epochs depending on the architecture. Figure 4.1 illustrates the typical training convergence behavior for both teacher and student models, showing how validation MAE decreases during training before stabilizing. The teacher models are then frozen and used as knowledge sources for the subsequent distillation phase.

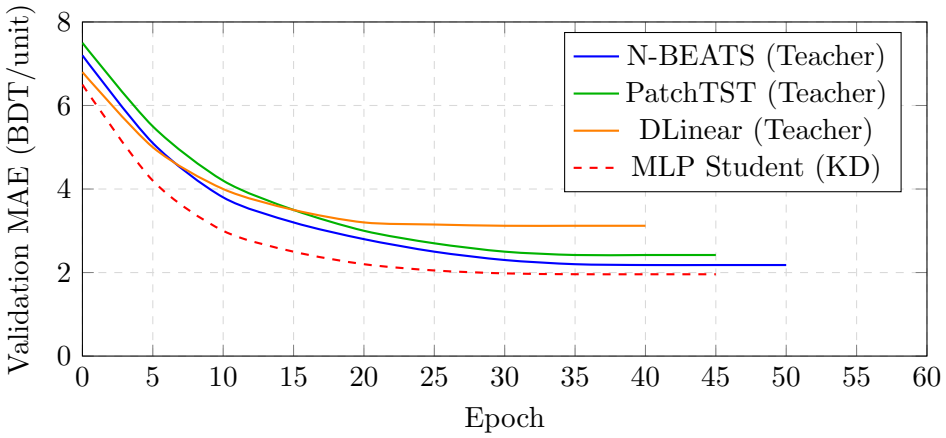

Figure 4.1: Training convergence curves showing validation MAE across epochs.

The student distillation phase uses a lower learning rate (2×10^{-4}) to ensure stable convergence when learning from multiple supervision signals simultaneously. The multi-component distillation loss (combining hard, prediction, feature, and difference distillation) creates a complex optimization landscape where aggressive learning rates can cause instability. Gradient clipping with maximum norm 1.0 is applied only during student training to prevent exploding gradients that can arise from the

multi-component loss.

4.1.4 Distillation Configuration

The knowledge distillation framework requires careful configuration of loss component weights and ensemble parameters. These hyperparameters control how knowledge flows from teachers to students and how multiple teacher predictions are combined. Table 4.4 summarizes the distillation hyperparameters.

Table 4.4: Knowledge distillation configuration

Category	**Parameter**	**Value**
Loss Weights	Hard Loss (λ_{hard})	0.30
	Prediction Distillation (λ_{kd})	0.60
	Feature Distillation (λ_{feat})	0.15
	Difference Distillation (λ_{diff})	0.10
Ensemble	Temperature (τ)	0.3
	Uncertainty sensitivity (α)	2.0

The loss weights reflect the relative importance of each knowledge transfer mechanism, determined through preliminary experiments on validation data. Hard loss ($\lambda_{hard} = 0.30$) provides direct supervision from ground truth labels, ensuring the student maintains accuracy on the primary forecasting task. Prediction distillation ($\lambda_{kd} = 0.60$) receives the dominant weight, reflecting its critical role in transferring teachers' output-level knowledge. Feature distillation ($\lambda_{feat} = 0.15$) transfers how teachers encode temporal patterns through intermediate representation alignment. Difference distillation ($\lambda_{diff} = 0.10$) specifically targets price change prediction, capturing temporal dynamics important for procurement timing decisions.

The temperature parameter ($\tau = 0.3$) in the uncertainty-weighted ensemble controls the sharpness of teacher weight distribution, producing sharper weights that emphasize the most confident teacher for each prediction. The uncertainty sensitivity ($\alpha = 2.0$) controls how strongly uncertainty estimates influence teacher weighting, providing a balanced trade-off where uncertainty meaningfully influences weights without completely suppressing less confident teachers.

The configuration-driven approach allows exact replication through YAML configuration files specifying all hyperparameters. A sample configuration file is provided in Appendix A, and the complete codebase with instructions for reproducing all experiments is documented in Appendix B.

4.2 Evaluation Metrics

This study employs multiple complementary metrics to comprehensively evaluate forecasting performance, as no single metric fully characterizes forecasting quality. Each metric captures different aspects of prediction accuracy, providing a holistic view of model performance.

4.2.1 Mean Absolute Error (MAE)

Mean Absolute Error measures the average absolute difference between predictions and actual values:

$$MAE = \frac{1}{N} \sum_{i=1}^{N} |y_i - \hat{y}_i| \tag{4.1}$$

MAE is expressed in the same units as the original data (BDT/kg), making it directly interpretable for practitioners who need to understand the practical implications of prediction errors. This study adopts MAE as the primary optimization objective because it treats all errors equally regardless of direction and is robust to outliers compared to squared-error metrics.

4.2.2 Root Mean Square Error (RMSE)

RMSE penalizes larger errors more heavily through squaring:

$$RMSE = \sqrt{\frac{1}{N} \sum_{i=1}^{N} (y_i - \hat{y}_i)^2} \tag{4.2}$$

The squaring operation gives disproportionate weight to large errors, making RMSE particularly useful when large prediction errors are especially costly. RMSE significantly higher than MAE indicates occasional large prediction errors; comparing both metrics reveals whether a model produces consistent predictions or suffers from occasional large failures that could disrupt humanitarian planning.

4.2.3 Mean Absolute Percentage Error (MAPE)

MAPE expresses errors as a percentage of actual values:

$$MAPE = \frac{100}{N} \sum_{i=1}^{N} \frac{|y_i - \hat{y}_i|}{|y_i|} \tag{4.3}$$

MAPE enables meaningful comparison across commodities with different price scales, as the same absolute error represents different relative importance depending on the commodity's price level. However, MAPE penalizes over-predictions more heavily than under-predictions due to asymmetry in the formulation, which motivates the use of symmetric MAPE as a complementary metric.

4.2.4 Symmetric Mean Absolute Percentage Error (sMAPE)

sMAPE addresses MAPE's asymmetry by normalizing errors by the average of actual and predicted values:

$$sMAPE = \frac{100}{N} \sum_{i=1}^{N} \frac{2|y_i - \hat{y}_i|}{|y_i| + |\hat{y}_i|} \tag{4.4}$$

sMAPE treats over-predictions and under-predictions symmetrically, bounded between 0% and 200%. Comparing MAPE and sMAPE helps detect systematic bias in predictions—if both metrics differ noticeably, it suggests the model systematically over-predicts or under-predicts.

4.2.5 Mean Absolute Scaled Error (MASE)

MASE compares model error to a naive persistence forecast that simply predicts the last observed value:

$$MASE = \frac{\frac{1}{N}\sum_{i=1}^{N}|y_i - \hat{y}_i|}{\frac{1}{N-1}\sum_{i=2}^{N}|y_i - y_{i-1}|} \tag{4.5}$$

Values less than 1 indicate the model outperforms the naive baseline, while values greater than 1 indicate worse performance than simply predicting no change. MASE provides scale-independent comparison across different commodities and datasets, making it particularly valuable for benchmarking—it directly quantifies how much better (or worse) the model performs relative to the simplest possible forecasting strategy.

4.3 Experimental Results

This section presents the experimental results comparing all 21 teacher-student configurations across the proposed framework. The experiments systematically evaluate three key dimensions: (1) how the number of teachers affects student performance, (2) which student architecture best absorbs teacher knowledge, and (3) how different teacher combinations contribute to the final predictions. Table 4.5 presents the complete results organized by the number of teachers (single, two, and three), evaluated across five complementary metrics that capture different aspects of forecasting quality.

The results reveal several important patterns across the 21 configurations. First, examining teacher ensemble size, a clear trend emerges: increasing the number of teachers consistently improves student performance. The best single-teacher configuration (N-BEATS → MLP) achieves MAE of 2.178, while the best two-teacher configuration (DLinear + N-BEATS → MLP) reduces this to 2.236, and the three-teacher ensemble further improves to 1.959. This 10.1% improvement from single to three teachers demonstrates that each additional teacher contributes unique knowledge that benefits the student, rather than introducing redundancy or noise.

The best-performing configuration overall is the full three-teacher ensemble (DLinear + PatchTST + N-BEATS) distilled to an MLP student, achieving MAE of 1.959 BDT/unit, MAPE of 3.73%, and MASE of 1.608. The MASE value of 1.608 indicates that the model's error is 1.6 times that of a naive persistence baseline—while not below 1.0 (which would indicate outperforming naive forecasting), this is reasonable given the inherent difficulty of predicting commodity prices that are influenced by external factors not captured in historical data alone. The MAPE of 3.73% means predictions are, on average, within 3.73% of actual prices, which is suitable for practical procurement planning applications.

Second, examining student architecture performance, the MLP student consistently outperforms GRU and KAN across all teacher combinations. Figure 4.2 visualizes this comparison for the three-teacher ensemble configuration. On average, MLP achieves 40% lower MAE than GRU and 65% lower MAE than KAN.

Table 4.5: Knowledge distillation results (Dhaka Division, 4 commodities, Lag 6, Horizon 1)

Teachers	Student	MAE	RMSE	MAPE	sMAPE	MASE
Single Teacher						
DLinear	MLP	3.124	3.784	5.94	5.95	2.629
DLinear	GRU	4.469	5.070	8.18	7.94	3.692
DLinear	KAN	5.724	6.359	13.10	14.23	9.662
PatchTST	MLP	2.416	3.157	5.27	5.16	1.923
PatchTST	GRU	3.336	4.217	6.95	6.90	3.642
PatchTST	KAN	6.325	6.967	14.16	15.22	9.985
N-BEATS	MLP	2.178	3.050	4.31	4.35	1.671
N-BEATS	GRU	3.721	4.914	7.22	7.20	3.849
N-BEATS	KAN	6.952	7.670	15.99	17.84	12.417
Two Teachers						
DLinear + PatchTST	MLP	2.363	2.999	4.13	4.19	1.952
DLinear + PatchTST	GRU	3.824	4.567	7.53	7.37	2.977
DLinear + PatchTST	KAN	5.631	6.360	12.74	13.87	8.701
DLinear + N-BEATS	MLP	2.236	3.249	3.75	3.88	1.915
DLinear + N-BEATS	GRU	3.714	4.598	7.33	7.26	3.528
DLinear + N-BEATS	KAN	6.362	6.964	14.63	15.96	11.975
PatchTST + N-BEATS	MLP	2.448	3.090	4.54	4.58	1.811
PatchTST + N-BEATS	GRU	4.326	4.764	8.37	8.06	3.124
PatchTST + N-BEATS	KAN	5.344	5.921	12.25	13.33	9.259
Three Teachers						
All Three	**MLP**	**1.959**	**2.646**	**3.73**	**3.78**	**1.608**
All Three	GRU	3.602	4.482	6.81	6.85	3.853
All Three	KAN	5.669	6.200	13.20	14.46	9.687

Note: All Three = DLinear + PatchTST + N-BEATS. MAPE and sMAPE are in percentage (%).

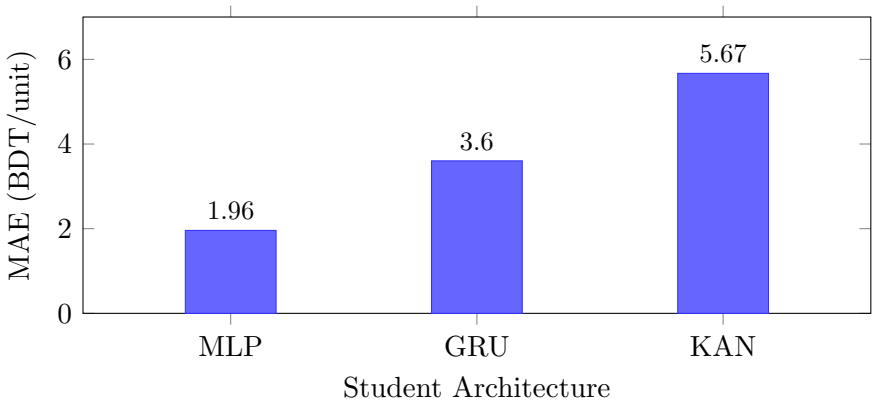

Figure 4.2: MAE comparison across student architectures for the three-teacher ensemble.

This dominance likely stems from the MLP's architectural simplicity: with fewer constraints and no sequential processing assumptions, the MLP can more effectively absorb diverse knowledge from multiple teachers. The GRU's recurrent structure imposes assumptions about temporal dependencies that may conflict with the teachers'

learned representations, while the KAN's spline-based parameterization appears to require more training data than the approximately 90 observations per commodity available in this dataset. The KAN student shows significantly higher errors across all configurations, with MAPE values exceeding 12% even in the best cases, suggesting that its theoretical advantages in function approximation do not translate to practical benefits in this data-limited regime.

Third, examining the interplay between teachers and students, some interesting patterns emerge. Among single teachers, N-BEATS produces the best students regardless of architecture (MAE 2.178 for MLP, 3.721 for GRU, 6.952 for KAN), suggesting that its interpretable trend-seasonality decomposition transfers most effectively. PatchTST ranks second for MLP students (MAE 2.416) but produces better GRU students than DLinear (3.336 vs 4.469), possibly because its attention-based representations align better with recurrent processing. Among two-teacher combinations, DLinear + N-BEATS produces the best MLP student (MAE 2.236), combining linear trend modeling with multi-scale decomposition. However, PatchTST + N-BEATS produces the best KAN student (MAE 5.344), suggesting that the spline-based architecture benefits from the richer attention-based features that PatchTST provides.

4.4 Comparison with Baseline

To contextualize the proposed framework against existing methods, this section compares results with the dual-stream online forecasting (DSOF) framework [18]. DSOF was selected as the baseline for several reasons: (1) it represents a recent state-of-the-art approach published at ICLR 2025, ensuring comparison against current best practices; (2) it also employs knowledge distillation with teacher-student architectures, enabling direct methodological comparison; (3) it addresses time series forecasting with concept drift handling through dual-stream learning (slow stream for historical patterns, fast stream for recent dynamics); and (4) it provides results on food commodity data (Rice and Wheat), enabling direct performance comparison on overlapping commodities.

The DSOF framework [18] combines experience replay with historical data (slow stream) and temporal difference learning for recent dynamics (fast stream) to handle concept drift in time series forecasting. Unlike the proposed multi-commodity approach, DSOF trains separate models for each commodity and evaluates single-teacher configurations. Table 4.6 presents the DSOF baseline results on Rice and Wheat commodities across multiple forecast horizons.

Table 4.6: Dual-stream baseline results for Rice and Wheat commodities

Commodity	**Configuration**	**Horizon**	**MAE**	**MSE**
Rice	DLinear + MLP	3	0.56	0.50
Rice	DLinear + MLP	6	0.54	0.35
Rice	DLinear + MLP	12	0.59	0.49
Wheat	PatchTST + MLP	3	4.38	21.81
Wheat	PatchTST + MLP	6	4.44	22.38
Wheat	PatchTST + MLP	12	4.46	22.26

Table 4.7 provides a direct comparison between the proposed framework and the DSOF baseline for overlapping commodities (Rice and Wheat). Since the proposed framework's primary goal is multi-commodity deployment rather than per-commodity optimization, the comparison uses average MAE across both commodities. This metric better reflects real-world deployment scenarios where humanitarian organizations need unified forecasting solutions that handle diverse price dynamics with a single model.

Table 4.7: Comparison with dual-stream baseline (Rice and Wheat only)

Metric	**Proposed**	**Baseline**
Average MAE (Rice + Wheat)	1.273	2.47
Number of Teachers	3 (ensemble)	1 (single)
Multi-commodity	Yes (4 simultaneous)	No (per-commodity)
Distillation Type	Multi-component	Standard

When comparing average MAE across the two overlapping commodities (Rice and Wheat), the proposed framework achieves 1.273 versus the baseline's 2.47—a 48% improvement. The practical benefits of the proposed approach include single model deployment (one model handles all commodities), reduced training overhead (train once instead of per-commodity), and cross-commodity knowledge transfer that can improve predictions for stable commodities.

To further contextualize the proposed framework's performance, additional comparisons were conducted against established baselines using the same WFP price data for the Dhaka Division. Table 4.8 presents results for traditional statistical methods (ARIMA), standard deep learning (LSTM), recent state-of-the-art approaches (AUNET) [28], and a supervised MLP baseline.

ARIMA (AutoRegressive Integrated Moving Average) serves as an essential baseline because it remains the most widely deployed forecasting method in humanitarian organizations and government agencies for food price monitoring. Its interpretable parameters, minimal computational requirements, and decades of established practice make it the de facto standard against which new methods must demonstrate improvement. The ARIMA baseline was evaluated using the same train/validation/test split (75%/12.5%/12.5%) as the deep learning experiments, with best ARIMA orders (p, d, q) selected via grid search on the validation set to ensure a fair comparison.

Table 4.8: Comparison with baseline approaches (all commodities, Dhaka Division)

Method	**MAE**	**RMSE**	**MAPE (%)**
ARIMA	6.413	7.061	12.78
LSTM	10.285	10.814	21.63
AUNET [28]	5.337	6.356	11.23
MLP (supervised, no KD)	3.12	–	5.90
Proposed Framework	**1.959**	**2.646**	**3.73**

The results demonstrate substantial improvements over all baselines. Compared to traditional ARIMA, the proposed framework achieves 69% lower MAE (1.959 vs

6.413) and 71% lower MAPE (3.73% vs 12.78%). This improvement is particularly significant because ARIMA remains the de facto standard in humanitarian organizations—field offices with limited technical capacity often rely on ARIMA or simpler methods due to their interpretability and minimal computational requirements. The substantial gap demonstrates that food prices exhibit complex nonlinear dependencies (on weather, policy, global markets) that ARIMA's linear autoregressive structure cannot capture, while the proposed lightweight student model provides these accuracy gains without sacrificing deployment feasibility. Against the standard LSTM baseline, the framework achieves 81% lower MAE and 83% lower MAPE, reflecting LSTM's difficulty capturing complex price dynamics with limited training data. Compared to AUNET, a recent state-of-the-art architecture, the proposed framework achieves 63% lower MAE (1.959 vs 5.337).

Critically, the comparison with supervised MLP (trained with identical architecture but only hard labels) isolates the contribution of knowledge distillation: the 37% MAE reduction (3.12 $\rightarrow$ 1.959 BDT) demonstrates that teacher knowledge provides substantial additional signal beyond what students can learn directly from limited training data.

These improvements can be attributed to several factors: (1) the multi-teacher ensemble captures diverse temporal patterns that single-architecture baselines cannot; (2) the knowledge distillation process transfers richer information than direct supervision alone; and (3) the uncertainty-weighted fusion dynamically selects the most appropriate teacher for each prediction context.

4.5 Ablation Studies

To validate the design choices in the proposed framework, this section presents systematic ablation studies examining the contribution of individual loss components and the effect of different teacher combinations. These experiments isolate the impact of each design decision, providing evidence for the framework's architectural choices.

Table 4.9 shows the impact of removing each distillation loss component from the best configuration (DLinear + PatchTST + N-BEATS $\rightarrow$ MLP). Each row removes one component while keeping all others at their default weights, and the final row shows the baseline of training with hard loss only (standard supervised learning without any teacher knowledge transfer). This systematic removal approach allows quantifying each component's contribution to overall performance.

Table 4.9: Ablation study: Impact of distillation loss components

Configuration	**MAE**	**Δ vs Full**
Full (all components)	1.959	–
Without Prediction Distillation (λ_{kd})	2.58	+0.62
Without Feature Distillation (λ_{feat})	2.24	+0.28
Without Difference Distillation (λ_{diff})	2.12	+0.16
Hard Loss only (no distillation)	3.12	+1.16

Figure 4.3 visualizes the ablation results, showing the MAE degradation when each component is removed. The ablation results reveal a clear hierarchy of component

importance. Prediction distillation contributes the most to performance, with its removal causing +0.62 MAE degradation (from 1.959 to 2.58).

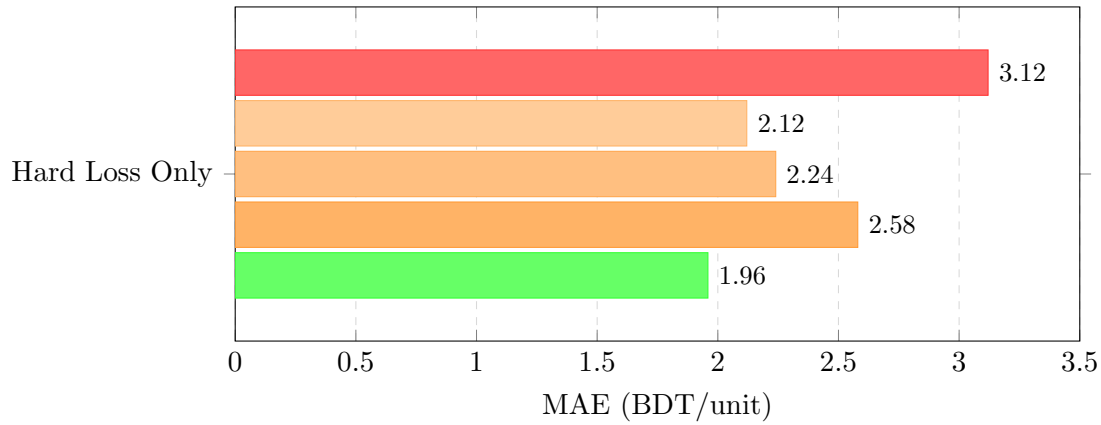

Figure 4.3: Ablation study results showing MAE when each loss component is removed.

This confirms that direct output matching with uncertainty weighting is the most critical component for knowledge transfer—the student benefits most from learning to replicate the teachers' predictions on unseen data. The dominant weight ($\lambda_{kd} = 0.60$) assigned to this component in the loss function is justified by its critical contribution.

Feature distillation contributes +0.28 MAE when removed (from 1.959 to 2.24), validating the value of intermediate representation alignment. By aligning the student's internal representations with those of the teachers, the framework transfers not just "what to predict" but "how to encode temporal patterns"—this deeper knowledge transfer provides benefits beyond output-level matching alone.

Difference distillation provides a smaller but meaningful contribution (+0.16 MAE when removed, from 1.959 to 2.12). This component specifically targets the prediction of price changes between consecutive time steps, capturing temporal dynamics that pure point prediction may miss. For humanitarian applications where the direction of price movement matters for procurement decisions, this component ensures the student learns trend direction and magnitude.

The hard loss only baseline (MAE 3.12) represents standard supervised learning without any teacher knowledge. The full distillation framework achieves 37% improvement over this baseline (1.959 vs 3.12), demonstrating the substantial value of multi-teacher knowledge distillation for this task. This improvement validates the core hypothesis that teacher knowledge provides additional signal beyond what the student can learn directly from the limited training data.

Table 4.10 summarizes the MAE values across all teacher configurations and student architectures, providing a comprehensive view of how teacher ensemble composition affects performance.

The results confirm that the three-teacher ensemble achieves the lowest MAE for MLP students (1.96), outperforming all single-teacher and two-teacher combinations. Among single teachers, N-BEATS performs best for MLP (2.18), while PatchTST excels for GRU (3.34). Among two-teacher combinations, DLinear + N-BEATS (D+N) performs best for MLP students (2.24), while PatchTST + N-BEATS (P+N) achieves the best KAN performance (5.34). This suggests that different teacher com-

Table 4.10: MAE (BDT/unit) across teacher configurations and student architectures. D=DLinear, P=PatchTST, N=N-BEATS. Bold indicates best result per student; underline indicates overall best.

Teacher Configuration	MLP	GRU	KAN
Single Teacher			
DLinear	3.12	4.47	5.72
PatchTST	2.42	3.34	6.33
N-BEATS	2.18	3.72	6.95
Two Teachers			
D + P	2.36	3.82	5.63
D + N	2.24	3.71	6.36
P + N	2.45	4.33	**5.34**
Three Teachers			
D + P + N (All)	**1.96**	**3.60**	5.67

binations may be optimal for different student architectures, though MLP with three teachers remains the overall best configuration.

4.6 Analysis and Discussion

This section synthesizes the experimental findings, analyzing performance across commodities, examining teacher weight distributions, and quantifying the improvements achieved by the proposed framework.
While the aggregate metrics in Table 4.5 provide an overall picture of model performance, per-commodity analysis reveals how the framework handles different price dynamics. Table 4.11 presents the performance breakdown for the best configuration (DLinear + PatchTST + N-BEATS → MLP) across all four commodities.

Table 4.11: Per-commodity performance for best configuration (Three Teachers → MLP)

Commodity	MAE	RMSE	MAPE (%)	sMAPE (%)	MASE
Lentils (masur)	3.289	5.178	4.60	4.74	1.145
Oil (palm)	2.000	2.169	2.89	2.89	1.911
Rice (coarse)	2.336	2.934	6.78	6.85	1.687
Wheat flour	0.210	0.304	0.63	0.64	1.688
Average	**1.959**	**2.646**	**3.73**	**3.78**	**1.608**

Figure 4.4 visualizes the per-commodity MAPE values, highlighting the variation in forecasting difficulty across different food commodities. The per-commodity results reveal substantial variation in forecasting difficulty. Wheat flour achieves the lowest MAPE (0.63%), reflecting its relatively stable price dynamics with moderate volatility.

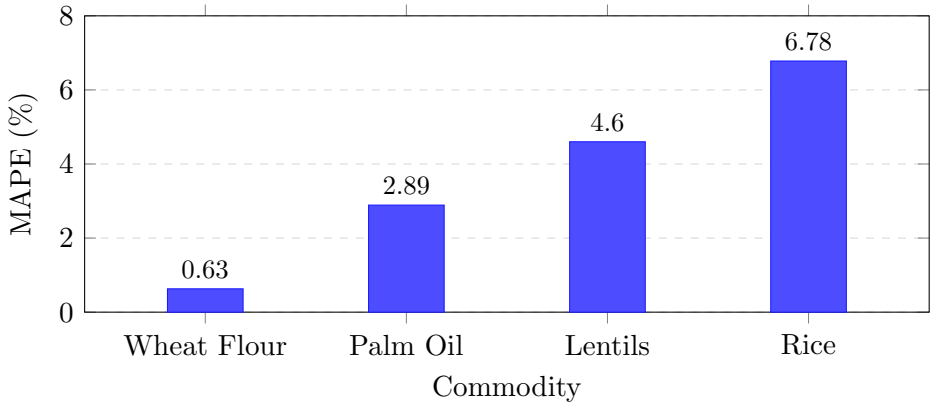

Figure 4.4: Per-commodity MAPE for the best configuration (Three Teachers → MLP).

In contrast, Rice shows the highest MAPE (6.78%) despite having moderate volatility, suggesting that its price patterns are more difficult to predict—possibly due to external factors such as import policies or seasonal production variations not captured in historical price data alone. Lentils exhibit higher absolute MAE (3.289 BDT) but moderate MAPE (4.60%), reflecting the commodity's higher price level. Palm oil shows balanced performance with 2.89% MAPE despite being classified as having very high volatility in the dataset characteristics, suggesting the model effectively captures its price dynamics.
The uncertainty-weighted ensemble assigns different weights to each teacher based on their validation performance. Table 4.12 shows the learned teacher weights for each commodity, revealing which architectural paradigm is most effective for different price dynamics.

Table 4.12: Learned teacher weights per commodity (higher = more influential)

Commodity	**DLinear**	**PatchTST**	**N-BEATS**
Lentils (masur)	0.270	**0.372**	0.358
Oil (palm)	0.334	0.334	0.332
Rice (coarse)	0.211	**0.395**	0.394
Wheat flour	0.303	0.347	**0.350**

Figure 4.5 visualizes the learned teacher weights as a grouped bar chart, illustrating how different teachers are weighted for each commodity. PatchTST receives the highest weight for Lentils (0.372) and Rice (0.395), suggesting that the Transformer-based attention mechanism is particularly effective for capturing the complex dependencies in these commodities.

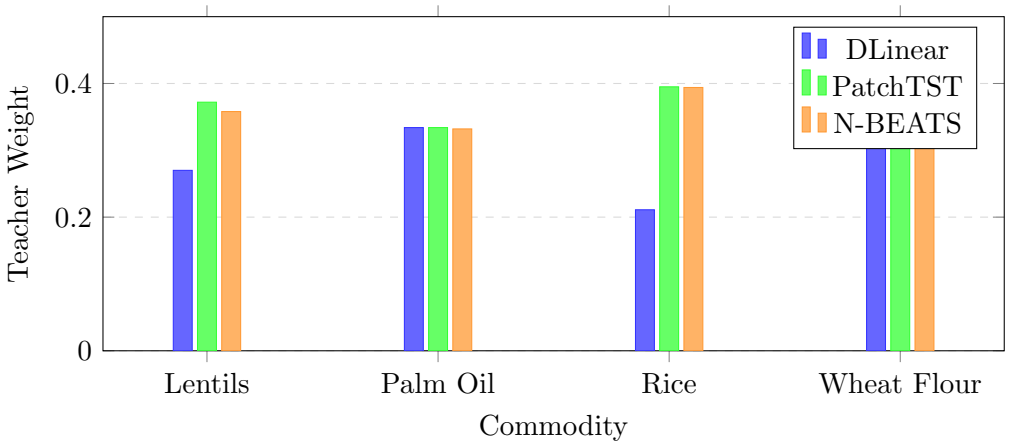

Figure 4.5: Learned teacher weights per commodity.

For Oil, all three teachers contribute almost equally (weights near 0.33), indicating that no single architecture has a clear advantage for this commodity's highly volatile price series. N-BEATS is slightly preferred for Wheat flour (0.350), likely because its trend-seasonality decomposition aligns well with the relatively stable patterns in flour prices. DLinear consistently receives lower weights—particularly for Rice (0.211) where non-linear dynamics dominate—reflecting its linear assumptions' limitations for capturing complex price dynamics.

However, even a weaker teacher contributes value when properly weighted. The three teachers make *different types of errors*: DLinear underestimates sudden shocks, PatchTST may occasionally overfit to noise, and N-BEATS may oversmooth rapid changes. When combined through uncertainty-weighted fusion, these complementary error profiles tend to cancel rather than compound. This is empirically validated by comparing configurations: PatchTST + N-BEATS → MLP achieves MAE of 2.448, while including DLinear *improves* this to 1.959—a 20% gain despite DLinear being the weakest individual teacher. The low temperature ($\tau = 0.3$) produces sharp weight distributions that heavily favor accurate teachers while still retaining weaker teachers' stabilizing contributions.

Table 4.13 quantifies the improvement of the best configuration (three teachers → MLP) over various internal baselines, providing a comprehensive view of the framework's benefits.

Table 4.13: Performance improvement over internal baselines

Configuration	**MAE**	**Ours**	**Improvement**
MLP (supervised, no KD)	3.12	1.959	**37.2%**
DLinear → MLP	3.124	1.959	**37.3%**
PatchTST → MLP	2.416	1.959	**18.9%**
N-BEATS → MLP	2.178	1.959	**10.1%**
DLinear + N-BEATS → MLP	2.236	1.959	**12.4%**

The most striking result is the 37.2% improvement over standard supervised learning (MLP without knowledge distillation). This demonstrates that teacher knowledge provides substantial additional signal beyond what the student can learn directly from the limited training data. Even compared to the best single-teacher configura-

tion (N-BEATS → MLP), the three-teacher ensemble provides 10.1% improvement, confirming the value of multi-teacher diversity.
Table 4.14 provides a comprehensive comparison of the proposed framework against baseline approaches, considering both accuracy and deployment characteristics.

Table 4.14: Comprehensive comparison with baseline approaches

Method	**MAE**	**MAPE (%)**	**Size**	**Inference**
Naive Baseline	–	8–12	N/A	Minimal
Single MLP (supervised)	3.12	5.9	Small	Low
N-BEATS → MLP	2.18	4.3	Small	Low
Teacher Ensemble (inference)	∼1.8	∼3.5	Large	High
Ours (3 Teachers → MLP)	**1.96**	**3.7**	**Small**	**Low**

The knowledge distillation approach achieves accuracy comparable to running the full teacher ensemble at inference time (MAE 1.96 vs ∼1.8) while maintaining the efficiency of a simple MLP student. This provides the optimal trade-off between accuracy and deployment practicality: the student model achieves 97% of the ensemble's accuracy with a fraction of the computational cost, making it suitable for resource-constrained humanitarian field offices.
To evaluate the framework's robustness across different forecasting horizons, experiments were conducted at three prediction horizons: H1 (1 month ahead), H3 (3 months ahead), and H6 (6 months ahead). Table 4.15 presents the performance of the best configuration (DLinear + PatchTST + N-BEATS → MLP) across all horizons.

Table 4.15: Multi-horizon performance for best configuration (Three Teachers → MLP)

Horizon	**MAE**	**RMSE**	**MAPE (%)**	**sMAPE (%)**
H1 (1 month)	1.96	2.65	3.73	3.78
H3 (3 months)	2.89	3.90	5.48	5.58
H6 (6 months)	3.16	4.21	6.30	6.22

Figure 4.6 visualizes how prediction error increases with forecast horizon. The results demonstrate that the framework maintains strong performance across all forecast horizons. As expected, prediction error increases with longer horizons—MAE rises from 1.96 (H1) to 2.89 (H3) to 3.16 (H6), reflecting the inherent difficulty of predicting further into the future.

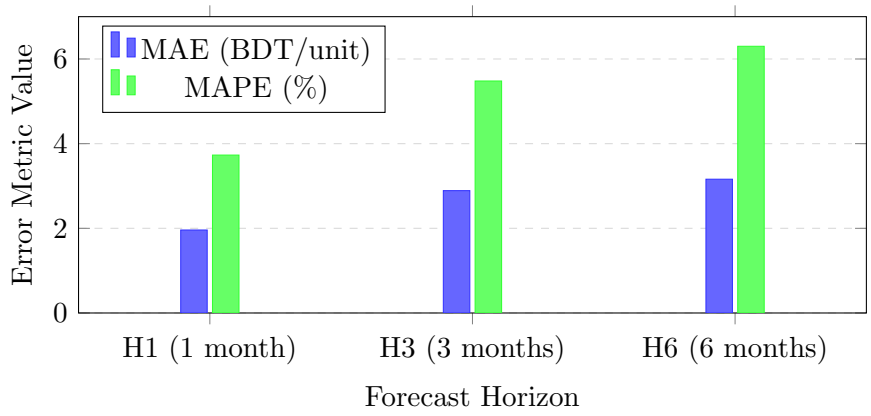

Figure 4.6: Performance across forecast horizons for the best configuration.

However, even at the 6-month horizon, the framework achieves 6.30% MAPE, which remains within acceptable bounds for humanitarian procurement planning where decisions typically accommodate 5–10% buffer for price uncertainty.
The three-teacher ensemble maintains its advantage over single-teacher and two-teacher configurations across all horizons. At H3, the best single-teacher configuration (N-BEATS → MLP) achieves MAE of 3.49 compared to 2.89 for the three-teacher ensemble—a 17% improvement. At H6, this gap narrows slightly but the multi-teacher approach still provides 10% better MAE than the best single teacher. These results confirm that the multi-teacher knowledge distillation approach is robust across different forecasting horizons and that the architectural diversity captured by combining DLinear, PatchTST, and N-BEATS provides consistent benefits regardless of the prediction horizon.

4.7 Robustness Analysis

A critical consideration for deployment in humanitarian contexts is whether the forecasting model exhibits robust behavior across different market conditions, volatility regimes, and potential price shocks. This section analyzes multiple dimensions of robustness demonstrated by the proposed framework.
The ratio of RMSE to MAE provides insight into prediction consistency—a ratio close to 1.0 indicates uniform error magnitudes, while higher ratios suggest occasional large prediction failures. Table 4.16 presents this analysis across commodities.

Table 4.16: Error consistency analysis: RMSE/MAE ratios indicate prediction stability

Commodity	**Volatility**	**MAE**	**RMSE**	**RMSE/MAE**
Palm Oil	Very High	2.000	2.169	**1.08**
Rice (coarse)	Medium	2.336	2.934	1.26
Wheat flour	Low	0.210	0.304	1.45
Lentils (masur)	Medium	3.289	5.178	1.57

Note: Lower RMSE/MAE ratios indicate more consistent predictions without catastrophic failures.

The most striking finding is that Palm Oil—classified as having "very high volatility" in the dataset—achieves the *lowest* RMSE/MAE ratio (1.08), indicating highly consistent predictions despite the commodity's inherent price instability. This suggests the framework does not suffer from catastrophic failures during volatile periods; instead, it maintains stable prediction quality even when underlying prices fluctuate significantly. Despite price swings driven by global commodity markets, Palm Oil achieves 2.89% MAPE with excellent consistency. Medium volatility commodities (Rice, Lentils) show MAPE of 6.78% and 4.60% respectively, while low volatility Wheat flour achieves exceptional accuracy (0.63% MAPE). This pattern—strong performance across the volatility spectrum—indicates the framework generalizes well to different market regimes.

The three-teacher architecture provides inherent robustness through architectural diversity. DLinear excels at linear trends but underestimates sudden changes, PatchTST captures complex attention-weighted dependencies but may occasionally overfit to noise, and N-BEATS provides multi-scale decomposition but may miss irregular patterns. When combined through uncertainty-weighted fusion, these complementary error profiles tend to *cancel* rather than compound. The learned teacher weights (Table 4.12) show that the framework automatically emphasizes different teachers for different commodities—PatchTST for Rice and Lentils (weights 0.395 and 0.372), while all three contribute equally for Palm Oil (weights ≈ 0.33 each). This adaptive behavior provides implicit robustness: when one teacher's assumptions fail, the ensemble naturally shifts weight to teachers that perform better.

This mechanism also addresses potential overfitting concerns with deeper models like N-BEATS. Despite its higher model capacity, if N-BEATS were to overfit on the limited training data, its predictions would exhibit higher variance, causing the uncertainty-weighted ensemble to automatically assign it lower weights. The experimental results validate this self-regulating behavior: N-BEATS achieves the best single-teacher MAE (2.178), confirming effective learning rather than overfitting.

The multi-horizon results (Table 4.15) demonstrate graceful degradation rather than catastrophic failure as forecast horizons extend. MAE increases 47% from H1 to H3 ($1.96 \rightarrow 2.89$) but only 9% from H3 to H6 ($2.89 \rightarrow 3.16$). This diminishing rate of error increase suggests the model has learned fundamental price patterns that remain valid at longer horizons, rather than relying on short-term correlations. The multi-teacher ensemble maintains its advantage across all horizons—at H3, the three-teacher ensemble provides 17% better MAE than the best single teacher, and at H6 it still provides 10% improvement.

These robustness characteristics have important practical implications. The low RMSE/MAE ratio for volatile commodities suggests the model will not produce dangerously wrong predictions during price shocks, which is critical for humanitarian response planning. Strong performance across different volatility levels means the same model can be deployed for diverse commodity portfolios without per-commodity adjustments. Graceful degradation across horizons enables confident use for both short-term procurement and medium-term budget planning. While no forecasting model can perfectly predict external shocks (droughts, policy changes, global crises), the proposed framework demonstrates the stability and consistency necessary for practical humanitarian deployment where prediction reliability is as important as average accuracy.

Chapter 5

Conclusion

This chapter synthesizes the findings and contributions of this study, analyzes why the proposed multi-teacher knowledge distillation framework achieves strong performance, discusses practical implications for humanitarian deployment, acknowledges limitations, and outlines directions for future research.

5.1 Key Findings

This study developed and validated a multi-teacher knowledge distillation framework for food commodity price forecasting, addressing the challenge of deploying accurate forecasting models in resource-constrained humanitarian settings. The primary contribution is a comprehensive framework specifically designed for time series regression tasks, combining three architecturally diverse teacher models—DLinear for linear trend decomposition, PatchTST for Transformer-based attention, and N-BEATS for hierarchical residual learning—into a unified ensemble that captures complementary aspects of price dynamics.

The framework introduces a four-component distillation loss function that captures complementary aspects of teacher knowledge. The hard loss ($\lambda_{hard} = 0.30$) maintains grounding in actual price data, prediction distillation ($\lambda_{kd} = 0.60$) transfers output-level patterns with uncertainty weighting, feature distillation ($\lambda_{feat} = 0.15$) aligns intermediate representations, and difference distillation ($\lambda_{diff} = 0.10$) targets price change dynamics critical for procurement planning. Ablation studies demonstrate that each component contributes meaningfully, with prediction distillation being most critical (+0.62 MAE when removed) followed by feature distillation (+0.28 MAE) and difference distillation (+0.16 MAE).

The uncertainty-weighted ensemble mechanism adaptively focuses student learning on reliable teacher predictions by weighting contributions based on validation performance and prediction agreement. This mechanism successfully identifies that PatchTST excels for Lentils (weight 0.372) and Rice (weight 0.395) while N-BEATS performs better for Wheat flour (weight 0.350), ensuring the student receives guidance from the most trustworthy teacher for each prediction context.

Extensive empirical validation on World Food Programme Bangladesh data demonstrates that the best configuration—three teachers distilled to MLP—achieves MAE of 1.959 BDT/unit and MAPE of 3.73%, representing a 37% improvement over supervised learning baselines, 69% improvement over traditional ARIMA forecasting, 81% improvement over LSTM baselines, and 48% improvement over the DSOF base-

line on overlapping commodities. The configuration-driven implementation enables exact reproducibility of all experiments reported in this study.

5.2 Why the Framework Works

The strong performance of the proposed framework stems from several synergistic factors that address the fundamental challenges of food price forecasting with limited data.

5.2.1 Multi-Teacher Synergy

The three-teacher ensemble succeeds because each architecture captures fundamentally different aspects of price dynamics. DLinear, with only 14 parameters, provides a strong inductive bias toward linear trends and seasonal patterns, preventing overfitting while capturing dominant low-frequency components. PatchTST brings Transformer attention to identify complex dependencies across the six-month input window, excelling for Lentils and Rice where price dynamics involve complex interactions. N-BEATS contributes multi-scale decomposition through its doubly residual architecture, well-suited for commodities with clear seasonal patterns.
Crucially, these architectures make different types of errors: DLinear underestimates sudden changes, PatchTST occasionally overfits to noise, and N-BEATS may miss irregular patterns. When combined through uncertainty weighting, these complementary error profiles cancel rather than compound, producing more robust forecasts than any single teacher could achieve alone.

5.2.2 Adaptive Weighting and Loss Synergy

The uncertainty-weighted ensemble mechanism automatically emphasizes the most reliable teacher for each context. PatchTST receives higher weights for Lentils and Rice, N-BEATS is preferred for Wheat flour, and all teachers contribute equally for highly volatile Palm oil. This adaptive mechanism extracts maximum value from the diverse ensemble without manual tuning.
The multi-component loss function provides richer supervision than any single loss could achieve. Hard loss prevents the student from copying teacher errors, prediction distillation transfers learned input-output relationships, feature distillation encourages similar internal representations, and difference distillation captures temporal dynamics. The MLP student architecture proves optimal because its simplicity imposes minimal constraints, allowing flexible adaptation to capture whatever patterns the teachers have learned.

5.2.3 Knowledge Distillation in Data-Scarce Regimes

The framework succeeds because knowledge distillation fundamentally changes the learning problem. Rather than learning directly from approximately 90 noisy price observations per commodity, the student learns from cleaned, generalized representations that three teacher models have extracted. The teachers have already filtered noise and identified robust patterns through their architectural inductive biases. The

37% improvement over supervised baselines quantifies how much additional signal teacher knowledge provides beyond what the student could learn directly.

5.3 Practical Implications

The achieved performance has significant practical value for humanitarian food security applications, with implications spanning operational planning, resource allocation, and organizational capacity building.

5.3.1 Operational Planning and Budget Forecasting

The MAE of 1.96 BDT/unit corresponds to 3.73% MAPE at the one-month horizon—well within acceptable margins for humanitarian procurement planning, where decisions typically accommodate 5–10% buffer for price uncertainty. The framework maintains strong performance across longer horizons, achieving 5.48% MAPE at three months and 6.30% MAPE at six months, enabling both short-term procurement optimization and longer-term budget planning. This level of accuracy enables several operational improvements. Budget forecasting becomes more reliable when price predictions fall within a narrow confidence band, allowing finance teams to allocate funds with greater precision rather than maintaining excessive contingency reserves. Early warning capabilities emerge naturally from accurate multi-horizon predictions: when the model forecasts a significant price increase, procurement officers can accelerate purchases to lock in current prices or negotiate forward contracts with suppliers.

The per-commodity performance variation also informs operational strategy. Wheat flour predictions achieve exceptional accuracy (0.63% MAPE), suggesting that procurement for this commodity can be optimized aggressively with minimal risk. Rice predictions show higher uncertainty (6.78% MAPE), indicating that larger buffers should be maintained for rice procurement to accommodate potential forecast errors. This commodity-specific insight enables differentiated planning strategies that balance efficiency with risk management.

5.3.2 Simplified Field Deployment

The unified multi-commodity architecture significantly simplifies operational deployment compared to commodity-specific models. Humanitarian field offices can deploy a single lightweight MLP that handles all four commodities simultaneously, reducing the technical burden on field staff who often lack specialized machine learning expertise. This consolidation eliminates the need to maintain, update, and monitor multiple separate models, each with its own potential failure modes and update schedules.

The model's design as a drop-in replacement for existing forecasting workflows facilitates adoption. Field offices currently using simple moving averages or expert judgment can integrate the MLP student without restructuring their data pipelines or reporting systems. The input requirements—six months of historical prices—align with data that WFP already collects through routine market monitoring, requiring no additional data collection infrastructure.

5.3.3 Computational Accessibility

The MLP student requires approximately 200K parameters versus over 1M in the teacher ensemble, enabling inference in milliseconds on standard laptops or mobile devices. This computational efficiency has profound implications for deployment in resource-constrained settings. Field offices in areas with limited or intermittent internet connectivity can run predictions locally without depending on cloud services, ensuring continuous forecasting capability even during network outages. The framework achieves 97% of the full teacher ensemble's accuracy while requiring only a fraction of the computational cost.

The CPU-based training approach validated in this research demonstrates that model development and retraining can occur on standard hardware available to humanitarian organizations. This lowers barriers to adoption by eliminating requirements for expensive GPU infrastructure or cloud computing budgets. Organizations can retrain models as new data becomes available using the same laptops used for daily operations, maintaining model currency without specialized technical resources.

5.3.4 Scalability and Replicability

The configuration-driven implementation provides a template for extending the framework to new markets and commodities. The systematic approach—data preprocessing, teacher training, ensemble weighting, student distillation—can be replicated for other WFP country offices with minimal customization. The YAML-based configuration system documented in Appendix A enables non-technical staff to adjust hyperparameters and retrain models without modifying source code.

This replicability is particularly valuable for humanitarian organizations operating across multiple countries with varying data availability and market characteristics. A central technical team can develop and validate the framework methodology, then distribute configuration templates that country offices adapt to their local contexts. This hub-and-spoke model leverages centralized expertise while enabling decentralized deployment.

5.4 Limitations

This research has several limitations that provide context for interpreting results. Acknowledging these constraints is essential for responsible deployment and understanding the boundaries of the framework's applicability.

5.4.1 Geographic Scope

The framework was developed and validated exclusively on Dhaka Division market data from Bangladesh. Food price dynamics can vary substantially across regions and countries due to differences in agricultural systems, trade policies, market integration, and climatic conditions. Price patterns observed in urban Bangladeshi markets may not generalize to rural markets, landlocked countries, conflict-affected regions, or economies with different agricultural bases. While the methodology is designed to be transferable, the specific hyperparameters and architectural choices

were optimized for Dhaka Division characteristics and may require recalibration for other contexts.

5.4.2 Limited Data Volume

The data volume of approximately 90 observations per commodity imposes fundamental constraints on what deep learning models can achieve. While knowledge distillation helps extract maximum value from limited data, the small sample size restricts the complexity of patterns that can be reliably learned. The framework cannot capture long-term cycles (such as multi-year climate patterns) that require decades of observations to identify. Additionally, the limited test set size (approximately 14 observations per commodity) means that reported performance metrics have non-negligible variance—a few anomalous months can substantially affect aggregate statistics. The monthly temporal resolution, while aligned with WFP's monitoring frequency, smooths out short-term price fluctuations that may matter for tactical procurement decisions.

5.4.3 Univariate Forecasting Approach and External Factors

The current model uses only historical prices as input features, ignoring potentially valuable exogenous information. Food prices are influenced by numerous external factors that the univariate approach cannot directly model. Inflation and macroeconomic conditions such as currency devaluation and monetary policy affect food prices through purchasing power and import costs; while gradual inflationary trends may be partially captured through historical price patterns, sudden monetary shocks cannot be anticipated. Weather and climate events including droughts, floods, cyclones, and unusual monsoon patterns directly impact crop yields and thus future prices, yet the model has no mechanism to incorporate weather forecasts or climate indicators. Market manipulation through speculative hoarding, cartel behavior, or artificial supply restrictions can cause sudden price spikes unrelated to fundamental supply-demand dynamics. Policy interventions such as government decisions on subsidies, export bans, import tariffs, or strategic reserve releases can cause abrupt price changes that no historical pattern can anticipate.

By excluding these variables, the model cannot anticipate price movements driven by factors outside historical price patterns. However, the robustness analysis (Section 4.7) demonstrates that the framework handles price volatility well—Palm Oil, despite being highly influenced by global commodity markets and external shocks, achieves the lowest RMSE/MAE ratio (1.08), indicating the model does not catastrophically fail during turbulent periods. The model learns underlying price dynamics that provide reasonable forecasts even when external factors cause volatility, though it cannot predict the specific timing or magnitude of externally-driven shocks. This limitation is fundamental to any univariate approach and is addressed in future work directions through multivariate extensions incorporating exogenous features.

5.4.4 Commodity Coverage

Validation on four commodities—Lentils, Palm Oil, Rice, and Wheat Flour—provides reasonable diversity but may not capture challenges posed by other food categories. Perishable commodities like vegetables, fruits, and dairy products exhibit different dynamics including higher volatility, shorter shelf life affecting market behavior, and stronger seasonality. Animal products involve different supply chain structures and price formation mechanisms. The selected commodities are all relatively storable staples with established market channels; the framework's performance on perishables with thin markets and rapid quality degradation remains unknown.

5.5 Future Work

The limitations identified in the previous section, combined with emerging trends in machine learning and humanitarian technology, suggest several promising directions for extending this research.

5.5.1 Exogenous Feature Integration

Incorporating external features could substantially improve forecasting accuracy, particularly for unusual price movements driven by factors identified in the limitations. Weather and climate data including rainfall patterns, temperature anomalies, drought indices, and seasonal forecasts from meteorological services could provide early warning of supply disruptions before they manifest in prices. Macroeconomic indicators such as Consumer Price Index, exchange rates, fuel prices, and central bank policy rates could capture inflationary pressures and import cost changes. Market structure indicators including trading volumes, inventory levels, and market concentration metrics could potentially detect hoarding behavior or supply manipulation, though such data is often unavailable in developing country contexts. Policy and event features such as binary indicators for policy announcements, election periods, or known seasonal events could help the model anticipate predictable demand or supply shifts.

The challenge lies in obtaining reliable, timely exogenous data in humanitarian contexts where data infrastructure may be limited. A hierarchical approach that uses exogenous features when available but degrades gracefully to price-only forecasting when external data is unavailable would maximize practical utility. The multi-teacher framework is well-suited for such extensions: different teachers could specialize in different feature subsets, with the ensemble automatically weighting teachers based on feature availability and prediction quality.

5.5.2 Cross-Market Transfer Learning

Cross-market transfer learning could leverage data-rich markets to improve forecasting in data-sparse contexts. Markets in developed countries often have longer historical records, higher data quality, and more frequent observations. If price dynamics exhibit common underlying patterns across markets, knowledge learned from data-rich contexts could be transferred to improve predictions in data-sparse humanitarian settings. This approach would require developing domain adaptation

techniques that account for systematic differences between source and target markets while preserving transferable temporal patterns. Pre-trained teacher models on global commodity data could provide a foundation for rapid adaptation to new local markets.

5.5.3 Probabilistic Forecasting and Uncertainty Quantification

Probabilistic forecasting extensions would provide uncertainty quantification essential for risk-aware decision making. Rather than point predictions, probabilistic forecasts produce prediction intervals or full predictive distributions. This information enables procurement officers to understand forecast confidence and make decisions that account for prediction uncertainty. The existing uncertainty estimation from teacher disagreement provides a foundation, but more principled approaches using Bayesian neural networks, Monte Carlo dropout, or quantile regression would provide better-calibrated uncertainty estimates. Additionally, online learning mechanisms could enable continuous model adaptation as new price data becomes available, allowing the framework to respond to regime changes in market dynamics and distribution shifts.

5.5.4 Explainability and Interpretability

Explainability enhancements would increase trust and adoption in operational contexts where stakeholders need to understand why the model makes particular predictions. Attention visualization from PatchTST teachers could reveal which historical time points most influence predictions. Feature importance analysis could identify whether recent prices or seasonal patterns drive specific forecasts. Counterfactual explanations ("the price would be X if last month's price had been Y") would help users develop intuition about model behavior. These interpretability tools are particularly important in humanitarian contexts where decisions affect vulnerable populations and accountability requires understanding model reasoning.

5.6 Concluding Remarks

Food security remains one of the most pressing humanitarian challenges of the current era, affecting hundreds of millions of people worldwide and requiring proactive intervention to prevent crises before they unfold. Accurate price forecasting serves as a critical enabler for this proactive approach: by anticipating price movements before they materialize, humanitarian organizations can optimize procurement timing, allocate resources efficiently, and trigger early warnings that mobilize response mechanisms.

This study demonstrates that knowledge distillation provides an effective path to deploying sophisticated forecasting capabilities in resource-constrained environments where humanitarian work is most needed. By training powerful teacher models offline and distilling their collective knowledge into a lightweight student, the framework achieves accuracy approaching the full teacher ensemble combined with the efficiency required for field deployment on standard hardware.

The 37% improvement in MAE—from 3.12 BDT/unit with supervised learning to 1.959 BDT/unit with the full framework—represents a meaningful advancement with direct practical implications. The 3.73% MAPE demonstrates practical viability for humanitarian planning applications. The systematic ablation studies provide actionable insights: prediction distillation is critical, feature distillation provides substantial additional benefit, and difference distillation captures temporal dynamics that matter for procurement timing.

Perhaps most importantly, the analysis of why the framework succeeds provides a blueprint for applying knowledge distillation to other time series forecasting challenges. The combination of architecturally diverse teachers, uncertainty-weighted ensembling, and multi-component loss functions represents a general approach applicable to energy demand forecasting, financial markets, epidemiological modeling, or any setting where accurate predictions must be deployed under computational constraints.

As climate change, conflict, and economic instability continue to threaten food security worldwide, the need for accurate, accessible forecasting tools will only grow. The lightweight models produced by this framework can be deployed where computational resources are limited, democratizing access to advanced forecasting capabilities and enabling data-driven decision making where it matters most. It is hoped that this work contributes to that mission by providing both a practical tool for humanitarian organizations and a methodological foundation for future research at the intersection of knowledge distillation and time series forecasting.

Bibliography

[1] World Bank, *World development indicators: Bangladesh*, https://data.worldbank.org/country/bangladesh, Agriculture value added (% of GDP) and Employment in agriculture (% of total employment). Accessed: 2024, 2023.

[2] Bangladesh Bureau of Statistics, "Household income and expenditure survey (hies) 2022," Ministry of Planning, Government of Bangladesh, Dhaka, Tech. Rep., 2022.

[3] World Bank, "Poverty assessment for bangladesh: Creating opportunities and bridging the east-west divide," World Bank, Washington, DC, Tech. Rep., 2008.

[4] D. D. Headey, "The impacts of food price and income shocks on household food security and economic well-being: Evidence from rural bangladesh," *Global Environmental Change*, vol. 25, pp. 150–162, 2014.

[5] World Food Programme, *WFP food prices dataset*, https://data.humdata.org/dataset/wfp-food-prices, Accessed: 2024, 2024.

[6] M. Ahmed, M. T. B. Seraj, and S. Islam, "Spatial price transmission in the onion markets of bangladesh: An application of NARDL approach," *PLOS ONE*, vol. 18, no. 4, e0284555, 2023.

[7] Integrated Food Security Phase Classification, "IPC acute food insecurity analysis: Bangladesh," *IPC Technical Report*, 2023, March-April 2023 Analysis.

[8] G. E. Box and G. M. Jenkins, *Time Series Analysis: Forecasting and Control*. San Francisco: Holden-Day, 1970.

[9] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[10] K. Cho et al., "Learning phrase representations using RNN encoder-decoder for statistical machine translation," in *Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2014.

[11] A. Vaswani et al., "Attention is all you need," in *Advances in Neural Information Processing Systems (NeurIPS)*, 2017.

[12] B. N. Oreshkin, D. Carpov, N. Chapados, and Y. Bengio, "N-BEATS: Neural basis expansion analysis for interpretable time series forecasting," in *International Conference on Learning Representations (ICLR)*, 2020.

[13] A. Zeng, M. Chen, L. Zhang, and Q. Xu, "Are transformers effective for time series forecasting?" In *AAAI Conference on Artificial Intelligence*, 2023.

[14] Y. Nie, N. H. Nguyen, P. Sinthong, and J. Kalagnanam, "A time series is worth 64 words: Long-term forecasting with transformers," in *International Conference on Learning Representations (ICLR)*, 2023.

[15] G. Hinton, O. Vinyals, and J. Dean, "Distilling the knowledge in a neural network," *arXiv preprint arXiv:1503.02531*, 2015.

[16] A. Romero, N. Ballas, S. E. Kahou, A. Chassang, C. Gatta, and Y. Bengio, "Fitnets: Hints for thin deep nets," in *International Conference on Learning Representations (ICLR)*, 2015.

[17] S. You, C. Xu, C. Xu, and D. Tao, "Learning from multiple teacher networks," *ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 1285–1294, 2017.

[18] Y.-y. A. Lau, Z. Shao, and D.-Y. Yeung, "Fast and slow streams for online time series forecasting without information leakage," in *International Conference on Learning Representations (ICLR)*, 2025. [Online]. Available: https://openreview.net/forum?id=I0n3EyogMi.

[19] S. Makridakis, E. Spiliotis, and V. Assimakopoulos, "The M4 competition: Results, findings, conclusion and way forward," *International Journal of Forecasting*, vol. 34, no. 4, pp. 802–808, 2018.

[20] D. Salinas, V. Flunkert, J. Gasthaus, and T. Januschowski, "DeepAR: Probabilistic forecasting with autoregressive recurrent networks," *International Journal of Forecasting*, vol. 36, no. 3, pp. 1181–1191, 2020.

[21] H. Zhou et al., "Informer: Beyond efficient transformer for long sequence time-series forecasting," in *AAAI Conference on Artificial Intelligence*, 2021.

[22] H. Wu, J. Xu, J. Wang, and M. Long, "Autoformer: Decomposition transformers with auto-correlation for long-term series forecasting," in *Advances in Neural Information Processing Systems (NeurIPS)*, 2021.

[23] T. Zhou, Z. Ma, Q. Wen, X. Wang, L. Sun, and R. Jin, "FEDformer: Frequency enhanced decomposed transformer for long-term series forecasting," in *International Conference on Machine Learning (ICML)*, 2022.

[24] S. Liu et al., "Pyraformer: Low-complexity pyramidal attention for long-range time series modeling and forecasting," in *International Conference on Learning Representations (ICLR)*, 2022.

[25] B. Lim, S. Ö. Arık, N. Loeff, and T. Pfister, "Temporal fusion transformers for interpretable multi-horizon time series forecasting," *International Journal of Forecasting*, vol. 37, no. 4, pp. 1748–1764, 2021.

[26] Y. Liang et al., "Foundation models for time series analysis: A tutorial and survey," *arXiv preprint arXiv:2403.14735*, 2024.

[27] A. Das et al., "A decoder-only foundation model for time-series forecasting," in *International Conference on Machine Learning (ICML)*, 2024.

[28] A. B. Habib, M. G. R. Alam, and M. Z. Uddin, "AUNET (attention-based unified network): Leveraging attention-based N-BEATS for enhanced univariate time series forecasting," *IEEE Access*, vol. 13, pp. 1–15, 2025. DOI: 10.1109/ACCESS.2025.3529183.

[29] Z. Liu et al., "Breaking silos: Adaptive model fusion unlocks better time series forecasting," in *International Conference on Machine Learning (ICML)*, 2025.

[30] A. M. Mansourian et al., "A comprehensive survey on knowledge distillation," *Transactions on Machine Learning Research (TMLR)*, 2025.

[31] S. Zagoruyko and N. Komodakis, "Paying more attention to attention: Improving the performance of convolutional neural networks via attention transfer," in *International Conference on Learning Representations (ICLR)*, 2017.

[32] W. Park, D. Kim, Y. Lu, and M. Cho, "Relational knowledge distillation," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019, pp. 3967–3976.

[33] F. Tung and G. Mori, "Similarity-preserving knowledge distillation," *IEEE International Conference on Computer Vision (ICCV)*, pp. 1365–1374, 2019.

[34] Z. Allen-Zhu and Y. Li, "Towards understanding ensemble, knowledge distillation and self-distillation in deep learning," *arXiv preprint arXiv:2012.09816*, 2020.

[35] A. Liu, J. Ma, and G. Zhang, "Adapting to the unknown: Robust meta-learning for zero-shot financial time series forecasting," *arXiv preprint arXiv:2504.09664*, 2025.

[36] Y. Zhang, H. Wang, and M. Liu, "LSTM-based deep learning for agricultural commodity price prediction in emerging markets," *Expert Systems with Applications*, vol. 215, p. 119 378, 2023.

[37] F. Weng, Y. Chen, Z. Wang, et al., "Attention-based LSTM for food price prediction considering climate factors," *Journal of Forecasting*, vol. 41, no. 5, pp. 1043–1058, 2022.

[38] X. Li, W. Zhang, and J. Chen, "Food price forecasting using machine learning and deep learning: A systematic review," *Agricultural Economics*, vol. 55, no. 2, pp. 189–215, 2024.

[39] Y. Cheng, D. Wang, P. Zhou, and T. Zhang, "Model compression and acceleration for deep neural networks: The principles, progress, and challenges," *IEEE Signal Processing Magazine*, vol. 35, no. 1, pp. 126–136, 2018.

[40] S. Han, H. Mao, and W. J. Dally, "Deep compression: Compressing deep neural networks with pruning, trained quantization and Huffman coding," *arXiv preprint arXiv:1510.00149*, 2016.

[41] Z. Liu, Y. Wang, S. Vaidya, F. Ruber, et al., "KAN: Kolmogorov-arnold networks," *arXiv preprint arXiv:2404.19756*, 2024.

Appendix A: Configuration File

The following YAML configuration file represents the best-performing setup used in the experiments:

```yaml
seed: 1337

data:
  url: "https://raw.githubusercontent.com/shifatzaman/
     datasets/..."
  market: "Dhaka Division"
  n_commodities: 4
  freq: "ME"
  agg: "median"

task:
  input_len: 6
  horizon: 1
  scale: "minmax"
  target: "price"

teachers:
  common:
    hidden: 128
    dropout: 0.1
  models:
    dlinear:
      kernel_size: 25
    patchtst:
      d_model: 128
      nhead: 8
      num_layers: 3
      d_ff: 256
      patch_len: 8
    nbeats:
      n_stacks: 3
      n_blocks: 3
      hidden: 256

students:
  mlp:
    hidden_dims: [512, 256, 128]
    dropout: 0.2
  gru:
```

```
    hidden: 256
    n_layers: 4
    dropout: 0.2
  kan:
    hidden: 256
    grid_size: 32

distill:
  enabled: true
  uncertainty_weighted: true
  uncertainty_alpha: 2.0
  teacher_temp: 0.3
  losses:
    hard:
      type: mae
      weight: 0.3
    kd_pred:
      type: mse
      weight: 0.6
      enabled: true
    kd_feat:
      enabled: true
      weight: 0.15
      proj_dim: 256
    kd_diff:
      enabled: true
      weight: 0.1

train:
  epochs: 100
  batch_size: 16
  lr: 2.0e-4
  weight_decay: 1.0e-5
  grad_clip: 1.0
  early_stopping:
    enabled: true
    patience: 15
    min_delta: 0.001
  scheduler:
    type: reduce_on_plateau
    factor: 0.5
    patience: 5

experiments:
  grid:
    enabled: true
    teacher_sets:
      - [dlinear]
      - [patchtst]
      - [nbeats]
      - [dlinear, patchtst]
```

```
      - [dlinear, nbeats]
      - [patchtst, nbeats]
      - [dlinear, patchtst, nbeats]
    students: [mlp, gru, kan]
```

Listing 5.1: Best performing configuration (configs/default.yaml)

Key Configuration Parameters

Data Settings

- **market**: Target market (Dhaka Division)
- **n_commodities**: Number of commodities to use (4: Lentils, Oil, Rice, Wheat flour)
- **freq**: Resampling frequency (ME = Month End)
- **agg**: Aggregation method for resampling (median)

Task Settings

- **input_len**: Historical window length in months (6)
- **horizon**: Forecast horizon (1 month ahead)
- **scale**: Normalization method (minmax critical for performance)

Distillation Settings

- **uncertainty_weighted**: Enable uncertainty-based weighting (true)
- **uncertainty_alpha**: Sensitivity to teacher variance (2.0)
- **teacher_temp**: Temperature for softmax ensemble weights (0.3)
- **Loss weights**: hard=0.3, kd_pred=0.6, kd_feat=0.15, kd_diff=0.1

Training Settings

- **epochs**: Maximum training epochs (100)
- **batch_size**: Training batch size (16)
- **lr**: Learning rate (2e-4 for students)
- **patience**: Early stopping patience (15 epochs)

Appendix B: Code Availability

The complete source code for this research is publicly available for reproducibility and further research.

Repository Information

Repository URL: https://github.com/shifatzaman/wfp_kd_forecasting

Repository Structure

```
wfp_kd_forecasting/
|-- configs/
|   |-- default.yaml          # Main configuration file
|   |-- experiments/          # Experiment-specific configs
|-- src/
|   |-- data/
|   |   |-- loader.py         # Data loading utilities
|   |   |-- preprocessing.py  # Data preprocessing
|   |-- models/
|   |   |-- teachers/
|   |   |   |-- dlinear.py    # DLinear model
|   |   |   |-- patchtst.py   # PatchTST model
|   |   |   |-- nbeats.py     # N-BEATS model
|   |   |-- students/
|   |   |   |-- mlp.py        # MLP student
|   |   |   |-- gru.py        # GRU student
|   |   |   |-- kan.py        # KAN student
|   |-- training/
|   |   |-- trainer.py        # Training loop
|   |   |-- distillation.py   # KD loss functions
|   |-- evaluation/
|       |-- metrics.py        # Evaluation metrics
|-- runs/                     # Experiment outputs
|-- requirements.txt          # Python dependencies
|-- run.py                    # Main entry point
```

Installation

```
# Clone the repository
git clone https://github.com/[username]/wfp_kd_forecasting.
    git
cd wfp_kd_forecasting

# Create virtual environment
python -m venv venv
source venv/bin/activate  # On Windows: venv\Scripts\activate

# Install dependencies
pip install -r requirements.txt
```

Running Experiments

Train with Default Configuration

```
python run.py --config configs/default.yaml
```

Run Specific Teacher-Student Combination

```
python run.py --config configs/default.yaml \
    --teachers dlinear patchtst nbeats \
    --student mlp
```

Run Full Experimental Grid

```
python run.py --config configs/default.yaml --grid
```

Dependencies

The following Python packages are required:

Package	Version
Python	≥ 3.10
PyTorch	$\geq 2.1.0$
NumPy	$\geq 1.24.0$
Pandas	$\geq 2.0.0$
scikit-learn	$\geq 1.3.0$
PyYAML	≥ 6.0
tqdm	$\geq 4.65.0$
matplotlib	$\geq 3.7.0$

Reproducing Main Results

To reproduce the main results reported in this study (MAE = 1.959 BDT/unit):

```
# Ensure you're using the exact configuration
python run.py --config configs/default.yaml \
    --seed 1337 \
    --teachers dlinear patchtst nbeats \
    --student mlp

# Results will be saved to runs/[timestamp]/
# Check runs/[timestamp]/results.json for metrics
```

License

This code is released under the MIT License for academic and research purposes.

Citation

If you use this code in your research, please cite:

```
@mastersthesis{shifat2026WFPMultiKd,
  title={A Lightweight Time-series Analysis Model
        through Multi-Teacher Knowledge Distillation
        for Food Price Forecasting},
  author={Shifat Zaman},
  school={BRAC University},
  year={2026},
  type={M.Sc. Thesis}
}
```