Universidade Estadual de Campinas
Instituto de Computação

Sadeeq Olalekan Bello

Cognitive Architecture-Driven Ensemble Learning for Real-Time Churn Prediction

Aprendizado Ensemble Baseado em Arquitetura Cognitiva para Previsão de Rotatividade em Tempo Real

CAMPINAS
2024

Sadeeq Olalekan Bello

Cognitive Architecture-Driven Ensemble Learning for Real-Time Churn Prediction
Aprendizado Ensemble Baseado em Arquitetura Cognitiva para Previsão de Rotatividade em Tempo Real

Dissertação apresentada ao Instituto de Computação da Universidade Estadual de Campinas como parte dos requisitos para a obtenção do título de Mestre em Ciência da Computação.

Dissertation presented to the Institute of Computing of the University of Campinas in partial fulfillment of the requirements for the degree of Master in Computer Science.

Supervisor/Orientador: Prof. Dr. Marcelo da Silva Reis
Co-supervisor/Coorientador: Prof. Dr. Julio Cesar dos Reis

Este exemplar corresponde à versão final da Dissertação defendida por Sadeeq Olalekan Bello e orientada pelo Prof. Dr. Marcelo da Silva Reis.

CAMPINAS
2024

Ficha catalográfica
Universidade Estadual de Campinas (UNICAMP)
Biblioteca do Instituto de Matemática, Estatística e Computação Científica
Ana Regina Machado - CRB 8/5467

B417c Bello, Sadeeq Olalekan, 1991-
Cognitive architecture-driven ensemble learning for real-time churn prediction / Sadeeq Olalekan Bello. – Campinas, SP : [s.n.], 2024.

Orientador(es): Marcelo da Silva Reis.
Coorientador(es): Julio Cesar dos Reis.
Dissertação (mestrado) – Universidade Estadual de Campinas (UNICAMP), Instituto de Computação.

1. Arquitetura cognitiva. 2. Aprendizado de máquina. 3. Predição de comportamento de cliente. I. Marcelo da Silva Reis, 1979-. II. Julio Cesar dos Reis, 1987-. III. Universidade Estadual de Campinas (UNICAMP). Instituto de Computação. IV. Título.

Informações complementares

Título em outro idioma: Aprendizado ensemble baseado em arquitetura cognitiva para previsao de rotatividade em tempo real
Palavras-chave em inglês:
Cognitive architecture
Machine learning
Customer behavior prediction
Área de concentração: Ciência da Computação
Titulação: Mestre em Ciência da Computação
Banca examinadora:
Marcelo da Silva Reis [Orientador]
Ricardo Ribeiro Gudwin
Fabricio Martins Lopes
Data de defesa: 19-12-2024
Programa de Pós-Graduação: Ciência da Computação

Identificação e informações acadêmicas do(a) aluno(a)
- ORCID do autor: https://orcid.org/0009-0004-4743-9368
- Currículo Lattes do autor: http://lattes.cnpq.br/1115543769135241

Universidade Estadual de Campinas
Instituto de Computação

Sadeeq Olalekan Bello

Cognitive Architecture-Driven Ensemble Learning for Real-Time Churn Prediction
Aprendizado Ensemble Baseado em Arquitetura Cognitiva para Previsão de Rotatividade em Tempo Real

Campinas, 19 de dezembro de 2024

Vita brevis,
ars longa,
occasio praeceps,
experimentum periculosum,
iudicium difficile.

(Hippocrates)

Acknowledgements

The act of gratitude is one of the highest virtues, and I humbly take this opportunity to express my deepest thanks to all who have contributed to the successful completion of my dissertation. This work is not the result of my efforts alone but of the unwavering support, encouragement, and guidance I received from numerous individuals and organization throughout this academic journey.

First and foremost, I am profoundly grateful to Almighty Allah, the beneficent and most merciful, for His infinite blessings, guidance, and sustenance. It is through His mercy that I have been able to navigate the challenges of this journey with strength, perseverance, and clarity of purpose.

I owe immense gratitude to my advisor, Professor Marcelo da Silva Reis, for his unwavering guidance, expertise, and encouragement. His insightful feedback and tireless commitment have been instrumental in shaping the direction and quality of this research. I am equally indebted to my co-advisor, Professor Julio Cesar dos Reis, whose thoughtful advice and intellectual contributions have enriched this dissertation. Their mentorship has left an indelible mark on my academic and professional growth.

This research would not have been possible without the support of the Ministry of Science, Technology and Innovation, with resources from Law No. 8,248, of October 23, 1991, within the scope of PPI-SOFTEX, coordinated by Softex and published in Cognitive Architecture (Phase 3), DOU 01245.003479/2024-10.

To my siblings Taofeeq Bello, Ahmed Bello, and Sheriff Bello your steadfast support, encouragement, and belief in my abilities have been a constant source of inspiration. Thank you for always being there as pillars of strength and for understanding the demands of this journey. To my beloved fiancée, Abdulah Zulayah, words cannot fully capture the depth of my gratitude for your unwavering patience, love, and encouragement. Your support has been the foundation upon which I stood, especially during the most challenging moments.

I also extend my heartfelt appreciation to my friends and colleagues Fillipe Santos, Adebowale Tope, Akinmuda Olusegun, Moshood Kehinde, Seyed Haddadi, Patrick Araujo, Gabriel Kakimoto, Yuji Sakabe, Gabriel Oliveira, Joao Victor, Arthur Hendricks, Joao Cadenuto for their intellectual discussions, and words of encouragement that have made this journey both rewarding and memorable. Your support, whether through professional collaboration or personal motivation, has been invaluable.

To everyone whose paths I have crossed during this academic endeavor, your contributions, no matter how small they may seem, have left an imprint on my success. Whether through kind words, shared experiences, or professional advice, I am forever grateful for your presence in this chapter of my life.

Lastly, to those unmentioned but not forgotten, your support, prayers, and encouragement have meant the world to me. This work is as much yours as it is mine, and I hope it stands as a testament to the collective effort that made it possible.

Resumo

A rotatividade de clientes continua sendo um problema urgente para empresas em serviços baseados em assinatura, telecomunicações e mercados on-line, onde reter clientes é vital para o sucesso a longo prazo. Prever a rotatividade, principalmente em tempo real, apresenta desafios significativos devido à natureza dinâmica e complexa do comportamento do cliente. Os modelos preditivos tradicionais geralmente dependem de dados estáticos e regras fixas, o que limita sua capacidade de se adaptar aos padrões em constante mudança das ações do consumidor. Essa limitação dificulta a identificação rápida e precisa de clientes em risco. O principal desafio na previsão de rotatividade em tempo real está no processamento de fluxos de dados contínuos e no ajuste dinâmico de modelos para levar em conta as flutuações no comportamento do cliente, tudo isso mantendo alta precisão preditiva. Esta dissertação de mestrado investiga e avalia um sistema de previsão adaptável que pode processar efetivamente dados de clientes em tempo real e se ajustar dinamicamente às mudanças nos padrões de comportamento, mantendo alta precisão de previsão. Esta pesquisa propõe e avalia uma arquitetura cognitiva que incorpora aprendizado de conjunto dinâmico para previsão de rotatividade em tempo real. Nossa arquitetura, implementada por meio do Cognitive Systems Toolkit (CST), combina vários modelos de aprendizado de máquina cujas contribuições são ponderadas dinamicamente com base no desempenho anterior. O sistema aproveita a modularidade e a adaptabilidade para gerenciar grandes volumes de dados do cliente em tempo real, com codelets especializados lidando com processamento de dados, treinamento de modelo e ajuste de peso de conjunto. A avaliação conduzida implementa uma estrutura abrangente, testando o sistema em conjuntos de dados balanceados e desbalanceados. Avaliamos seis modelos de aprendizado de máquina (KNN, Regressão Logística, Árvore de Decisão, Floresta Aleatória, SVM e Naive Bayes) individualmente e em configurações de conjunto. A análise compara abordagens de conjunto estáticas versus dinâmicas, emprega validação cruzada em várias dobras de dados e valida os resultados por meio de testes estatísticos usando o teste de classificação assinada de Wilcoxon. Nossas descobertas demonstram que a abordagem de conjunto dinâmico da arquitetura cognitiva supera significativamente os modelos individuais e conjuntos estáticos, alcançando 95,92% de precisão em dados balanceados em comparação com 82,93% em dados desbalanceados. A solução mostra força particular no tratamento de padrões de dados dinâmicos, com significância estatística em melhorias de precisão ($p = 0,0098$) e recall ($p = 0,0020$). A validação entre conjuntos de dados foi conduzida em três conjuntos de dados de telecomunicações (IBM Telco, Churn-data-UCI e Telco-Customer-Churn), confirmando a generalização e robustez da arquitetura dentro do domínio das telecomunicações. Embora a avaliação tenha sido específica para conjuntos de dados de telecomunicações, o design modular da abordagem sugere aplicabilidade potencial a outros setores, o que justifica uma investigação mais aprofundada. Nossas descobertas destacam a eficácia da combinação de arquiteturas cognitivas com aprendizado de conjunto dinâmico para abordar as complexidades da previsão de rotatividade de clientes em setores acelerados e orientados a dados.

Palavras-chave: Rotatividade de clientes; Arquitetura cognitiva; Aprendizado de conjunto; Previsão em tempo real; Retenção de clientes; Kit de ferramentas do sistema cognitivo

Abstract

Customer churn remains a pressing issue for businesses in subscription-based services, telecommunications, and online marketplaces, where retaining customers is vital for long-term success. Predicting churn, particularly in real-time, presents significant challenges due to customer behavior's dynamic and complex nature. Traditional predictive models often rely on static data and fixed rules, which limit their ability to adapt to the ever-changing patterns of consumer actions. This limitation makes it difficult to identify at-risk customers promptly and accurately. The core challenge in real-time churn prediction lies in processing continuous data streams and dynamically adjusting models to account for fluctuations in customer behavior, all while maintaining high predictive accuracy. This M.Sc. dissertation investigates and evaluates an adaptive prediction system that can effectively process real-time customer data and dynamically adjust to changing behavior patterns while maintaining high prediction accuracy. This research proposes and evaluates a cognitive architecture incorporating dynamic ensemble learning for real-time churn prediction. Our architecture, implemented through the Cognitive Systems Toolkit (CST), combines multiple machine learning models whose contributions are dynamically weighted based on previous performance. The system leverages modularity and adaptability to manage large volumes of customer data in real-time, with specialized codelets handling data processing, model training, and ensemble weight adjustment. The conducted evaluation implements a comprehensive framework, testing the system across both balanced and imbalanced datasets. We evaluate six machine learning models (KNN, Logistic Regression, Decision Tree, Random Forest, SVM, and Naive Bayes) individually and in ensemble configurations. The analysis compares static versus dynamic ensemble approaches, employs cross-validation across multiple data folds, and validates results through statistical testing using the Wilcoxon signed-rank test. Our findings demonstrate that the cognitive architecture's dynamic ensemble approach significantly outperforms individual models and static ensembles, achieving 95.92% accuracy on balanced data compared to 82.93% on imbalanced data. The solution shows particular strength in handling dynamic data patterns, with statistical significance in precision ($p = 0.0098$) and recall ($p = 0.0020$) improvements. Cross-dataset validation was conducted on three telecommunication datasets (IBM Telco, Churn-data-UCI, and Telco-Customer-Churn), confirming the architecture's generalizability and robustness within the telecommunication domain. While the evaluation was specific to telecommunication datasets, the approach's modular design suggests potential applicability to other industries, which warrants further investigation. Our findings highlight the effectiveness of combining cognitive architectures with dynamic ensemble learning in addressing the complexities of customer churn prediction in fast-paced, data-driven industries.

Keywords: Customer Churn; Cognitive Architecture; Ensemble Learning; Real-Time Prediction; Customer Retention; Cognitive System Toolkit

List of Figures

2.1 Flowchart depicting our systematic review process following PRISMA guidelines. 25
2.2 The core of the CST architecture. Codelets (rectangular blocks) process information independently and interact with Memory Objects (yellow circles) through input/output connections (LI, GI, and Out). The Raw Memory stores Memory Objects which contain Information (I), Time Stamp (T), and Evaluation (E) properties. Codelets are organized into Coalitions for coordinated processing, managed by the Coderack scheduling system. . . . 33

3.1 The proposed cognitive architecture framework for real-time churn prediction. 41
3.2 Workflow of the proposed cognitive architecture methodology. 44

4.1 Systematic approach for assessing model and ensemble performance across multiple datasets and scenarios. The framework implements incremental data processing to evaluate both static performance metrics and dynamic adaptation capabilities. 64
4.2 Model accuracy comparison between balanced and imbalanced datasets. . . 68
4.3 Model F1-Score comparison between balanced and imbalanced datasets. . . 69
4.4 Dynamic ensemble accuracy over time (balanced vs. imbalanced datasets). 70
4.5 Static ensemble accuracy over time (balanced vs. imbalanced datasets). . . 72
4.6 Model weights over time for the imbalanced dataset. 76
4.7 Model weights over time for the balanced dataset. 77
4.8 Accuracy across different folds, for the imbalanced (left) and balanced (right) datasets. 78
4.9 Precision across different folds, for the imbalanced (left) and balanced (right) datasets. 78
4.10 F1-Score across different folds, for the imbalanced (left) and balanced (right) datasets. 79

List of Tables

2.1 Churn prediction studies reviewed in this dissertation. 29
2.2 Cognitive architecture studies reviewed in this dissertation. 31

3.1 Description of features in the Churn-in-Telecom Dataset. The features represent various aspects of customer behavior and service usage patterns. 46
3.2 Dataset Characteristics Summary. The Class Distribution column shows the number of churned customers versus non-churned customers (churn/not-churn) for each dataset. 61

4.1 Performance metrics for imbalanced Churn-in-Telecom dataset, the evaluation of individual models on the imbalanced version of the dataset. 66
4.2 Performance metrics for the balanced Churn-in-Telecom dataset, highlighting the top two performing models across all evaluation metrics. 67
4.3 Performance metrics with various models removed from the ensemble. The table highlights the significant drop in accuracy for both imbalanced and balanced datasets when Decision Tree and Random Forest are removed. . . 74
4.4 Results of the performance measures using a significant threshold of 0.05. . 80
4.5 Results of the performance measures using a significant threshold of 0.05. . 81
4.6 Model performance comparison across different datasets (highlighted cells indicate top two performing models by accuracy for each dataset). 83
4.7 Dynamic vs. static ensemble accuracy over time for IBM Telco dataset. Values highlighted in green indicate where the dynamic ensemble achieved higher accuracy compared to the static ensemble. 84
4.8 Dynamic vs. static ensemble accuracy over time for Churn-data-UCI dataset. 84
4.9 Accuracy comparison between the adaptive ensemble and the proposed RTCP dynamic ensemble across different datasets. 85

Contents

Glossary **14**

1 Introduction **15**
1.1 Context and Motivation . . . 15
1.2 Importance of Real-Time Churn Prediction . . . 17
1.3 Problem Characterization . . . 18
1.4 Objective and Justification . . . 19
1.5 Research Questions . . . 21
1.6 Significance of the Study . . . 21
1.7 Research Contributions . . . 22
1.8 Dissertation Organization . . . 23

2 Fundamental Concepts and Literature Review **24**
2.1 Literature Review Process . . . 25
2.2 Customer Churn: Concepts and Importance . . . 26
2.3 Traditional Churn Prediction Methods . . . 26
2.4 Real-Time Churn Prediction . . . 27
2.5 Cognitive Architectures: An Overview . . . 29
2.6 Ensemble Learning . . . 34
2.7 Findings and Identification of Research Gaps . . . 37
2.8 Positioning and Discussion . . . 38
2.9 Final Remarks . . . 39

3 Methodology **41**
3.1 Cognitive Architecture Implementation . . . 42
3.2 Workflow of the Proposed Methodology . . . 44
3.3 Data Processing and Analysis . . . 45
3.3.1 Dataset Structure and Features . . . 46
3.3.2 Data Preprocessing . . . 47
3.3.3 Model Evaluation Framework . . . 49
3.3.4 Evaluation Metrics . . . 49
3.3.5 Data Flow Management . . . 49
3.4 Implementation . . . 51
3.4.1 Memory Object Implementation . . . 51
3.4.2 Codelet System Implementation . . . 51
3.4.3 Integration and Execution Framework for Coordinating Ensemble Learning Within the Cognitive Architecture . . . 52
3.4.4 Execution Flow Implementation . . . 52
3.5 Dynamic Weight Management . . . 53

3.6 Experimental Setup 56
3.7 Computational Environment 58
3.8 Multi-Dataset Evaluation 61
3.9 Comparative Framework 61
3.10 Final Remarks 62

4 Results and Discussion **64**
4.1 Model Effectiveness 65
4.1.1 Individual Model Performance 65
4.1.2 Comparison of Models 67
4.2 Ensemble Performance 69
4.2.1 Dynamic Weight Application Over Time 74
4.2.2 Statistical Test Results 79
4.3 Model Performance Across Datasets 82
4.4 Comparison with Existing Adaptive Ensemble Methods 84
4.5 Final Remarks 86

5 Conclusion **87**
5.1 Answering Research Questions 87
5.2 Key Findings 88
5.3 Implications of Findings 89
5.4 Broader Applications 90
5.5 Limitations and Future Research Directions 90
5.5.1 Limitations 90
5.5.2 Transitioning to a Fully Real-Time System 91
5.5.3 Expected Benefits 92
5.5.4 Future Research Directions 92
5.6 Final Thoughts 93

Bibliography **94**

Glossary

adaptability Adaptability is the capacity of the system to adjust dynamically to changes in data patterns, operational requirements, or environmental conditions without the need for manual intervention or frequent retraining. For the cognitive architecture framework, adaptability is achieved through mechanisms such as dynamic weight management in ensemble learning, which allows the system to prioritize high-performing models and respond in real time to shifts in customer behavior or data distribution..

real-time Real-time refers to the system's ability to process, analyze, and respond to incoming data continuously and within a constrained timeframe, often as events occur. In the context of churn prediction, real-time capability enables the system to provide actionable predictions based on live or near-live customer data, allowing for immediate interventions. Although the current implementation simulates real-time functionality, it highlights the potential for handling continuous data streams through dynamic model updates and incremental learning processes..

scalability Scalability refers to the ability of a system, process, or model to efficiently handle increasing amounts of data, workload, or computational resources without significant loss in performance or functionality. In the context of the proposed framework, scalability is demonstrated by the cognitive architecture's capacity to integrate additional machine learning models, process larger datasets, or adapt to increased data flow without requiring extensive redesigns or diminishing prediction..

Chapter 1

Introduction

1.1 Context and Motivation

Customer churn prediction is a vital aspect of customer relationship management, which enables businesses to identify and retain at-risk customers. By leveraging advanced machine learning techniques and data analytics, companies can proactively address churn, ultimately enhancing customer satisfaction and profitability [1]. As indicated by Aishwarya *et al.* [2], companies may experience involuntary churn if their clients relocate to a distant location, move to a long-term care facility, or pass away. Voluntary churn occurs when a consumer migrates to another business or a service provider. In most applications, analytical models omit the involuntary churn factors.

The current challenge in churn prediction is not simply about detecting patterns in historical data, but about identifying them in real-time. As noted by Aishwarya *et al.* [2], traditional machine learning models depend heavily on past customer interactions and often overlook the dynamic nature of customer behavior. Moreover, the sheer volume of data produced in today's digital world compounds is difficult to predict, particularly when models must adapt to rapidly shifting customer behaviors in real-time. The challenge is magnified in telecommunications, for instance, where churn prediction is crucial because of intense competition and the ease with which customers can switch providers.

The increasing complexity of churn prediction has prompted researchers to explore more advanced machine learning methods, such as ensemble learning and deep learning, which have been proven to enhance the prediction accuracy [2]. These methods allow for the analysis of large volumes of data, including transaction history and customer usage metrics, and offer significant improvements in prediction precision and efficiency. However, the use of static datasets remains a limitation because customer behavior often evolves in ways that static models cannot adequately address. For instance, Huang *et al.* [3] highlighted how real-time churn prediction models are often undermined by the inability to process and adapt to real-time shifts in customer behavior.

Real-time churn prediction introduces new possibilities for customer churn prediction, enabling companies to address churn as it occurs [4]. This approach is essential for industries such as telecommunications, online gaming, and e-commerce, where customers expect rapid responses to their behavior. However, despite these advancements, real-time churn prediction remains a significant challenge [5].

One of the most critical challenges is the processing and analysis of extensive, real-time data. Real-time churn prediction models must efficiently process continuous data streams, which requires substantial computational power [6]. Consequently, the models must balance the speed of data processing with the accuracy of predictions, treating real-time data flux as folds for practical analysis and efficient handling. Real-time churn prediction introduces complexities that traditional batch-processing models fail to manage, such as rapidly shifting consumer preferences, external market pressures, and behavioral patterns [7]. Arnab *et al.* [7] emphasized that real-time models must be dynamic and flexible to handle unpredictable shifts in customer behavior.

In addition, models must be adaptable to new customer behavior patterns without requiring frequent retraining. Aishwarya *et al.* [2] explained that many real-time models struggle with the issue of data drift, where the underlying data distribution changes over time, leading to inaccurate predictions if the models are not frequently updated. The combination of data imbalance, where churn events are less frequent than non-churn events, and the inability to dynamically adjust to new data further complicates real-time churn predictions.

These challenges in real-time churn prediction require a solution that can handle continuous data processing while adapting to the evolving patterns. Cognitive architectures address these challenges by processing and adjusting to dynamic data streams. As defined by Christian *et al.* [8], a cognitive architecture is a computational framework that models human cognition, providing systems that can efficiently process continuous data streams and adapt their responses based on new information without complete system retraining. These characteristics make cognitive architectures particularly suitable for real-time churn prediction because they can handle the complex, multidimensional patterns that emerge in customer behavior analysis while maintaining adaptability to new data patterns [8].

The increase in modern industrial technologies, including smart factories and large-scale data analyses, has made accurate churn predictions more critical. These new methods provide more information about the use of products and services. As noted by Wolniak *et al.* [9], real-time analytics provides businesses with the means to promptly detect churn trends and forecast which customers are likely to churn, thereby facilitating the implementation of timely intervention plans.

Several studies have highlighted the effectiveness of real-time data analytics in improving customer retention. Wolniak *et al.* [9] emphasized the relevance of real-time data collection and analysis for identifying churn trends within Industry 4.0. This study underscores the significance of adaptability in customer relationship management, as it enables businesses to respond promptly to potential churn. Similarly, Gokulnath *et al.* [10] employed Azure's data analytics functionalities to showcase the capacity of real-time data processing to manage substantial amounts of data, thereby facilitating timely and well-informed decision making.

In summary, the rationale for Real-Time Churn Prediction (RTCP) is to address the complexity of customer behavior. In the digital age, incorporating real-time analytics into churn prediction models is a powerful method of enhancing customer retention.

Recent advancements in RTCP methodologies have introduced various machine learning approaches to address customer churn. Long Short-Term Memory (LSTM) networks have shown promise for capturing temporal patterns in customer behavior [11]. In addition, deep-learning models and gradient-boosting algorithms have effectively processed real-time customer data streams. These methods have improved the ability to detect early warning signs of customer churn through the real-time analysis of customer interactions, transaction patterns, and usage behaviors[10].

Despite these methodological advancements, significant obstacles remain in effectively forecasting customer churns. The constantly shifting dynamics of customer actions and complex characteristics of modern market environments pose challenges for industry practitioners. These obstacles underscore the necessity of enhanced and adaptable approaches to churn prediction. One significant challenge pertains to managing imbalanced data, in which the number of churners is lower than that of non-churners, leading to potential biases in prediction outcomes. Various approaches, such as cost-sensitive learning techniques including focal loss and weighted loss, have demonstrated superior effectiveness to conventional resampling methods in addressing this particular issue [12].

1.2 Importance of Real-Time Churn Prediction

The RTCP represents a significant shift in how businesses approach customer retention. Unlike traditional models that react to churn after it occurs, RTCP allows companies to detect potential churn in real time, enabling immediate interventions to prevent customer loss and improve loyalty [12].

Hasan *et al.* [11] highlighted the effectiveness of using a Long Short-Term Memory (LSTM) neural network in forecasting customer churn in real-time within commercial sectors. Their findings demonstrated that these real-time models can provide more accurate and prompt customer behavior analysis, enabling timely corrective actions.

Real-time prediction models can provide businesses with a means of maximizing resources. According to Mallegowda *et al.* [13], parallel data-mining strategies expedite the churn prediction process and enable near-real-time analysis. This capability is essential for businesses that require swift responses to customer demand and market shifts.

In the e-commerce domain, Nalla *et al.* [14] highlighted the potential of real-time predictive analytics to increase conversion rates and reduce churn by delivering personalized recommendations and prompt interventions. Similarly, Soni *et al.* [15] stress the limitations of models that do not incorporate real-time data, thereby underscoring the significance of real-time capabilities in bolstering customer retention efforts in the banking sector.

The critical relevance of real-time churn prediction lies in its ability to transform customer retention strategies from reactive to proactive ones. Real-time data digestion and processing help businesses have a more transparent window into customer needs as they emerge [15]. Companies can leverage real-time data processing to gain accurate insights into customer preferences, thus enabling swift and effective responses. Integrating real-time analytics into churn prediction models is not merely an enhancement, but an essential requirement for businesses striving to succeed in highly competitive, data-driven environments.

Given the significance of Real-Time Churn Prediction (RTCP) in today's business environment, this M.Sc. Dissertation introduces a novel approach that integrates cognitive architectures modular systems inspired by human cognition into churn prediction models. Cognitive architectures enable adaptive and dynamic processing, which is particularly suited for handling the complexities of real-time data. By harnessing these capabilities, our solution seeks to overcome the limitations of traditional methods while addressing gaps observed in recent churn prediction strategies. This approach aims to provide more accurate and actionable insights for enhancing customer retention strategies by building on both established practices and emerging methodologies.

1.3 Problem Characterization

Despite significant advancements in churn prediction methods, accurately predicting when customers might leave remains challenging. Traditional models, which often rely on static historical data, struggle to capture the dynamic and fluid nature of customer interactions in real-time. This limitation underscores the need for innovative approaches to process and interpret continuous data streams, allowing for early detection of churn indicators and timely interventions. Nguyen *et al.* [12] emphasized the relevance of incorporating contemporary data into churn prediction models by investigating longitudinal and survival data modeling. The authors argued that conventional prediction methods mainly use static data and are inadequate for capturing dynamic customer behavior patterns over time, shaped by various factors such as market trends, competitive moves, and individual customer experiences.

A concrete example from the Security as a Service (SaaS) industry demonstrates the practical advantages of real-time churn prediction over traditional methods. Chakraborty *et al.* [7] implemented a real-time clickstream modeling framework for QuickBooks Online (QBO) that combines both current session data and historical user behavior to predict subscription cancellations. Their approach enabled CRM teams to design in-session proactive interventions to retain users before they churn, demonstrating a clear advantage over traditional models that only analyze historical data. This real-time approach allows businesses to:

- Detect potential churn indicators during active user sessions;
- Combine both immediate and historical user behavior patterns for more accurate predictions;
- Implement immediate intervention strategies.

Golkunath *et al.* [10] further emphasized the challenge of the vast amounts of data generated in sectors such as telecommunications and e-commerce. Real-time data processing is essential for managing this influx and for making prompt informed decisions. However, the existing models cannot effectively handle these demands.

Given the obstacles to managing large quantities of data and the constraints of traditional forecasting approaches, real-time churn prediction is essential. Real-time models process data quickly and continuously to analyze customer interactions, allowing for more timely and context-aware predictions. This enables businesses to respond rapidly to recent customer behavior as they progress, instead of relying solely on past data that may no longer be relevant. The capacity to handle and evaluate data in real time presents a promising solution to the complexities of consumer behavior [16].

1.4 Objective and Justification

This M.Sc. dissertation investigates and evaluates a cognitive architecture specifically designed for real-time churn prediction, addressing the limitations identified in existing approaches. Current methodologies often fail to adapt dynamically to real-time data streams, lack modularity, or focus on domain-specific applications, such as telecommunications, without considering broader contexts. This work bridges these gaps by proposing a solution that leverages cognitive architectures for adaptability and scalability across different industries, with an initial focus on telecommunications. The primary objectives of this study are threefold:

- Design a cognitive architecture that effectively processes real-time customer behavior data;
- Implement dynamic ensemble learning within the cognitive framework to leverage the strengths of multiple models while adapting to real-time data. This approach aims to improve the flexibility and accuracy of churn prediction by dynamically adjusting the model contributions based on their performance on recent data.
- Evaluate the system's effectiveness in real-time churn prediction scenarios to assess its potential to offer timely and accurate customer insights.

The justification for this research stems from the limitations identified in traditional churn prediction models in managing real-time data streams. Traditional methods often rely on batch processing, which significantly limits their ability to provide timely interventions in dynamic environments [17, 18]. For instance, systems like RICON demonstrated the potential of real-time prediction but faced computational scalability challenges when handling large datasets [17]. Similarly, models like the two-dimensional segment analysis proposed by Seo *et al.* [19] effectively reduced churn rates but required intensive data processing, which hinders their real-time applicability.

Our research introduces a novel application of cognitive architectures in real-time churn prediction (RTCP) using the Cognitive System Toolkit (CST) [20]. Cognitive architectures, such as MECA and CogToM, have demonstrated their ability to handle dynamic environments by integrating modularity and adaptive processes, as seen in applications like urban traffic control and autism spectrum interaction modeling [21, 22]. However, their integration with predictive analytics remains largely unexplored. This research gap presents the following opportunities:

- **Bridge the Divide Between Cognitive Computing and Predictive Analytics:** Cognitive architectures like MECA and CogToM highlight the importance of modularity and adaptability, which are essential for bridging cognitive computing with predictive tasks. Recent works [23] underline how these architectures can enhance decision-making processes in dynamic contexts, presenting a clear pathway for their application in churn prediction.

- **Develop More Adaptive and Responsive Churn Prediction Systems:** Adaptive ensemble frameworks, such as the work introduced by Shaikhsurab *et al.* [24], achieved remarkable accuracy but exhibited computational constraints and lacked generalizability across industries. Cognitive architectures provide an opportunity to overcome these limitations by enabling adaptive responses to shifting data patterns in real-time.

- **Address the Real-Time Processing Challenges Identified in Current Methods:** Systems like RICON and dynamic weighting approaches [17, 18] attempted to address real-time processing challenges but often required significant computational resources. Integrating ensemble learning within a cognitive architecture framework could enhance computational efficiency and scalability.

The selection of cognitive architecture over traditional ensemble learning approaches is motivated by several key factors:

- **Dynamic Adaptability:** Unlike static ensemble methods, cognitive architectures can continuously adjust to changing data patterns through their memory object system and codelet interactions.

- **Modular Processing:** The codelet-based architecture enables independent processing units that can be activated or deactivated based on performance, providing flexibility not available in traditional ensemble approaches.

- **Real-time Processing Capability:** The asynchronous nature of codelets and memory objects facilitates real-time data processing and model adaptation, surpassing the batch-processing limitations of conventional ensemble methods.

Through this research, we aim to address these gaps by combining cognitive architectures with ensemble learning. This integration leverages the adaptability of cognitive systems and the accuracy of ensemble methods, offering a robust and scalable solution for real-time churn prediction. By contextualizing this approach within the existing literature, our study establishes its novelty and potential impact.

We aimed to enhance the accuracy and timeliness of churn predictions by integrating ensemble learning with dynamic weighting within this cognitive framework. This approach offers several advantages.

- Continuous learning and adjustment capabilities;

- Improved handling of dynamic customer behavior patterns;

- Enhanced prediction accuracy through ensemble methods;

- Real-time adaptation to changing data patterns;

1.5 Research Questions

This study assessed the effectiveness of integrating dynamic ensemble learning within a cognitive architecture for real-time churn prediction. The following key research questions (RQs) guided this investigation.

RQ-1 How does dynamic ensemble learning within the proposed cognitive architecture improve prediction accuracy in real-time churn prediction?

RQ-2 How effectively does the proposed cognitive architecture adapt to real-time changes in data patterns?

Having identified our primary research questions, we focus on the broader outcomes and possible impacts of this study. By exploring these research questions, we aimed to drive the field of RTCP using various approaches. Our discoveries possess the capacity to increase our understanding of cognitive architecture frameworks in predictive analytics and provide actionable insights for enterprises working to enhance their customer retention strategies. The subsequent section outlines the key areas in which this study is expected to contribute to the field of RTCP.

1.6 Significance of the Study

We believe that the results of this study are significant in the RTCP domain. A cognitive architecture was developed to address the limitations of current prediction methods, and this study's importance lies in several areas.

The significance of this study lies in its potential to introduce a validated approach for customer retention strategies through real-time churn predictions. By developing a cognitive architecture that adapts dynamically to change in data pattern, this study provides a flexible approach to churn prediction. To evaluate its robustness, the system was tested on three diverse telecommunications datasets (IBM Telco, Churn-data-UCI, and Telco-Customer-Churn), showcasing its adaptability across different contexts and dynamic environments.

Key evaluation metrics, such as precision, recall, F1-score, were employed to quantify "changing data pattern of customers." For example, precision and recall were used to measure the system's ability to correctly identify churners in different data folds while adapting to changes in customer data patterns. These metrics underline the model's capacity to adjust dynamically, reflecting changes in customer behavior.

The modular architecture of our system facilitates the evaluation of the contributions of the individual components to the overall prediction. By segregating functions, such as data ingestion, model-specific predictions, and ensemble integration, we can systematically analyze the impact of each element on the final output. Furthermore, it allows for more straightforward modification and better improvement of specific components without affecting the entire system. This approach provides valuable insights into the prediction process and facilitates ongoing refinement and adaptation of the system to meet changing needs or incorporate new methodologies.

This study addresses the limitations of current prediction methods and offers broader applicability. The proposed cognitive architecture framework can be adapted to various domains that require dynamic real-time predictive systems, extending the potential impact of this research beyond its initial focus on churn prediction.

1.7 Research Contributions

This study builds on existing work in supervised machine learning and churn prediction by addressing the limitations of static ensemble methods through the integration of dynamic ensemble learning within a cognitive architecture. The primary contributions of this research are as follows:

1. **Integration of Dynamic Ensemble Learning within a Modular Cognitive Architecture:**

    - This study introduces a novel approach by integrating dynamic ensemble learning into a modular cognitive architecture tailored for real-time churn prediction. While ensemble methods are widely used in machine learning, the proposed framework employs dynamic weight adjustments that allow models to adapt in real-time based on their performance. Unlike static weighting schemes commonly found in traditional methods, this dynamic mechanism ensures that the ensemble remains responsive to evolving data patterns, thereby enhancing its predictive accuracy and reliability in dynamic environments.

2. **Explainability through Modular Design and Transparent Weight Management:**

    - The cognitive architecture is designed with modularity at its core, where each machine-learning model operates as an independent component. This modular structure not only facilitates scalability and flexibility but also enhances the explainability of the system. By isolating the roles and contributions of individual models, users can clearly understand how each component influences final predictions.

3. **Scalability and Flexibility:**

    - By combining cognitive architecture with ensemble learning, this study presents a scalable solution that can be extended to multiple domains beyond telecommunications. The modular design allows for the integration of additional machine learning models or datasets, ensuring broad applicability and ease of customization.

4. **Adaptation to Real-Time Data Streams:**

    - The proposed cognitive architecture leverages modularity to process real-time customer data, enabling the system to adapt dynamically to changes in customer data patterns in diferent folds of data. This contribution bridges the gap between static data-driven models and the need for real-time adaptability.

1.8 Dissertation Organization

The remainder of this dissertation is organized as follows:

- Chapter 2 - **Fundamental Concepts and Literature Review**
  - Covers the basics of customer churn prediction and its importance;
  - Discusses real-time churn prediction (RTCP);
  - Provides an overview of cognitive architectures;
  - Reviews ensemble learning techniques;
  - Presents systematic literature review methodology;
  - Analyzes existing approaches and identifies research gaps;
  - Positions current research within existing literature.
- Chapter 3 - **Methodology**
  - Details the implementation of the cognitive architecture;
  - Describes the workflow of the proposed methodology;
  - Covers data processing and model evaluation frameworks;
  - Discusses dynamic weight management and experimental setup.
- Chapter 4 - **Results and Discussion**
  - Presents the effectiveness of the model and compares results;
  - Analyzes ensemble performance and weight application over time;
  - Reports statistical test results and addresses reproducibility;
  - Answers the research questions and compares findings with existing adaptive ensemble methods.
- Chapter 5 - **Conclusion**
  - Summarizes key findings and discusses implications;
  - Explores broader applications and limitations;
  - Provides future research directions;
  - Ends with final thoughts on the research.

Chapter 2

Fundamental Concepts and Literature Review

This chapter provides an in-depth exploration of the key concepts essential for understanding this research. It addresses customer churn and its significance and then discusses traditional and real-time churn prediction methods. Subsequently, the chapter delves into cognitive architectures and the Cognitive System Toolkit (CST) framework (Subsection 2.5), which is crucial to the proposed approach. Finally, this chapter discusses ensemble learning and dynamic weighting, which are the integral components of the cognitive architecture employed in this study.

Real-time churn prediction is becoming increasingly vital across multiple industries due to the dynamic nature of customer behavior and the rapid pace of business operations [17]. Industries such as telecommunications, online gaming, and cloud-based services depend heavily on their ability to anticipate and respond swiftly to customer activity. In these sectors, where consumer preferences and behaviors shift frequently, conventional batch processing methods have proven insufficient. Instead, real-time churn prediction enables companies to act promptly, reducing the likelihood of customer loss and enhancing retention strategies [25]. The growing demand for real-time analytics allows businesses to deliver more timely interventions, optimize their service offerings, and better understand evolving customer preferences [18].

This chapter is organized into ten main sections. Following this introduction, Section 2.1 presents the literature review process and methodology. Section 2.2 explores customer churn concepts and their business impact. Section 2.3 examines traditional churn prediction methods, including classical machine learning and deep learning approaches. Section 2.4 investigates real-time churn prediction and its current challenges. Section 2.5 provides an overview of cognitive architectures and their evolution, including detailed examinations of ACT-R, Soar, ICARUS, and the Cognitive System Toolkit (CST) in Section 2.5. Section 2.6 discusses ensemble learning methods and their applications, including dynamic weighting approaches in Section 2.6. Section 2.7 identifies research gaps in current literature. Section 2.8 positions this research within the existing body of work, and Section 2.9 concludes with final remarks summarizing the chapter's key points.

2.1 Literature Review Process

To ensure a comprehensive review, we followed the PRISMA guidelines, using a structured approach to identify relevant literature. We employed Boolean expressions such as "real-time churn prediction" AND "churn prediction" AND "machine learning" and "cognitive architecture" AND "ensemble learning". The primary indexers used were ACM Digital Library, IEEE Xplore, and Google Scholar, covering the period from 2015 to 2024. The search yielded a total of 250 articles, of which 80 were selected for detailed analysis based on their direct relevance to real-time prediction and cognitive architectures.

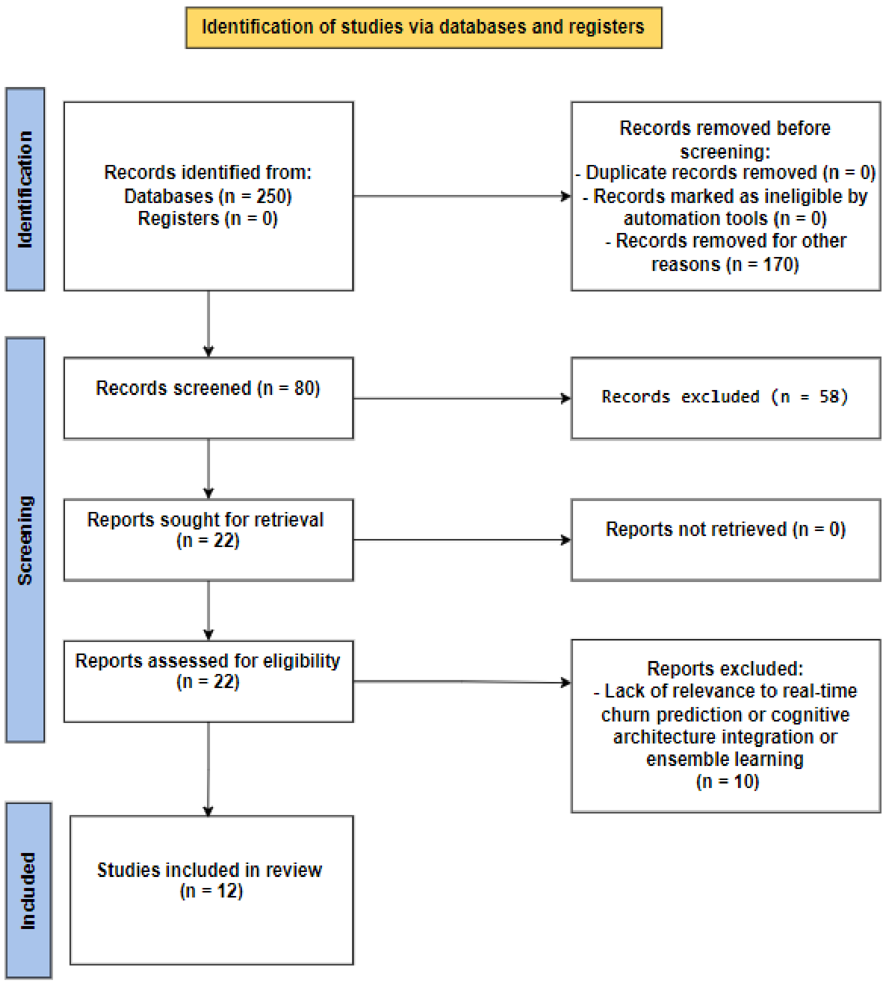
Identification of studies via databases and registers
Identification
Records identified from: Databases (n = 250) Registers (n = 0)
Records removed before screening: - Duplicate records removed (n = 0) - Records marked as ineligible by automation tools (n = 0) - Records removed for other reasons (n = 170)
Screening
Records screened (n = 80)
Records excluded (n = 58)
Reports sought for retrieval (n = 22)
Reports not retrieved (n = 0)
Reports assessed for eligibility (n = 22)
Reports excluded: - Lack of relevance to real-time churn prediction or cognitive architecture integration or ensemble learning (n = 10)
Included
Studies included in review (n = 12)

Figure 2.1: Flowchart depicting our systematic review process following PRISMA guidelines.

The selection process is depicted in a flow diagram (Figure 2.1), which clearly illustrates the systematic approach used to identify and select studies for this review. As a result, a total of 12 papers were chosen for in-depth analysis and discussion.

2.2 Customer Churn: Concepts and Importance

Customer churn, a critical concept in customer relationship management, refers to the process by which customers discontinue their relationship with a service provider. Lingling *et al.* [26] defined this as the process of customers switching from one service provider to another anonymously [27]. In the telecommunications and online service sectors, churn often occurs when customers migrate from one provider to another, particularly in saturated and highly competitive markets.

Impact of Churn on Businesses

The effect of customer churn on businesses, especially in service-oriented industries, is both significant and complex. Studies have consistently shown that retaining existing customers is significantly more cost-effective than acquiring new customers, with estimates ranging from four to six times more expensive [26, 28]. In addition, long-term customers contribute more to profitability and are less sensitive to competitive pressure, further emphasizing the need for effective churn management, especially in highly competitive markets. In highly competitive markets such as the Chinese mobile telecom sector, even a small improvement in churn prediction and prevention can lead to substantial increases in profit [26]. Celik *et al.* [29] also noted that the success of marketing strategies depends heavily on accurately identifying potential churners in advance, which can result in significant profit growth for companies.

Churn management's importance extends beyond specific industries. For instance, in the online gambling industry, customer retention is critical for a company's long-term viability [30]. This highlights the importance of churn prevention beyond specific industries, and is a universal concern in business-to-consumer contexts. Furthermore, Mustafa *et al.* [31] discussed several key factors influencing customer churn in the telecommunications industry, primarily through an analysis of the Net Promoter Score (NPS) dataset. This indicates that customers with low NPS are more likely to churn, suggesting that customer satisfaction is a critical determinant of retention. Interestingly, some studies indicate that traditional retention strategies such as attractive pricing plans can sometimes increase rather than decrease churn [30]. This contradictory finding underscores the complexity of the factors influencing customer churn, and the need for sophisticated prediction and prevention strategies.

2.3 Traditional Churn Prediction Methods

Classical Machine Learning Approaches. The literature reveals a wide array of machine learning approaches applied to the problem of churn prediction. These range from traditional classification algorithms to more advanced ensemble and deep learning methods.

Logistic Regression (LR) is cited as a common approach because of its interpretability and relatively low computational requirements [32]. Support Vector Machines (SVM) are also frequently used, likely because of their effectiveness in handling high-dimensional data [29].

Decision Trees and their ensemble counterpart, Random Forests (RF), are highlighted for their ability to handle non-linear relationships and high-dimensional feature space and Random Forests, in particular, are praised for their low bias and low variance, making them well-suited for the complex task of churn prediction [26].

More advanced techniques, such as Kernelized Extreme Learning Machine (KELM), have been proposed to categorize customer churn patterns, particularly in the telecom industry [28]. This approach combines the speed of extreme learning machines with the power of the kernel methods.

Ensemble methods, including bagging, boosting, stacking, and voting, have shown significant improvements in the prediction accuracy when applied to churn prediction tasks [33]. These methods leverage the strength of multiple base learners to create more robust and accurate predictions.

Deep Learning Approaches. Recently, deep learning methodologies have garnered significant momentum within the realm of customer churn prediction. Convolutional Neural Networks (CNNs), typically associated with image-processing tasks, have been adapted for churn prediction by transforming customer data into image-like representations [28]. This novel approach enables the leveraging of CNNs' powerful feature extraction capabilities of CNNs in the context of churn prediction.

Recurrent Neural Networks (RNNs), known for their proficiency with time-series data, have displayed capabilities in tasks related to churn prediction. A diverse array of RNN frameworks, including Long Short-Term Memory (LSTM) networks and Gated Recurrent Units (GRU), have been employed in churn prediction endeavors, yielding a range of success rates, as documented in the scholarly literature [30]. These specific architectures are proficient in identifying and modeling the temporal dependencies inherent in customer behavioral datasets.

Transformer-based architectures, which are then processed using a transformer model for classification, transform the input features into radar-chart images. This unique transformation of data aims to enhance the predictive accuracy, demonstrating its effectiveness in the telecommunications sector [34].

Although these machine learning frameworks are vital for predicting churn, they often struggle in rapidly changing, real-time settings because they rely on static datasets. This limitation highlights the need for more advanced methodologies capable of real-time analysis and intervention, thereby establishing the stage for the integration of real-time predictive models discussed in the following section.

2.4 Real-Time Churn Prediction

Real-time churn prediction (RTCP) is a critical area of study with practical applications across various industries including telecommunications, digital gaming, social platforms, and cloud service provision. Conventional methodologies for churn prediction frequently depend on batch processing techniques, which affect their capacity to deliver prompt intervention. Traditional churn prediction methods often rely on batch processing, which limits their ability to deliver timely intervention [35]. For instance, RICON, a sophisticated machine-learning system, was engineered to forecast customer churn probabilities in real-time by utilizing clickstream data, enabling comprehensive retention initiatives with timely product contextual assistance [17].

In the telecommunications sector, customer churn poses a significant challenge, leading to the adoption of data mining methodologies to identify potential churners and implement proactive measures [36]. Nevertheless, numerous existing models encounter latency issues and are unable to provide real-time insights, underscoring the need for real-time analytics in the telecommunications industry to enhance customer satisfaction and retention outcomes [35].

Real-time churn prediction is also essential in online gaming, as it helps operators identify potential churners and implement strategies to maintain engagement. Predictive features based on player in-game time have proven effective in this regard [17]. Similarly, in social networks, predicting user behavior is crucial for applications ranging from marketing to community management. A robust statistical model that uses graph attributes from a user's local community can accurately predict churn [6]. In cloud-based services, random forest-based models have shown high accuracy in predicting churn and guiding providers in recommending service adjustments to prevent customer loss [37].

In 2015, Castro *et al.* [38] examined feature representation methodologies for churn prediction in online gaming, highlighting sophisticated data transformation techniques but noting scalability limitations when applied to large datasets. The model struggled with computational efficiency, a recurring challenge in the field.

In 2017, Bertens *et al.* [39] introduced a survival ensemble model for churn prediction in mobile games, emphasizing robustness across different data distributions. While their approach demonstrated high predictive accuracy, it was resource-intensive and faced challenges in real-time adaptability. Similarly, Renjith *et al.* [40] developed a support vector machine-based model for e-commerce churn prediction, underscoring the importance of personalized retention actions. However, the model's dependency on static datasets limited its adaptability to dynamic customer behavior.

Lalwani *et al.* [41] proposed a machine learning system for churn prediction in telecommunications. Their methodology incorporated feature selection using the Gravitational Search Algorithm (GSA) and ensemble models like AdaBoost and XGBoost, achieving an AUC score of 84%. This approach demonstrated the importance of feature selection in improving prediction accuracy but was primarily designed for offline batch processing, limiting its real-time application.

In 2020, Tamuka *et al.* [42] presented a dynamic framework for churn prediction in online social platforms. By employing temporal graphs to model evolving user relationships, their approach highlighted the importance of capturing dynamic interactions in improving prediction performance. Random Forest achieved an F1 score of 87%, but computational costs associated with temporal graph generation posed scalability challenges.

In 2021, Almeida *et al.* [43] introduced a machine learning framework for churn prediction in cloud-based services. Their framework used SMOTE for addressing data imbalance and Recursive Feature Elimination (RFE) for feature selection. Gradient Boosting Machines (GBM) emerged as the top performer with an F1 score of 89% and precision of 91%. A key contribution of their work was the implementation of a distributed computing framework using Apache Spark to enhance scalability and computational efficiency for large-scale datasets.

In 2022, Arnab *et al.* [17] proposed RICON, a machine learning system for detecting high-risk customers in real time using clickstream data. While RICON showcased strong performance, its complex computational demands limited its scalability in larger datasets.

In 2023, Seo *et al.* [19] introduced a two-dimensional segment analysis model targeting high-risk churners. By combining segmentation with machine learning models such as Gradient Boosting Machines (GBM) and Random Forests, their approach achieved an AUC score of 91%. However, their segmentation-based method faced challenges in scalability, particularly with large datasets.

Table 2.1: Churn prediction studies reviewed in this dissertation.

Year	**Author(s)**	**Approach**	**Key Findings**	**Limitations**
2015	Castro et al. [38]	Feature representation	Effective for data transformation	Limited scalability
2017	Bertens et al. [39]	Survival ensemble	Robust across distributions	High computational demand
2017	Renjith et al. [40]	SVM-based	Emphasized personalized actions	Static dataset dependency
2022	Arnab et al. [17]	RICON	Strong performance with clickstream data	Scalability issues
2023	Seo et al. [19]	2D Segment Analysis	Improved churn rate reduction	Computational efficiency limitations

Despite these advancements, the implementation of real-time churn prediction remains a challenge. A primary difficulty lies in the requirement for timely and reliable data processing. Real-time systems must handle vast amounts of data rapidly, necessitating substantial computational resources and sophisticated algorithms to process and analyze data streams efficiently [44]. Furthermore, the dynamic nature of customer behavior presents an additional challenge, demanding models that can adapt to emerging patterns without significant delay [45].

2.5 Cognitive Architectures: An Overview

Cognitive architecture refers to a domain-generic computational model designed to simulate human cognition and behavior across various contexts and tasks [46, 47]. Unlike traditional machine learning methods that focus on pattern recognition and predictive tasks using statistical and computational techniques [47, 23], cognitive architectures aim to emulate the underlying mechanisms of human cognition. These architectures go beyond input-output mappings by modeling processes such as reasoning, learning, and memory retrieval [46].

Popular cognitive architectures such as Adaptive Control of Thought-Rational (ACT-R), Soar, and CLARION have been extensively applied to simulate a wide range of cognitive tasks, demonstrating their versatility and depth [47]. For instance, while machine learning models like neural networks excel in pattern recognition, cognitive architectures incorporate explicit rules and symbolic reasoning to replicate human-like decision-making processes [48].

Cognitive architectures have shown potential in managing complex, dynamic environments. In 2016, Paraense *et al.* [21]. Grassioto et al. [22] introduced a distributed cognitive architecture for urban traffic control, demonstrating the value of Global Workspace Theory for managing real-time data flow and decision-making. While the architecture was specifically applied to traffic control scenarios, it showcased notable scalability and flexibility. However, concluding that CST has inherent scalability problems based solely on this application is not justified. It is important to recognize that CST has been demonstrated in other domains, including transportation robotics, highlighting its broader applicability.

In 2016, Paraense *et al.* [21] presented a highly scalable implementation of CST, which demonstrated significant potential for applications beyond traffic control. The study highlighted the system's ability to adapt to various real-world scenarios, providing a foundation for its use in diverse domains. Including [21] offers a broader view of CST's scalability and versatility, ensuring a balanced assessment of its potential.

In 2017, Gudwin *et al.* [49] presented the Multipurpose Enhanced Cognitive Architecture (MECA), which combines elements of Dual Process Theory and Grounded Cognition. MECA provided a versatile framework applicable to both traffic control and transportation robotics. However, its evaluation lacked comprehensiveness, particularly in exploring its adaptability across diverse, real-world scenarios beyond these domains.

In 2022, Grassioto et al. [22] introduced CogToM, designed to mimic human cognition for autism spectrum interactions. Although effective in social interaction contexts, CogToM faced challenges related to real-time adaptability and scalability, similar to other cognitive architectures.

In 2016, Vallverdu et al. [23] merged the Lovheim Cube of Emotion with the Von Neumann architecture to enhance emotional and computational integration in AI systems. This work provided insights for human-like decision-making in AI, but issues related to scalability and real-time implementation persisted.

The existing studies in cognitive architecture highlight their capability for dynamic decision-making, making them a promising tool for real-time churn prediction. However, the lack of research integrating cognitive architectures with ensemble learning in real-time contexts represents a critical research gap.

Table 2.2: Cognitive architecture studies reviewed in this dissertation.

Year	Author(s)	Model	Key Findings	Limitations
2016	Paraense et al. [21]	Distributed architecture	Improved decision-making, high scalability	Application bias to traffic control
2016	Paraense et al. [20]	Scalable CST implementation	Adaptable to diverse real-world scenarios, scalable design	Lack of comprehensive evaluation
2017	Gudwin et al. [49]	MECA	Versatile cognitive modeling	Study focused on specific use cases, with potential for exploration in additional application domains
2022	Grassioto et al. [22]	CogToM	Effective in social interactions	Social interaction scenarios focused
2016	Vallverdú et al. [23]	Lovheim Cube + Von Neumann	Enhanced emotional integration	Scalability challenges

ACT-R (Adaptive Control of Thought-Rational)

ACT-R, developed by Anderson et al., has evolved through several iterations, with its latest version, ACT-R 7.21.6, which focuses on modular organization. The architecture's modular approach, incorporating sensory, motor, intentional, and declarative modules, reflects a commitment to domain-specific processing, aligning with theories of brain function specialization. A key strength of ACT-R lies in its integration of symbolic and sub-symbolic processes. The use of base activation for declarative chunks and utility calculations for production allows the model to simulate human memory retrieval and decision-making processes with remarkable fidelity. However, this hybrid approach also introduces complexity in parameter tuning, which potentially limits the generalisability of the architecture across diverse cognitive tasks [50].

Soar

Soar, with its long developmental history, presented a unified theory of cognition based on problem solving and learning. Its problem-space computational model provides a flexible framework for representing and solving a wide range of cognitive tasks. The architecture's impasse mechanism for handling insufficient knowledge is noteworthy. This feature enables dynamic problem-solving and learning, allowing Soar to adapt to novel situations. However, reliance on impasse-driven learning may lead to brittleness in complex real-world scenarios where clear impasses are not always identifiable [51].

ICARUS

ICARUS distinguishes itself from a clear separation of conceptual and procedural knowledge. This distinction aligns with psychological theories of declarative and procedural memory, potentially offering a more cognitively plausible model of knowledge representation. The hierarchical organization of concepts and skills in ICARUS provides a structured approach to knowledge representation that facilitates reasoning at multiple levels of abstraction. This feature is particularly advantageous for modeling complex cognitive tasks that require both high-level planning and low-level execution [52].

Cognitive System Toolkit (CST)

The Cognitive System Toolkit (CST), as defined by Paraense *et al.* [20] is a framework designed to facilitate the development of cognitive systems that are capable of emulating human-like cognitive functions. The CST provides a modular and flexible environment that supports the creation of complex cognitive architectures by allowing the integration of various cognitive models and components. One of CST's primary goals is to enable researchers and developers to construct cognitive systems that can perform a wide range of tasks, from perception and reasoning to learning and decision-making [53].

CST modularity is the cornerstone, allowing for a fluid combination of distinct cognitive models and elements. This modular framework is crucial for creating complex cognitive systems because it enables the integration of diverse specialized components that can collaborate to achieve a common objective. CST also emphasizes interoperability and ensures effective communication and information-sharing between different modules [54].

Furthermore, CST supports various cognitive architectures and provides a blueprint for the development of cognitive systems. This support allows researchers to experiment with different approaches to cognitive system design, foster innovation, and explore new ideas [53]. In summary, CST is a comprehensive platform that aids in constructing cognitive systems and promoting innovation and experimentation in the field of cognitive system development [54].

Key components in CST and their roles. The key components of CST include the following as defined by Paraense *et al.* [20].

Codelet: Codelets are microagents that represent the fundamental processing units within the Cognitive Systems Toolkit (CST) architecture. They perform specific tasks independently and are designed to operate continuously and cyclically. In the figure, the codelets are shown as rectangular blocks, illustrating how they interact with memory objects and the coderack. Each codelet has input local input and global input (LI and GI) connections that interact with memory objects, allowing it to access the required information and store the new information generated during its execution. The *proc()* label on the codelet represents its processing function that encapsulates the specific task it performs within the architecture.

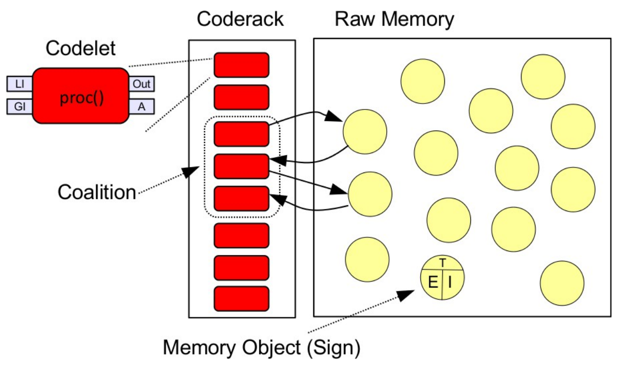

Figure 2.2: The core of the CST architecture. Codelets (rectangular blocks) process information independently and interact with Memory Objects (yellow circles) through input/output connections (LI, GI, and Out). The Raw Memory stores Memory Objects which contain Information (I), Time Stamp (T), and Evaluation (E) properties. Codelets are organized into Coalitions for coordinated processing, managed by the Coderack scheduling system.

Coderack: The coderack is a scheduling and coordination component that manages the execution of codelets. It holds a "rack" of codelets, selecting and activating them based on their importance, task order, and priority. The coderack groups codelets into coalitions, in which several codelets collaborate to perform complex cognitive functions. Coalition formation ensures that codelets work together in a coordinated manner, effectively implementing the cognitive processes of the system.

Coalition: Coalitions are groups of codelets that work together to achieve specific cognitive tasks. Within the CST core, coalitions are formed to enable more structured and coordinated processing among codelets. A coalition structure is necessary because certain cognitive functions require interdependence among codelets, where outputs from one codelet may serve as inputs for others.

Memory Object: Memory objects are the primary information holders in the CST architecture, and are represented as circular yellow elements in the figure. They store the information required by the codelets to perform their tasks and retain the results produced by the codelets. Each memory object has three key properties:

- **Information (I):** Represents the main content of the memory object, which can range from simple data to complex structures.
- **Time Stamp (T):** Indicates the last update time of the memory object, ensuring that codelets operate on the most recent data.
- **Evaluation (E):** Serves as a meta-information marker that is used for various purposes, such as assessing the relevance or accuracy of the stored information.

Raw Memory: Raw memory acts as a storage area for memory objects, maintaining a pool of generic information that can be accessed and modified by the codelets. This storage allows codelets to share information across different processing stages, thereby supporting parallel and asynchronous operations in the CST core. The arrows between the codelets and raw memory signify the flow of information, illustrating how codelets retrieve and update data within memory objects to complete their cognitive function.

Compared to static environments like Python-based implementations, the cognitive architecture approach offers several advantages:

- **Dynamic Memory Management:** Memory objects provide a structured approach to handling evolving data streams, unlike traditional static data structures.
- **Adaptive Processing:** Codelets can dynamically adjust their processing based on real-time performance metrics, a feature not inherent in standard ensemble implementations.
- **Parallel Processing:** The coalition structure enables efficient parallel processing of multiple models while maintaining coordination through the coderack system.

Summary: Figure 2.2 shows the core components of the CST architecture, emphasizing how codelets, coderacks, coalitions, and memory objects interact to form a parallel asynchronous cognitive processing system.

2.6 Ensemble Learning

Ensemble learning is a machine learning strategy that integrates various models to improve predictive success, stability, and generalization compared to standalone models. This method employs multiple models to mitigate errors and increase accuracy [55]. As highlighted by Kalita *et al.* [55], ensemble learning methods, including bagging, boosting, and stacking, aim to reduce the limitations of singular models by merging their predictions. For instance, bagging consists of training numerous models on distinct data subsets and then averaging their outcomes, which contributes to decreasing the variance and reducing overfitting. However, boosting focuses on sequential training models, in which each new model attempts to correct the errors of its predecessor, thereby improving the model's bias and accuracy.

Stacking, another popular ensemble learning method, involves training multiple base models (often diverse learners) and then combining their predictions using a meta-model. The meta-model is trained on the outputs of the base models, effectively learning how to best aggregate their predictions for improved accuracy. Stacking differs from bagging and boosting by utilizing a hierarchical structure, where the meta-model is trained separately to make decisions based on the collective outputs of the base models. This method has been shown to work particularly well in heterogeneous ensembles by leveraging the unique strengths of diverse models to enhance overall performance [55].

Sharma *et al.* [56] highlighted the application of ensemble learning in various domains, emphasizing its effectiveness in complex problem-solving scenarios, noting that ensemble methods are particularly beneficial for handling high-dimensional data and improving the stability of predictions in dynamic environments. This adaptability makes ensemble learning the preferred choice in fields such as finance, healthcare, and image processing, where data complexity and variability are significant challenges.

Huang *et al.* [57] highlighted the essential theories behind ensemble learning, emphasizing that the success of these strategies is significantly reliant on the uniqueness and independence of the contributing models. They argue that the more diverse the models, the better the ensemble can capture different aspects of the data, leading to improved performance. However, they also highlight that attaining the suitable balance between diversity and accuracy is key, as an excess of diversity could lead to higher computational expenses and complexities without substantial performance improvements. Gurram *et al.* [58] explored the limitations and challenges associated with ensemble learning, such as increased computational requirements and complexity of model selection and tuning. They suggested that while ensemble methods can significantly enhance performance, they require careful design and implementation to avoid pitfalls such as overfitting and model redundancy. In summary, ensemble learning represents a strong strategy that boosts model effectiveness through the integration of various models, utilizing their advantages and addressing their limitations. Its effectiveness across various domains is well documented, although it requires careful consideration of model diversity, computational costs, and implementation strategies to maximize its benefits [55, 56, 57, 58].

Ensemble learning methods can be effectively integrated into a cognitive architecture framework to enhance the predictive accuracy and adaptability. By combining the strengths of multiple models within the cognitive architecture, the system can leverage diverse algorithms to address the prediction tasks. This integration allows the cognitive architecture to adapt to varying data shifts and maintain robust performance in dynamic environments, which is crucial for real-time churn prediction.

In 2024, Shaikhsurab *et al.* [24] introduced a novel adaptive ensemble framework for customer churn prediction in telecommunications, achieving a 99.28% accuracy by combining models like XGBoost, LightGBM, LSTM, MLP, and SVM through stacking. It supports proactive retention strategies but has limitations, such as high computational demands, potential performance variations on non-telecom datasets, and an emphasis on accuracy over other crucial metrics like precision, recall, or F1-score.

In 2021, Wang *et al.* [18] proposed a novel VANet with cross-view label smoothing regularization (CVLSR) and cross-view distance metric introduced to enhance vehicle re-identification (Re-ID) performance across multiple viewpoints. It outperformed state-of-the-art methods in feature extraction, achieving high rank-1 accuracy and mean average precision (mAP) on VeRi-776 and VehicleID datasets by capturing both low-level and high-level semantics. However, potential limitations include the need for handling diverse environmental conditions and reducing computational complexity for real-time applications.

In 2017, Krawczyka *et al.* [25] provided a comprehensive survey of ensemble approaches for data stream analysis, addressing both classification and regression tasks. The study introduced a taxonomy of ensemble algorithms for data stream mining and explored concepts like imbalanced data streams and novelty detection, emphasizing the need to consider context, such as seasonal effects in electricity data. However, it noted the need for formal definitions of gradual drift and more research on handling incomplete data in streaming contexts. The study also identified low diversity in ensemble methods, suggesting further research is required to improve drift monitoring and handling.

In summary, ensemble learning is a significant approach for improving the predictive analysis efficiency through a combination of multiple models. The synergy between ensemble methods and cognitive architectures forms a core component of the proposed framework for real-time churn prediction, providing a robust solution to the challenges posed by dynamic data environments. Ensembles are often grouped into two types: heterogeneous and homogeneous. An ensemble is homogeneous if only a single type of learner is used [59]. This study focuses on heterogeneous ensembles by combining different single classifiers to construct an ensemble.

Heterogeneous Ensembles

The predictions produced by different classifier models employed in an ensemble must be compiled into one ultimate result. Various techniques have been developed to accomplish this. With voting, every classifier independently chooses a class for each instance, and then a voting scheme is used to determine the class of each instance [60]. The ensemble has three methods for assigning a class: (i) unanimous voting (when every classifier reaches consensus), (ii) simple majority voting (when at least one supports it more than half), or (iii) plurality voting or majority voting (MV) (when a class receives the greatest number of votes) [61]. Majority voting in a classification task refers to the use of the mode as a prediction of the ensemble. The main drawback of majority voting is the assumption that classifiers are equally reliable.

When certain classifiers are believed to have a greater impact than others, employing weighted majority voting (WV) may enhance performance, in contrast to standard majority voting. Here, the decisions of more qualified classifiers are assigned higher weights that are proportional to the estimated future performance of the classifiers. This estimation can be based on the classifier's outcomes from a separate validation or training dataset. The performance depends on the reliability of the estimated accuracies of the classifiers [61]. The voting variants mentioned above are also known as hard voting. In the case of soft voting (SV), the ultimate classification decision is derived from the average churn probabilities indicated by each classifier. Therefore, this method uses only classifiers that can calculate the probability of churning [62].

Stacking ensemble learning, or stacked generalization, is a technique that combines multiple classifiers. It was introduced by [63] and is very useful in the field of classification, particularly when classifiers tend to be accurate only in specific areas of the dataset [64, 65]. Stacking consists of two phases. First, the base or single learners (Level 0) were trained. The outputs of these models are collected in a new dataset, together with the real value that it is supposed to predict. This new dataset is used as input for the stacking model learner (level 1), also known as the meta-learner, which combines them and provides the final output. the classification problem, which is also referred to as the meta-classifier [61, 59]. Along with the selections that are necessary for algorithms and parameters, numerous subtle modifications can be introduced to the ensemble. These options result in a very versatile ensemble, but they make analyses and comparisons more difficult because there is considerable variation in the application of the basic idea [66].

Dynamic Weighting in Ensemble Model

Dynamic weighting is an elaborate method in ensemble models designed to refine the predictive accuracy of ensemble learning by adjusting the weights assigned to different models based on their performance rates or additional criteria. This method differs from static weighting because every model in the ensemble receives a constant weight, irrespective of its effectiveness. The work by [67] shows the integration of adaptive weighting in climate modeling by employing ensemble frameworks to improve the accuracy of climate forecasts.

This study underscores the necessity of dynamically modifying weights according to the temporal and spatial effectiveness of individual models, potentially improving the accuracy and dependability of climate forecasts. This technique facilitates collective adaptation to changing situations and refines the ability to predict over time. Catto *et al.* [68] explored dynamic weighting in the context of machine learning, particularly in reinforcement learning environments. They proposed a strategy that adjusts the weight of ensemble models according to their recent results, which assists in addressing nonstationary environments where the core data distribution can fluctuate over time.

This dynamic adjustment is crucial for maintaining the performance of the ensemble and ensuring that it remains robust to changes in the data [68]. Zhang *et al.* [69] discussed the application of dynamic weighting in financial forecasting models. They illustrated that by adaptively modifying the weights of ensemble models according to their historical performance, the ensemble can more effectively capture market dynamics and enhance forecasting precision. This approach is particularly beneficial in financial markets where conditions can change rapidly and unpredictably. Chen *et al.* [70] further supported the use of dynamic weighting in ensemble models by showing its effectiveness in improving model generalization and reducing overfitting.

As the methodology is outlined in detail, it is crucial to visualize the core structure of the Cognitive Systems Toolkit (CST) architecture that we employ. Figure 2.2 shows a schematic of this architecture. As shown in Figure 2.2, CST architecture is composed of several interconnected components that process sensory data. The core components of the CST architecture include codelets and memory objects. This architecture is constructed to reflect human cognitive mechanisms, with all elements carrying out specific roles similar to various cognitive operations in the human brain.

2.7 Findings and Identification of Research Gaps

Despite significant advancements in churn prediction models, several critical challenges persist. Current real-time churn prediction models often face issues with computational efficiency. Processing continuous, large-scale data streams in real-time demands substantial computational resources, leading to delays that can hinder timely interventions [25]. temporal change presents another major obstacle. As models are required to adapt to ever-increasing datasets, maintaining high accuracy while expanding their scope becomes increasingly challenging. Additionally, adaptability remains a pressing concern. Many existing models lack the flexibility to adjust to rapidly shifting customer behavior without a significant drop in performance [18]. These challenges underscore the need for more robust and scalable frameworks for real-time churn prediction.

Our literature review highlights several key insights across the domains of churn prediction, cognitive architectures, and ensemble learning. Traditional churn prediction models, though effective in static settings, often fail to meet the demands of real-time adaptability [71]. These models typically rely on static datasets, which limits their ability to react dynamically to customer behavior changes in real-time environments. Cognitive architectures, in contrast, excel in handling complex, dynamic environments by simulating human-like decision-making processes. By leveraging stored experiences as instances, these architectures enable decision-making that adapts to new patterns, mimicking the cognitive flexibility of human behavior [72].

Furthermore, ensemble learning methods have been shown to outperform individual models by combining their strengths. Extensive research has demonstrated that ensemble techniques, such as bagging and boosting, effectively reduce variance and bias, leading to more robust and accurate predictions compared to single models [73]. However, a significant gap in the literature remains: the integration of cognitive architectures and ensemble learning for real-time churn prediction is largely unexplored. While cognitive architectures offer dynamic decision-making capabilities and ensemble learning improves predictive accuracy, the synergy between these two approaches in a real-time context has yet to be fully investigated. This gap represents a crucial opportunity to advance churn prediction methodologies by combining the adaptability of cognitive systems with the accuracy of ensemble learning.

2.8 Positioning and Discussion

The literature review has provided an overview of the current state of research in real-time churn prediction and cognitive architectures. It has highlighted the strengths and weaknesses of existing methodologies and the potential benefits of integrating cognitive architectures into churn prediction models. The use of cognitive architectures in churn prediction is a relatively unexplored area, offering significant innovation potential. With their ability to mimic complex decision-making processes, cognitive architectures could provide a more nuanced and accurate approach to predicting customer churn. However, implementing cognitive architectures in churn prediction models presents its own challenges; these include the need for extensive data preprocessing, the complexity of implementing self-learning models, and the difficulty of evaluating such models.

Cognitive architectures offer a potentially promising solution to these challenges, providing flexible and modular frameworks that adapt to changing data streams in real time. However, integrating cognitive architectures with ensemble learning for churn prediction remains an area that has yet to be fully explored, particularly in real-time contexts.

Our research aims to address these challenges by developing a churn prediction model based on the Cognitive Systems Toolkit (CST). By leveraging the strengths of cognitive architectures and addressing their limitations, our research seeks to advance the field of churn prediction. We aim to demonstrate our model's practical applicability and effectiveness through real-world case studies, thereby contributing to the ongoing efforts to improve customer retention strategies in different industries. Our research is positioned at the intersection of churn prediction and cognitive architectures, offering a novel approach to a complex and critical problem. Through our study, we aim to enhance the accuracy of churn prediction models and contribute to the broader understanding of how cognitive architectures can be applied to real-world problems.

This M.Sc. dissertation proposes an innovative theoretical framework integrating cognitive architectures with ensemble learning for real-time churn prediction. This setup leverages the modular traits of cognitive architecture to create a flexible real-time prediction system. Within this cognitive structure, we propose implementing an ensemble of machine learning models, each specialized for different aspects of churn prediction. The cognitive architecture would serve as the comprehensive system, managing the flow of information, and decision-making processes. The ensemble learning component would be integrated within this structure, providing robust predictions based on multiple models. This combination could potentially offer several advantages over the traditional ensemble learning approaches.

- Real-time adaptability: The cognitive architecture's ability to handle dynamic environments could enable the system to adapt to changing patterns in customer data in real time;
- Improved accuracy: The ensemble learning component could provide more accurate predictions by leveraging the strengths of multiple models;
- Explainability: The modular structure nature of cognitive architectures could offer improved explainability of the prediction process;
- Scalability: The modular nature of both cognitive architectures and ensemble methods could allow for easy scalability and incorporation of new models or data sources.

2.9 Final Remarks

This chapter explored the key foundational concepts crucial to understanding the scope of this study, including customer churn, traditional and real-time churn prediction methods, cognitive architectures, and ensemble learning techniques and the works in the literature. Each element forms the basis for developing an adaptive real-time customer churn prediction model using cognitive architecture. While these foundational concepts lay the groundwork, significant gaps exist in current research, particularly in the application of cognitive architectures and ensemble methods to real-time churn prediction.

The related work analysis presented in this chapter has provided an overview of current advancements in real-time churn prediction, cognitive architectures, and ensemble learning. We discussed the critical role that churn prediction plays in industries where dynamic customer behavior necessitates real-time interventions. Despite the advancements of traditional predictive methods, they often struggle with challenges such as real-time adaptability, computational efficiency, and scalability.

The following chapter explores existing and related studies that address customer churn prediction. It examines where current methods fall short, and identifies opportunities for the application of cognitive architectures in this domain. This review highlights the rationale for our approach and positions this study within a broader context of the field.

Chapter 3

Methodology

This chapter outlines the research methodology employed to achieve the study's aims and objectives. It details the study's design, data collection and preprocessing processes, and computational model implementation. The chosen methodology is driven by the need for precision, replicability, and adaptability within the cognitive architecture framework, ensuring robust and generalizable outcomes.

The research follows a quantitative approach, integrating machine learning models within a cognitive architecture to develop a dynamic weighted ensemble of classifiers. These classifiers adapt to data over time, enhancing their predictive accuracy. The methods employed include data collection from external sources, data preprocessing to prepare the dataset for analysis, and the use of various computational models to evaluate performance.

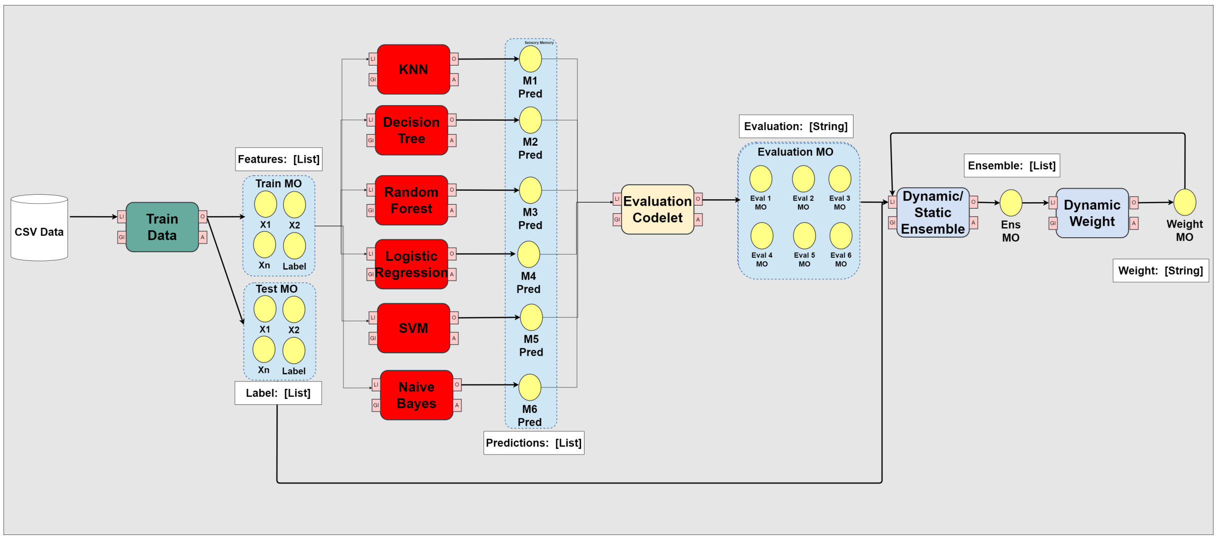

Figure 3.1: The proposed cognitive architecture framework for real-time churn prediction.

Figure 3.1 presents the cognitive architecture framework for real-time data processing and analysis. The system architecture integrates memory objects for efficient data storage with specialized codelets for processing tasks. This design facilitates continuous data processing while enabling dynamic adjustment of model weights based on performance metrics, ensuring adaptive response to changing patterns in customer behavior.

The methodology implements a systematic workflow comprising four interconnected phases. The initial phase involves data ingestion and preprocessing through the Data Reader Codelet, which prepares incoming data for analysis. The second phase encompasses model training and evaluation, where individual codelets process the prepared data using specific machine-learning algorithms. The third phase involves dynamic weight adjustment, where the system evaluates and updates model weights based on continuous performance assessment. The final phase consists of ensemble prediction aggregation, where weighted predictions from all models are combined to generate the final output.

3.1 Cognitive Architecture Implementation

The cognitive architecture framework in this research is designed specifically for real-time churn prediction and has been validated within the context of telecommunication datasets. This approach enables adaptive learning through a modular and distributed architecture, facilitating efficient data handling, real-time adaptability, and dynamic integration of individual model predictions. Figure 3.1 provides a comprehensive view of the architecture, illustrating its modular components and their interactions.

Data Input and Preprocessing: The system begins by reading raw CSV data, including training and testing datasets derived from the telecommunication industry. This raw data undergoes ingestion and preprocessing to ensure consistency and reliability before entering the model pipeline, making the architecture applicable for real-time data processing scenarios.

Memory Objects

Memory objects form the foundation of the system's data management, providing a standardized structure accessible to all components in the cognitive architecture. They serve three critical roles:

- **Data Storage and Access**: Memory objects manage the storage and retrieval of input features and labels, ensuring that consistent and clean data is accessible to all model codelets within the system.
- **Model States and Predictions**: They store the predictions and current states of each model, enabling efficient monitoring and tracking of each component's performance.
- **Dynamic Weight Storage**: These memory objects maintain dynamically adjusted weight values for the ensemble model, facilitating adaptive model combinations based on real-time performance metrics.

Train and Test Data Memory Objects: The train and test data memory objects store processed features and labels from the training and testing datasets. These memory objects ensure standardized data input across all models, enhancing consistency and reliability within the framework.

Codelet Implementation

The architecture includes several specialized codelets, each responsible for different aspects of the churn prediction process:

- **Data Reader Codelet**: This codelet is responsible for data ingestion and preprocessing. It prepares raw data to ensure standardized input for model training and testing. This preprocessing step ensures data consistency, improving model accuracy and performance.
- **Model Codelets**: Six independent model codelets are implemented in this study, specifically designed for this study to execute distinct machine learning algorithms. These codelets were configured to leverage the following algorithms, ensuring modularity and flexibility:
  - **K-Nearest Neighbors**: Utilizes instance-based learning.
  - **Logistic Regression**: Manages probability-based classification.
  - **Decision Tree**: Employs rule-based predictions to capture interpretable data splits.
  - **Random Forest**: Extends Decision Tree capabilities with an ensemble approach to improve robustness.
  - **Support Vector Machine**: Uses margin-based classification for optimized decision boundaries.
  - **Naive Bayes**: Implements probability-based classification tasks, providing a quick and effective prediction model.

  Each model codelet independently processes data and outputs predictions stored in shared memory objects for further evaluation and ensemble aggregation. This independent processing allows each model to contribute unique insights based on its algorithmic strengths.
- **Evaluation Codelet**: This codelet evaluates the predictions from each individual model based on performance metrics, such as accuracy, precision, recall, and F1-score. These evaluations inform the ensemble model's weight adjustments, as the performance of each model influences its contribution to the final prediction.
- **Ensemble Memory**: This memory object stores evaluation results from each model, acting as a centralized data storage for dynamic weight adjustments. This allows the ensemble model to access the latest performance evaluations and apply real-time adjustments, improving adaptability.
- **Dynamic/Static Ensemble Integration**:
  - **Dynamic Ensemble**: Adjusts model weights in real-time, based on the latest performance evaluations, providing adaptive responses to evolving data patterns.

  - **Static Ensemble**: Uses fixed weights for model integration, serving as a stable baseline for comparison. By combining individual model outputs according to their performance-based weights, the dynamic ensemble demonstrates a superior capacity to respond to shifting data distributions compared to the static approach.

- **Weight Memory Object**: This memory object maintains the dynamically adjusted weights for each model. By keeping track of recent performance, the architecture can dynamically allocate greater influence to models with higher predictive accuracy, ensuring that the ensemble continuously optimizes its predictions.

Dynamic Weight Manager Codelet

The **Dynamic Weight Manager Codelet** plays a crucial role in the adaptive capabilities of the cognitive architecture. It maintains a sliding window of recent instances to monitor model performance, applying specific learning and forgetting rates to adjust weights in response to changing data patterns. This codelet ensures that the system remains sensitive to recent trends, making necessary weight adjustments that allow the ensemble to adapt effectively over time.

3.2 Workflow of the Proposed Methodology

The research methodology follows a structured workflow, integrating the various components of the cognitive architecture into a cohesive system for real-time churn prediction. This workflow leverages cognitive architecture principles to enable dynamic ensemble learning, ensuring adaptability and scalability in processing and decision-making. The following workflow outlines the steps involved in data collection, preprocessing, model implementation, and evaluation stages. Each step is interconnected to ensure the cognitive architecture dynamically adapts to the data and produces accurate predictions.

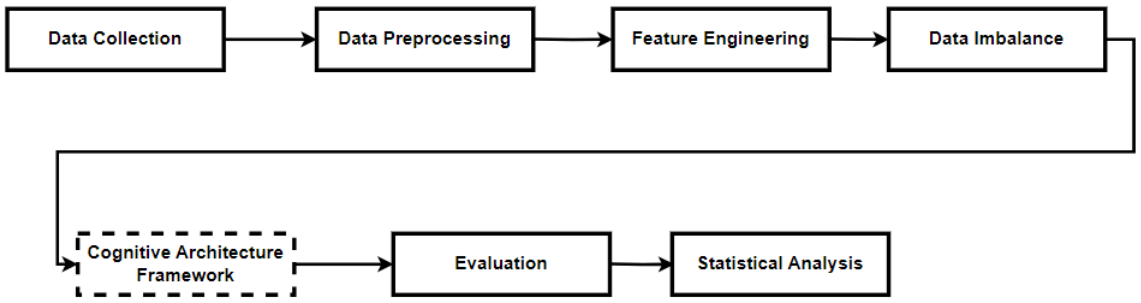

Figure 3.2: Workflow of the proposed cognitive architecture methodology.

Figure 3.2 highlights the end-to-end process, starting from data collection through preprocessing, model training, dynamic ensemble evaluation, and final comparative analysis. The key phases are as follows:

- **Data Collection**: Gathering external data sources for customer churn prediction.
- **Data Preprocessing**: Preparing and transforming the raw data for the models, including handling missing values, normalizing data, and encoding categorical variables.
- **Feature Engineering and Data Imbalance Management**: Creating relevant features and addressing class imbalance issues (SMOTENN technique).
- **Cognitive Architecture Framework + Machine Learning Models**: Implementing the cognitive architecture framework that includes six machine learning models KNN, Logistic Regression, Decision Tree, Random Forest, SVM, and Naive Bayes. A dynamic weight adjustment system, developed within this study, drives the integration, enabling the framework to optimize predictions based on the performance of individual models. This system ensures adaptability to evolving data patterns, improving prediction reliability
- **Dynamic Ensemble Model**: Aggregating predictions from individual models and continuously adjusting model weights dynamically based on recent performance metrics.
- **Evaluation**: Assessing model performance through accuracy, precision, recall, and F1-Score metrics, comparing static and dynamic ensemble approaches.
- **Statistical Analysis and Comparative Evaluation**: Conducting statistical analysis and comparing the dynamic ensemble with other approaches to assess the cognitive architecture's effectiveness.

This workflow emphasizes the proposed cognitive architecture's adaptability and scalability, ensuring that the system remains responsive to changing data conditions.

3.3 Data Processing and Analysis

This section presents the comprehensive data handling framework implemented in this study, encompassing data collection, preprocessing, and exploratory analysis. The framework ensures systematic data preparation and validation for the cognitive architecture's machine learning components.

Dataset Description and Collection

The main dataset utilized for this study is the **Churn-in-Telecom Dataset**, a well-established benchmark specifically designed for customer churn prediction in the telecommunications industry [74]. This dataset was selected for three primary reasons:

1. **Comprehensive Coverage of Customer Behavior Patterns:** The dataset provides detailed insights into temporal factors such as account duration, interaction metrics like customer service calls, and usage patterns including data consumption and call volumes.

2. **Established Use in Churn Prediction Research:** Its frequent application in related research ensures compatibility and comparability with existing works in the field.

3. **Relevance to Churn Prediction Studies:** Although the dataset is static, its diverse feature set and complexity enable robust evaluation of the proposed cognitive architecture's ability to adapt to simulated real-time prediction scenarios.

The dataset contains a total of **3,000 customer records**, each described by **11 attributes**. These attributes include customer-related features and service usage metrics, making it suitable for analyzing diverse aspects of churn behavior. The problem addressed is a **binary classification task**, with the following class distribution:

- **Churn Customers (Label: 1):** 400 instances
- **Non-Churn Customers (Label: 0):** 2,600 instances

This class distribution highlights an imbalance, which is typical in churn prediction problems and adds complexity to the evaluation of machine learning models. By leveraging this dataset, the study aims to test the efficiency of the proposed cognitive architecture in accurately predicting churn while addressing class imbalance and adaptability to dynamic prediction requirements in simulated environments.

3.3.1 Dataset Structure and Features

The Churn-in-Telecom Dataset structure reflects the complexity of customer behavior in telecommunications services. Table 3.1 presents the primary features utilized in this study. The target variable, Churn, indicates customer attrition through a binary classification where 1 represents service cancellation and 0 represents retention. The remaining features capture various dimensions of customer engagement and service utilization.

Index	**Feature Name**	**Description**
1	Churn (Target Label)	1 if customer cancelled service, 0 if not
2	Account Weeks	Number of weeks customer has had active account
3	Contract Renewal	1 if customer recently renewed contract, 0 if not
4	Data Plan	1 if customer has data plan, 0 if not
5	Data Usage	Gigabytes of monthly data usage
6	Cust Serv Calls	Number of calls into customer service
7	Day Mins	Average daytime minutes per month
8	Day Calls	Average number of daytime calls
9	Monthly Charge	Average monthly bill
10	Overage Fee	Largest overage fee in the last 12 months
11	Roam Mins	Average number of roaming minutes

Table 3.1: Description of features in the Churn-in-Telecom Dataset. The features represent various aspects of customer behavior and service usage patterns.

The features encompass three main categories. Service usage metrics include DayMins, DayCalls, and DataUsage, providing quantitative measures of customer engagement. Customer interaction indicators comprise CustServCalls and ContractRenewal, reflecting the quality of customer relationships. Financial metrics such as MonthlyCharge and OverageFee capture the economic aspects of customer behavior. This comprehensive feature set enables detailed analysis of churn patterns while providing sufficient complexity to evaluate the adaptive capabilities of the cognitive architecture.

3.3.2 Data Preprocessing

The preprocessing implements a systematic approach to prepare the dataset for the cognitive architecture, ensuring data quality and consistency across all processing stages. This framework addresses three critical aspects of data preparation: data cleaning, feature scaling, and class balance adjustment.

Data Cleaning and Validation

The initial preprocessing phase focused on ensuring data integrity through systematic cleaning procedures. Missing value analysis revealed minimal data gaps addressed through statistical imputation methods appropriate to each feature's distribution. Duplicate record identification and removal prevented potential bias in model training. This cleaning process established a reliable foundation for subsequent analysis while maintaining the dataset's statistical properties.

Feature Scaling Implementation

Feature scaling formed a critical component of the preprocessing pipeline, particularly for optimizing model performance. The implementation adopted standardization as the primary scaling method, transforming numerical features to a standard normal distribution with zero mean and unit variance. This standardization approach notably enhanced the performance of gradient-based algorithms within the cognitive architecture, including the Logistic Regression and Support Vector Machine components.

The standardization process followed a protocol to prevent data leakage. Scaling parameters were computed from training data and applied to validation and test sets. This approach maintained the integrity of the evaluation process by ensuring that test data characteristics did not influence the scaling parameters. The standardization transformation is defined as:

$$z = \frac{x - \mu}{\sigma}, \tag{3.1}$$

where x represents the original feature value, μ is the mean, and σ is the standard deviation of the training data for each feature.

Feature Engineering and Encoding

The feature engineering process implemented specific transformations for different dataset types. Categorical features underwent binary encoding for two-level factors, converting them to numerical representations suitable for model processing. The encoding scheme mapped the churn status to binary values, with 0 representing Not-churn and 1 indicating Churn. This systematic encoding approach ensured consistent interpretation across all components of the cognitive architecture.

Class Imbalance Management

The dataset's significant class imbalance necessitated a resampling approach. To address this challenge, the implementation utilized SMOTENN (Synthetic Minority Oversampling Technique combined with Edited Nearest Neighbors). This technique combines synthetic sample generation for the minority class with intelligent undersampling of the majority class, producing a balanced dataset while maintaining the statistical significance of the original data distribution.

SMOTENN: Addressing Class Imbalance

The SMOTENN (Synthetic Minority Oversampling Technique and Edited Nearest Neighbors) method was chosen for this study due to its dual benefits of addressing class imbalance and mitigating noise in the dataset. SMOTENN operates in two stages: first, it synthesizes new samples for the minority class (churn customers) by interpolating between existing data points. This oversampling helps to balance the dataset by increasing the representation of the minority class. Second, the Edited Nearest Neighbors (ENN) algorithm is applied to the majority class to remove noisy or borderline examples, further improving the dataset's quality and representativeness.

The decision to use SMOTENN was motivated by its ability to balance datasets effectively while retaining critical decision boundaries between classes. Unlike basic oversampling or undersampling techniques, SMOTENN provides a nuanced approach by both expanding the minority class and refining the majority class, making it particularly suitable for datasets with complex feature interactions like the Churn-in-Telecom Dataset. The implementation of SMOTENN followed a careful validation process to ensure that the synthetic samples and the cleaned majority class did not introduce bias.

Data Partitioning Strategy

The dataset was divided into training, validation, and test sets to ensure a robust evaluation of the models:

- **Training Set** (70%): Used for training the individual machine learning models within the cognitive architecture.
- **Validation Set** (15%): Used for hyperparameter tuning and model selection to optimize model performance before final testing.
- **Test Set** (15%): Used to evaluate the generalization ability of the trained models and the ensemble, providing an unbiased assessment of their predictive performance.

3.3.3 Model Evaluation Framework

The model evaluation framework implemented a comprehensive approach to assess the cognitive architecture's performance under various conditions. This framework incorporated both cross-validation during development and holdout validation for final performance assessment.

Validation Strategy and Cross-Validation Implementation

The validation strategy combined a three-data partitioning approach with 5-fold cross-validation to ensure reliable model evaluation and generalization. The dataset was divided into a training set (70%), a validation set (15%), and a test set (15%). The training set was used to develop the model and estimate parameters, such as weights and coefficients. The validation set enabled the tuning of hyperparameters, including learning rates and regularization strength, to improve model performance. Finally, the test set was reserved for an unbiased evaluation of the model's performance on unseen data, ensuring it could generalize effectively.

In addition to this, 5-fold cross-validation was used during the training phase to further assess model stability. This method divided the training data into five equal parts, or folds. Each fold served as a validation set once, while the remaining folds formed the training set. The results from all folds were averaged, providing a robust estimate of model performance and reducing the impact of data variance. Unlike the dedicated validation set, which was used for hyperparameter tuning, cross-validation focused on assessing overall model stability and detecting issues like overfitting or underfitting.

By combining the partitioning strategy with cross-validation, the evaluation framework ensured the model was both optimized and reliable for real-world applications.

3.3.4 Evaluation Metrics

The evaluation incorporated multiple metrics to provide a comprehensive assessment of the cognitive architecture's effectiveness. These metrics addressed various aspects of prediction quality, including accuracy, precision, recall, and F1-score. The framework paid particular attention to metrics relevant to imbalanced classification scenarios, ensuring meaningful evaluation of the architecture's performance in realistic churn prediction contexts.

3.3.5 Data Flow Management

The data flow design ensured proper isolation between the training, validation, and testing phases. Feature scaling parameters derived solely from training data prevented information leakage while maintaining consistent transformation across all dataset partitions. This systematic approach to data management preserved the integrity of the evaluation process while enabling reliable assessment of the cognitive architecture's performance under various operating conditions.

The preprocessing and evaluation frameworks established a robust foundation for assessing the cognitive architecture's effectiveness in real-time churn prediction. This comprehensive data preparation and validation approach ensured reliable evaluation of the architecture's performance while maintaining scientific rigor throughout the experimental process.

The preprocessing pipeline is implemented within the Data Reader Codelet, which reads the raw data from the external source, applies the necessary preprocessing steps, and then stores the processed data in memory objects for further use by the model-specific codelets. By ensuring that the data is clean, well-structured, balanced, and appropriately scaled, the preprocessing step helps to enhance the accuracy, fairness, and reliability of the machine learning models used within the cognitive architecture.

Model Configurations

Each of the six machine learning models implemented in the cognitive architecture was configured to optimize performance on the selected dataset. The configurations included:

1. **KNN (K-Nearest Neighbors):**
   - **Hyperparameters:** Hyperparameters: Number of neighbors (k), distance metric (Euclidean).
2. **Logistic Regression:**
   - **Hyperparameters:** Regularization strength (C), solver type.
3. **Decision Tree:**
   - **Hyperparameters:** Hyperparameters: Maximum depth, minimum samples split.
4. **Random Forest:**
   - **Hyperparameters:** Number of trees, maximum depth, minimum samples split.
5. **SVM (Support Vector Machine):**
   - **Hyperparameters:** Kernel type (linear, RBF), regularization parameter (C), gamma.
6. **Naive Bayes:**

With the data processing framework established, the following section details the technical implementation of the cognitive architecture components, demonstrating how they process and analyze the prepared data for real-time churn prediction.

3.4 Implementation

The implementation phase translated the theoretical framework into a functional cognitive architecture system for real-time churn prediction. This section details the technical implementation of the core architectural components, focusing on the memory object structure, codelet functionality, and their integration within the cognitive framework described in Figure 3.1.

3.4.1 Memory Object Implementation

The memory object system forms the foundation of data management within the cognitive architecture. This implementation comprises three primary memory object types, each serving specific functions in the architecture:

- The Data Memory Objects maintain the processed dataset, managing both features and corresponding labels. This component implements efficient data access methods, enabling rapid retrieval and update operations during model training and prediction phases.
- The Model State Memory Object maintains the current states of all machine learning models within the architecture. This component stores model predictions and performance metrics, facilitating dynamic adjustment of model weights and ensemble behavior.
- The Evaluation Metrics Memory Object maintains the evaluation results of individual models. It stores accuracy, precision, recall, and F1-score metrics, enabling the architecture to assess model performance and provide feedback to the dynamic weight adjustment process.
- The Prediction Memory Object manages individual model's ongoing predictions and performance metrics. Supporting the ensemble and dynamic weight adjustment process while optimizing memory usage.

3.4.2 Codelet System Implementation

The codelet system implements the processing logic of the cognitive architecture through specialized components, each handling specific aspects of the churn prediction process:

- The Data Reader Codelet implements the data ingestion pipeline, managing the transformation of raw data into a structured format suitable for model processing. This component handles proper data flow through the architecture.
- The Model Codelets implement six distinct machine learning algorithms: K-Nearest Neighbors (KNN), Logistic Regression, Decision Tree, Random Forest, Support Vector Machine (SVM), and Naive Bayes. Each codelet operates independently, implementing its respective algorithm's training and prediction logic while maintaining communication with the shared memory objects.

- The Evaluation Codelet evaluates model predictions. It receives the output from the Model Codelets and conducts performance evaluations using metrics like accuracy, precision, recall, and F1-score. The Evaluation Codelet analyzes individual model performance and updates the evaluation metrics in shared memory objects, providing feedback for subsequent ensemble adjustments.
- The Ensemble Codelet implements the final prediction aggregation logic, combining individual model predictions according to their dynamic weights. This component manages the weighted voting process, producing the final churn predictions based on the ensemble's collective intelligence.
- The Dynamic Weight Manager Codelet implements the adaptive weight adjustment system, monitoring model performance and updating ensemble weights. This component utilizes a sliding window mechanism to track recent performance metrics, implementing the weight adjustment algorithm detailed in the previous section.

3.4.3 Integration and Execution Framework for Coordinating Ensemble Learning Within the Cognitive Architecture

The integration framework implements the communication and coordination mechanisms between system components. This framework ensures proper sequencing of operations, from data preprocessing through model training to final prediction generation. The implementation utilizes a message-passing architecture, enabling asynchronous operation of codelets while maintaining data consistency through the memory object system.

3.4.4 Execution Flow Implementation

The execution flow implementation manages the sequential data processing through the cognitive architecture. This implementation follows a structured approach:

1. The initial phase activates the Data Reader Codelet, triggering the preprocessing pipeline and populating the Data Memory Object with processed features and labels.
2. The second phase initiates parallel execution of Model Codelets, each training its respective algorithm on the preprocessed data and generating predictions stored in the Prediction Memory Object.
3. The third phase activates the Evaluation Codelet, which receives predictions from the Model Codelets and performs model evaluation using metrics such as accuracy, precision, recall, and F1-score. The results are stored in the Evaluation Metrics Memory Object, providing performance feedback for the subsequent ensemble phase.
4. The fourth phase executes the Ensemble Codelet, implementing the dynamic weighted ensemble and static ensemble prediction aggregation process to generate final churn predictions.

5. The final phase activates the Dynamic Weight Manager Codelet, implementing the weight adjustment algorithm based on recent model performance metrics.

This modular and dynamic design allows the cognitive architecture to be adaptive, scalable, and capable of handling various machine-learning tasks efficiently. Separating each model into its codelet provides flexibility for further expansion and modification, while the dynamic weight management system ensures that the architecture continuously improves its decision-making process.

In summary, the cognitive architecture integrates multiple independent machine learning models into a cohesive ensemble system. The Dynamic Weight Manager Codelet plays a crucial role in ensuring that the ensemble adapts to changing conditions by adjusting the models' weights based on their performance. This structure allows for modularity, scalability, and robustness in processing and decision-making.

While the implementation framework establishes the system's structure, the dynamic weight management system provides its adaptive capabilities. The following section details this crucial component that enables real-time adaptation to changing prediction patterns.

3.5 Dynamic Weight Management

The dynamic weight management system plays a crucial role in the adaptive capabilities of the cognitive architecture's ensemble model. This system is designed to adjust the influence of individual machine learning models in the ensemble based on their performance over time. By continuously updating the weights assigned to each model, the dynamic weight management system ensures that the ensemble remains responsive to shifts in data patterns and model accuracy. This section explains the weight adjustment algorithm, the process of tracking model performance, and the subsequent weight modifications that enhance the ensemble's adaptability.

Weight Adjustment Algorithm

The weight adjustment algorithm is based on the principle of adaptive weighting, where the contribution of each model to the final ensemble prediction is adjusted according to its recent performance. The algorithm operates using the following steps

1. **Sliding Window Mechanism:**
    - **Definition:** Maintains a fixed-size window of recent predictions for each model.
    - **Function:** As new predictions are made, the oldest predictions are removed to keep the window size constant, ensuring that weight adjustments are based on the most recent performance.
2. **Weighted Prediction History:**
    - **Recency Weighting:** Assigns higher weights to more recent predictions within the sliding window, gradually reducing the influence of older predictions based on the forgetting rate.

- **Purpose:** Ensures that recent performance has a greater impact on weight adjustments, allowing the ensemble to respond swiftly to changes in data patterns.

3. **Calculating Weighted Accuracy:** It represents the proportion of correct predictions within the sliding window, weighted by their recency, and is computed as:

$$\text{Weighted Accuracy} = \frac{\sum_{i=1}^{n} w_i \cdot \text{Correct}_i}{\sum_{i=1}^{n} w_i}, \tag{3.2}$$

where w_i is the weight assigned to the i^{th} prediction, and Correct_i is 1 if the prediction is correct, 0 otherwise.

4. **Weight Update:** The weight update formula calculates the new weight for each model based on its performance:

$$W_{\text{new}} = W_{\text{current}} + \text{Learning Rate} \times (\text{Weighted Accuracy} - W_{\text{current}}), \tag{3.3}$$

where W_{new} is the updated weight, W_{current} is the current weight, and *Learning Rate* is a parameter controlling the speed of weight adjustment.

   - **Explanation:** Adjusts the current weight towards the weighted accuracy. A higher learning rate results in more significant weight changes.
   - **Learning Rate:** Controls the speed at which weights are updated in response to performance changes.
   - **Forgetting Rate:**Determines how quickly the influence of older predictions diminishes, ensuring adaptability.

5. **Normalization of Weights:**

   - **Process:** After updating, weights are normalized so that the sum of all model weights equals 1.

$$W_i = \frac{W_i}{\sum_{j=1}^{m} W_j}, \quad \text{for } i = 1, \ldots, m, \tag{3.4}$$

   where W_i is the weight of the i^{th} model, and m is the total number of models in the ensemble.

   - **Purpose:** Prevents any single model from disproportionately dominating the ensemble, maintaining a balanced influence across all models.

Implementation Details

The dynamic weight management system is implemented within the Dynamic Weight Manager Codelet. Key components and their functionalities include:

1. **Sliding Window Mechanism:**

- **Data Structure:** Utilizes a queue or circular buffer to efficiently manage the sliding window of recent predictions.
- **Operations:** Enqueues new predictions and dequeues the oldest ones as new data arrives.

2. **Performance Tracking:**
   - **Metric Calculation:** Continuously calculates weighted accuracy for each model within the sliding window.
   - **Historical Data:** Maintains historical performance data to inform weight adjustments.

3. **Weight Adjustment Process:**
   - **Update Execution:** Applies the weight update formula after each batch of predictions is processed.
   - **Adaptive Influence:** Increases weights for models demonstrating improved performance and decreases weights for underperforming models.

Parameter Justification

- **Window Size (20):** Balances responsiveness with stability, ensuring that recent performance is adequately represented without being overly sensitive to fluctuations.
- **Learning Rate (0.1):** Provides a moderate speed of weight adjustment, allowing the ensemble to adapt without drastic changes that could destabilize performance.
- **Forgetting Rate (0.35):** Ensures that older predictions gradually lose influence, maintaining the focus on recent performance metrics.
- **Minimum Weight (0.02) and Maximum Weight (1.0):** Prevent any model from being entirely excluded or from having excessive influence, maintaining a fair balance within the ensemble.

Benefits of Dynamic Weight Management

- **Adaptability:** Enables the ensemble to respond to changes in data distribution and model performance in real time.
- **Enhanced Performance:** By emphasizing well-performing models, the ensemble achieves higher accuracy and reliability.
- **Robustness:** Mitigates the impact of underperforming models, ensuring that the ensemble remains effective even as individual model performances fluctuate.

Having detailed the system's components and adaptive mechanisms, the following section presents the experimental framework designed to evaluate the cognitive architecture's effectiveness in real-time churn prediction scenarios.

3.6 Experimental Setup

The experimental setup for this research establishes the conditions under which the cognitive architecture and its components were evaluated. The experiments were designed to rigorously test the architecture's performance and adaptability, focusing on its ability to process data, train individual models, and dynamically adjust the ensemble based on model performance. This section outlines the datasets used, model configurations, dynamic weight management settings, and evaluation procedures implemented to ensure a comprehensive and reliable system assessment.

Dataset Selection

Two versions of a binary classification churn dataset were utilized to evaluate the cognitive architecture under different class distribution scenarios:

1. **Imbalanced Dataset:**
    - **Description:** The dataset contains 3,000 instances with 11 features. The class distribution reflects the natural imbalance observed in real-world scenarios: 2,600 instances belong to the majority class (non-churn), while 400 instances represent the minority class (churn).
    - **Purpose:** To assess how the cognitive architecture performs in handling class imbalance, particularly in real-world distributions, and its impact on the overall model accuracy and recall for the minority class.
2. **Balanced Dataset:**
    - **Description:** A balanced dataset was created from the imbalanced dataset using the SMOTENN (Synthetic Minority Over-sampling Technique combined with Edited Nearest Neighbors) technique. This process was applied only to the training set to ensure equal representation of both classes while maintaining an imbalanced test set to simulate real-world deployment scenarios.
    - **Purpose:** To evaluate the impact of training on a balanced dataset while testing on an imbalanced distribution. This approach mitigates biases introduced by class imbalance during training without compromising the relevance of the testing phase.

Dynamic Weight Management Configuration

The Dynamic Weight Manager Codelet was configured to dynamically adjust the influence of each model in the ensemble based on their performance. The configuration was designed to ensure that the ensemble remained responsive to changes in data patterns and model accuracy. The parameters used were:

- **Sliding Window Size:** 20 instances, used to track recent model predictions;

- **Learning Rate:** 0.1, controlling the speed of weight adjustment based on model performance;
- **Forgetting Rate:** 0.35, ensuring that older predictions have progressively less influence on weight adjustments;
- **Minimum Weight:** 0.02, ensuring that no model is entirely excluded from the ensemble;
- **Maximum Weight:** 1.0, capping the influence of any single model within the ensemble.

These settings ensured that the dynamic weight management system could quickly adapt to shifts in model performance, allowing the ensemble to maintain optimal decision-making.

Evaluation Procedure

The cognitive architecture and its components were evaluated using a systematic procedure to ensure an accurate and reliable assessment of model performance. The evaluation procedure included the following steps:

1. **Data Splitting:**
    - **Training Set:** 70% of the data, used to train the individual models;
    - **Validation Set:** 15% of the data, is used to evaluate both individual models and the ensemble;
    - **Test Set:** 15% of the data, used to evaluate the generalization ability of the trained models and the ensemble.
2. **Cross-Validation:**
    - **Approach:** 5-fold cross-validation was applied during model training to ensure that models generalized well across different subsets of the data, mitigating the risk of overfitting.
3. **Model Training and Prediction:**
    - **Process:** Each model-specific codelet trained its respective machine learning model on the training set and generated predictions on the testing set;
    - **Storage:** Trained models and predictions were stored in shared memory objects for aggregation.
4. **Dynamic Weight Adjustment and Ensemble Evaluation:**
    - **Process:** The Dynamic Weight Manager Codelet dynamically adjusted the weights of each model based on their performance within the sliding window;

- **Ensemble Decision:** The Ensemble Codelet aggregated predictions using the updated weights to produce the final ensemble decision;
- **Comparison:** Ensemble performance was compared against individual model performances and static ensemble methods.

5. **Performance Metrics Calculation:**
   - **Metrics Evaluated:** Accuracy, Precision, Recall, F1-Score, ROC-AUC;
   - **Procedure:**Metrics were calculated for each model, the dynamic ensemble, and the static ensemble to assess effectiveness.
6. **Statistical Analysis:**
   - **Objective:** Determine the statistical significance of performance differences between dynamic and static ensembles;
   - **Method:** Conducted a Wilcoxon signed-rank statistical test to validate findings.
7. **Model Combination Analysis:**
   - **Objective:** Identify optimal model combinations that yield the highest performance metrics;
   - **Process:** Evaluated all possible model combinations to determine which subsets provided the best accuracy, precision, recall, and F1-Score

Implementing these experimental procedures requires specific computational resources and configurations. The following section details the technical environment established to ensure reliable and reproducible evaluation of the cognitive architecture.

3.7 Computational Environment

The experimental conditions are meticulously designed to ensure the study's findings' reliability, validity, and reproducibility. This section outlines the hardware specifications, software environment, configurations, computational resources, libraries, tools, and reproducibility measures employed during the experiments.

Hardware Specifications

The experiments were conducted on a high-performance computing system to accommodate the computational demands of training multiple machine-learning models and managing dynamic ensemble processes. The detailed hardware specifications are as follows:

1. Processor (CPU)
   - Model: Intel® Core™ i7-1165G7 (11th Generation);

- Cores: Quad-core (4 cores);
- Base Speed: 2.8 GHz;
- Maximum Turbo Speed: 4.7 GHz.

2. Memory (RAM)
   - Capacity: 16 GB DDR4;
   - Configuration: Dual Channel (8GB x 2);
   - Speed: 2666 MHz.
3. Graphics Processing Unit (GPU)
   - Model: Intel Iris Xe Graphics;
   - Type: Integrated with shared memory.
4. Storage
   - Type: PCIe NVMe M.2 SSD;
   - Capacity: 256 GB.

Software Environment

The software environment is configured to provide a stable and efficient platform for implementing the cognitive architecture and running machine learning experiments. The specific software components and their versions are detailed below:

1. Operating System
   - Distribution: Windows;
   - Kernel Version: Windows 11 OS.
2. Programming Language
   - Java: OpenJDK 11.
3. Machine Learning Libraries
   - Weka: Version 3.8.6;
   - Apache Commons Math: Version 3.6.1 (for additional mathematical computations);
   - SMOTENN Implementation: In-house implementation integrated within the preprocessing pipeline.
4. Development Tools
   - Integrated Development Environment (IDE): IntelliJ IDEA Ultimate Edition 2023.1;

- Build Tools: Maven 3.8.6 for project dependency management and build automation.

5. Version Control
   - Git: Version 2.34.1, with the repository hosted on GitHub for source code management and collaboration; github.com/H-IAAC/meta-7.git.
6. Additional Tools
   - Data Visualization: Matplotlib and Seaborn (via Python scripts) for generating EDA figures and correlation heatmaps;
   - Statistical Analysis: for conducting advanced statistical tests to validate performance differences.

Libraries and Tools

The implementation relied on a combination of libraries and tools to facilitate data processing, model training, and evaluation:

1. Java Libraries
   - Weka: Core library for implementing machine learning algorithms and handling data preprocessing tasks;
   - Apache Commons Math: Supplementary library for advanced mathematical operations and statistical computations.
2. Python Libraries
   - Pandas: For data manipulation and preprocessing tasks not directly handled within Java.
   - Matplotlib & Seaborn: For creating visualizations such as correlation heatmaps and feature distributions during EDA.
3. Data Visualization and Statistical Analysis
   - Scipy: Utilized for conducting statistical tests to determine the significance of performance differences between models and ensembles.
4. Additional Tools
   - Jupyter Notebooks: Facilitated exploratory data analysis and prototyping of data preprocessing scripts.

By meticulously documenting and standardizing all aspects of the experimental setup, the study ensures that results can be reliably reproduced and validated by other researchers, thereby enhancing the credibility and impact of the findings.

3.8 Multi-Dataset Evaluation

To evaluate the generalizability of the proposed RTCP system, this study includes three publicly available telecom churn datasets: the Churn-in-Telecom Dataset, the Churn-data-UCI Dataset, and the IBM Telco Dataset. This multi-dataset approach ensures that the proposed solution is tested under different scenarios, addressing potential dataset dependency and aligning with the recommendations for broader validation.

Dataset Structure and Features

The three datasets employed in this study vary in size and feature composition, offering a comprehensive basis for assessing the RTCP system's adaptability:

1. Churn-in-Telecom Dataset: Contains 3,000 customer records with 11 attributes, focusing on various aspects of customer behavior and service usage;
2. IBM Telco Dataset: Includes 7,043 customer records and 33 features, covering demographic information, account characteristics, and service utilization patterns;
3. Churn-data-UCI Dataset: Comprises 30,000 customer records with 20 features, capturing diverse customer service interactions and usage metrics.

The three datasets employed in this study vary in size and feature composition, offering a comprehensive basis for assessing the RTCP system's adaptability:

Table 3.2: Dataset Characteristics Summary. The Class Distribution column shows the number of churned customers versus non-churned customers (churn/not-churn) for each dataset.

Dataset Name	Size	Features	Class Distribution	Source	Format
Churn-in-Telecom	3,000	11	400/2,600	[74]	CSV
IBM Telco	7,043	33	1,869/5,174	[24]	CSV
Churn-data-UCI	30,000	20	7,597/22,403	[24]	CSV

The same preprocessing steps were applied consistently to all three datasets to ensure comparability and maintain high data quality throughout the analysis as discussed in 3.3.

The performance of the proposed RTCP system was evaluated using all three datasets. Additionally, a comparison with an adaptive ensemble framework from related work was conducted, using the same performance metrics—accuracy, precision, recall, and F1-score. This comparative analysis assessed the system's effectiveness across different datasets, demonstrating its generalizability and adaptability in various data scenarios.

3.9 Comparative Framework

To establish a baseline for evaluating the proposed RTCP system, this study compares with an existing **adaptive ensemble framework** described in related work [24]. The chosen framework employs ensemble learning techniques and adaptive weighting strategies, similar to the RTCP system. However, it is not designed specifically for real-time processing but is a robust point of reference for adaptive performance across churn prediction datasets.

Overview of the Adaptive Ensemble Framework in the Literature

The adaptive ensemble framework uses dynamic weight adjustment to prioritize well-performing models, similar to the proposed RTCP system. It was evaluated across three telecom churn datasets: the Churn-in-Telecom Dataset, the Churn-data-UCI Dataset, and the IBM Telco Dataset. The framework's design aligns with the goal of adaptive prediction accuracy, allowing for a direct comparison with the RTCP system in terms of performance metrics such as accuracy, precision, recall, and F1-score.

Evaluation Strategy

In this comparative evaluation, the proposed RTCP system and the adaptive ensemble framework were subjected to the same preprocessing steps, data partitioning strategies, and performance metrics. The evaluation aims to demonstrate the effectiveness of the RTCP system's real-time adaptation capabilities compared to the adaptive ensemble's batch-processing design. This comparative analysis will be revisited in the Results chapter, where a detailed performance comparison will be presented to highlight the strengths and limitations of each approach across different datasets.

3.10 Final Remarks

This chapter presented our comprehensive methodology, designed and implemented using a cognitive architecture framework, for real-time churn prediction. Our methodology implements multiple integrated components to achieve adaptive and accurate prediction capabilities.

The implementation establishes a cognitive architecture that combines memory objects and specialized codelets. Memory objects implement efficient data management through dedicated components for data storage, model state tracking, and prediction management. The codelet system implements specialized processing units, including data readers, model processors, and weight managers, working together to achieve real-time prediction capabilities.

The dynamic weight management system incorporates adaptive mechanisms for optimizing ensembles. By employing sliding window approaches and weight adjustment algorithms, the system maintains responsiveness to evolving data patterns while ensuring prediction stability. The implementation leverages precisely calibrated parameters, such as learning rates and forgetting factors, to maximize system effectiveness.

The evaluation implements testing procedures across multiple datasets. By utilizing three distinct telecom datasets - Churn-in-Telecom, IBM Telco, and Churn-data-UCI -, the methodology ensures comprehensive validation of the system's generalizability. The implementation includes comparative analysis against existing adaptive ensemble frameworks, establishing clear benchmarks for performance assessment. The computational environment implements specific hardware and software configurations, ensuring reproducibility and consistent performance evaluation.

This methodological framework provides the foundation for subsequent experimental evaluation and analysis, implementing a comprehensive approach to real-time churn prediction through cognitive architectures. The next chapter will present the results obtained through this methodology, demonstrating its effectiveness in addressing the challenges of real-time customer churn prediction.

Chapter 4

Results and Discussion

This chapter presents a comprehensive analysis of the performance of cognitive architecture in real-time churn prediction by examining its predictive accuracy and adaptive capabilities. The evaluation framework presented in Figure 4.1 implements multiple assessment approaches to validate the system's effectiveness across different scenarios and datasets. The analysis proceeded in five key phases:

1. Individual model performance evaluation on balanced and imbalanced datasets;
2. Comparative analysis of static versus dynamic ensemble approaches;
3. Assessment of model contributions through systematic component removal;
4. Statistical validation of performance differences;
5. Cross-dataset evaluation and comparison with existing methods.

The experiments implemented a stream-based evaluation approach that processed the data incrementally in folds to simulate real-world scenarios. This methodology enables the assessment of both immediate- and long-term adaptive capabilities. Figure 4.1 illustrates the systematic evaluation framework used in this study.

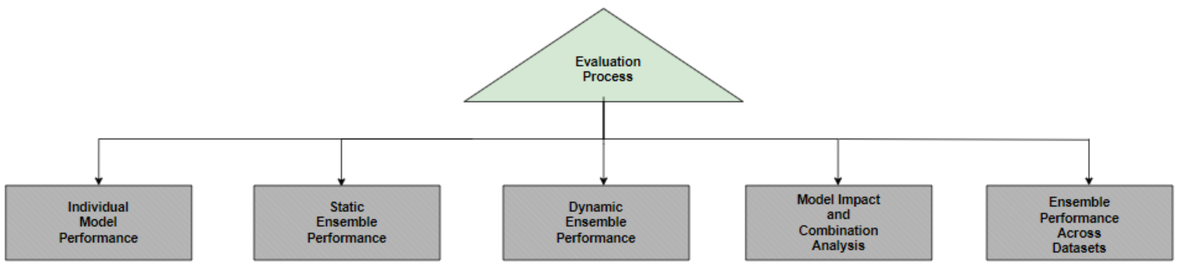
Evaluation Process
Individual Model Performance
Static Ensemble Performance
Dynamic Ensemble Performance
Model Impact and Combination Analysis
Ensemble Performance Across Datasets

Figure 4.1: Systematic approach for assessing model and ensemble performance across multiple datasets and scenarios. The framework implements incremental data processing to evaluate both static performance metrics and dynamic adaptation capabilities.

The evaluation implements multiple performance metrics to ensure a comprehensive assessment.

1. **Accuracy:** Measures overall prediction correctness;
2. **Precision:** Evaluates prediction reliability;
3. **Recall:** Assesses prediction completeness;
4. **F1-score:** Provides balanced assessment of precision and recall.

In addition, statistical validation was implemented using Wilcoxon signed-rank tests to verify the significance of the performance differences between the static and dynamic approaches. This ensures that the observed improvements represent genuine advances in the predictive capability rather than random variations.

To establish the generalizability of the results, the evaluation was extended across the three datasets.

1. Churn-in-Telecom Dataset (3,000 records, 11 features);
2. IBM Telco Dataset (7,043 records, 33 features);
3. Churn-data-UCI Dataset (30,000 records, 20 features).

The following sections present the detailed results of these evaluations, beginning with individual model performance and progressing through ensemble behavior, statistical validation, and comparative analysis. This structured approach enables a systematic assessment of the effectiveness of the cognitive architecture in real-time churn prediction while maintaining clear connections to the research objectives established in chapter 1.

4.1 Model Effectiveness

This section provides a comprehensive analysis of the performance of individual machine-learning models across both imbalanced and balanced datasets. The performance of each model was evaluated using the following key classification metrics: accuracy, precision, Recall, F1-Scor. These metrics are crucial for understanding the behavior of models under different data distributions and assessing their ability to generalize effectively to unseen data. The models are compared not only in terms of their static performance but also in terms of their adaptability when the class imbalance is addressed.

4.1.1 Individual Model Performance

Performance on the Imbalanced Dataset

On an imbalanced dataset, where minority class 1 (churn) is significantly underrepresented, the model performance varies significantly. Although some models achieved high accuracy, this metric alone did not provide a complete picture, as it often masked poor performance in recalling minority-class instances. Models that performed well in terms of accuracy sometimes exhibited lower recall and F1-Scores, indicating difficulties in correctly identifying minority classes.

Model	Accuracy	Precision	Recall	F1-Score	ROC Area
Naive Bayes	77.73%	0.7758	0.7773	0.7764	0.8501
Decision Tree	**85.20%**	**0.8511**	**0.8520**	**0.8509**	**0.8701**
Random Forest	**87.60%**	**0.8763**	**0.8760**	**0.8744**	**0.9307**
KNN	78.53%	0.7983	0.7853	0.7714	0.8198
SVM	80.93%	0.8076	0.8093	0.8077	0.7904
Logistic Regression	81.07%	0.8089	0.8107	0.8090	0.8783

Table 4.1: Performance metrics for imbalanced Churn-in-Telecom dataset, the evaluation of individual models on the imbalanced version of the dataset.

The results in Table 4.1 show that Naive Bayes achieved an accuracy of 77.73%, yet its recall was also 77.73%, suggesting that while the model could identify churn cases, it struggled to generalize its predictions effectively. The model's ROC-AUC score of 0.8501 highlighted its moderate ability to distinguish between classes but underscored the limitations of linear models on imbalanced datasets.

On the other hand, nonlinear models, such as Random Forest and Decision Tree, handled the imbalanced dataset more effectively. Random Forest achieved the highest accuracy at 87.60%, with a precision of 87.63%, recall of 87.60%, and F1-Score of 87.44%. These metrics indicate a more balanced performance, where the model was able to maintain high precision while correctly identifying a significant portion of the minority class instances. DecisionTree also performed well, with an accuracy of 85.20% and an F1-Score of 85.09%, demonstrating its ability to model complex feature interactions even under class imbalance.

KNN, while achieving a slightly weaker overall performance, achieved an accuracy of 78.53% and an F1-Score of 77.14%. Its lower performance in recall (78.53%) and ROC-AUC (0.8198) reflects the challenges faced by this distance-based model in distinguishing between the majority and minority classes when the data are skewed.

SVM and Logistic Regression, both linear models, performed similarly, with SVM achieving an accuracy of 80.93% and an F1-Score of 80.77%, whereas Logistic Regression reached an accuracy of 81.07% and an F1-Score of 80.90%. These models struggled to balance recall and precision on an imbalanced dataset, likely because of their reliance on linear decision boundaries, which limited their ability to model the complex relationships inherent in the data.

Performance on the Balanced Dataset

After balancing the dataset using the SMOTENN technique, all models exhibited substantial improvements across key metrics, particularly in recall and F1-Score. This highlights the models' improved ability to correctly classify both the majority and minority classes, as the balancing process alleviated the bias towards the majority class.

Model	Accuracy	Precision	Recall	F1-Score	ROC Area
Naive Bayes	84.68%	0.8597	0.8468	0.8475	0.8899
Decision Tree	92.11%	0.9211	0.9211	0.9211	0.9361
Random Forest	**98.01%**	**0.9801**	**0.9801**	**0.9801**	**0.9982**
KNN	**98.82%**	**0.9883**	**0.9882**	**0.9882**	**0.9892**
SVM	82.86%	0.8283	0.8286	0.8279	0.8216
Logistic Regression	82.77%	0.8276	0.8277	0.8267	0.8976

Table 4.2: Performance metrics for the balanced Churn-in-Telecom dataset, highlighting the top two performing models across all evaluation metrics.

KNN demonstrated the most significant performance gain on the balanced dataset, achieving an accuracy of 98.82% and corresponding F1-Score of 98.82%. This marked improvement illustrates KNN's strength in environments where the class distribution is more equitable, allowing the model to leverage its distance-based approach more effectively.

Similarly, the Random Forest improved dramatically, achieving an accuracy of 98.01% and an F1-Score of 98.01%. The model's high precision (98.01%) and ROC-AUC (0.9982) underscore its capacity to capture complex nonlinear relationships in the data when the class imbalance is corrected. DecisionTree also performed exceptionally well, with an accuracy of 92.11% and an F1-Score of 92.11%, reflecting its ability to generalize more effectively under balanced data conditions.

Even models that struggled with an imbalanced dataset, such as NaiveBayes, showed marked improvements. NaiveBayes achieved an accuracy of 84.68% and an F1-Score of 84.75%, indicating that balancing the data allowed it to generalize better across both classes. The ROC-AUC score of 0.8899 further highlights its enhanced discriminative ability on a balanced dataset.

SVM and Logistic Regression both improved the performance but still lagged behind the nonlinear models. The SVM achieved an accuracy of 82.86% and an F1-Score of 82.79%, whereas Logistic Regression achieved an accuracy of 82.77% and an F1-Score of 82.67%. These results suggest that, while balancing the data helped these models, their linear nature prevented them from fully capturing the complex patterns present in the dataset.

The analysis of the individual model performance across both balanced and imbalanced datasets established the baseline capabilities for each algorithm. These individual performance characteristics provide the foundation for understanding how models contribute to ensemble performance, and inform the following comparative analysis.

4.1.2 Comparison of Models

Building on the individual performance metrics established above, this section presents a systematic comparison of model behaviours across different scenarios. Comparative analysis revealed how different machine learning algorithms respond to variations in data distribution, particularly focusing on the distinction between linear and nonlinear approaches.

The comparison of model performance across both datasets provides key insights into how different machine learning algorithms respond to variations in data distribution. The analysis revealed that nonlinear models, such as Random Forest and KNN, consistently outperformed other models, particularly on the balanced dataset. These models excelled because of their ability to capture complex nonlinear patterns in the data, which became more pronounced when the class distributions were equalized.

On the imbalanced dataset, Random Forest and Decision Tree stood out, with Random Forest achieving the highest accuracy (87.60%) and F1-Score (87.44%). These models are better equipped to handle skewed class distributions because of their inherent ability to model complex feature interactions.

In contrast, linear models, such as Naive Bayes, SVM, and Logistic Regression, struggled with the imbalanced dataset, with lower recall and F1-Scores. However, their performance improved notably on the balanced dataset, with NaiveBayes increasing its accuracy to 84.68%, and both SVM and Logistic Regression achieving accuracies above 82%.

The most substantial improvement was observed in KNN, which achieved a near-perfect performance on the balanced dataset with an accuracy of 98.82% and an F1-Score of 98.82%. This demonstrates the KNN's sensitivity to class distribution, as balancing the data significantly enhances its ability to distinguish between classes.

Overall, the comparison highlighted the critical importance of data balancing in classification tasks. The balanced dataset allowed all models to perform more consistently across all metrics, reducing bias towards the majority class and enabling more accurate and reliable predictions. Non-linear models, particularly Random Forest and KNN, showed the greatest capacity to exploit the more equitable data distribution, whereas linear models, though improved, still lagged behind in terms of their flexibility and ability to model complex patterns.

Figures 4.2 and 4.3 visually depict the comparison of model accuracy and F1-Score across both datasets, further illustrating the performance disparities between the models on imbalanced and balanced data.

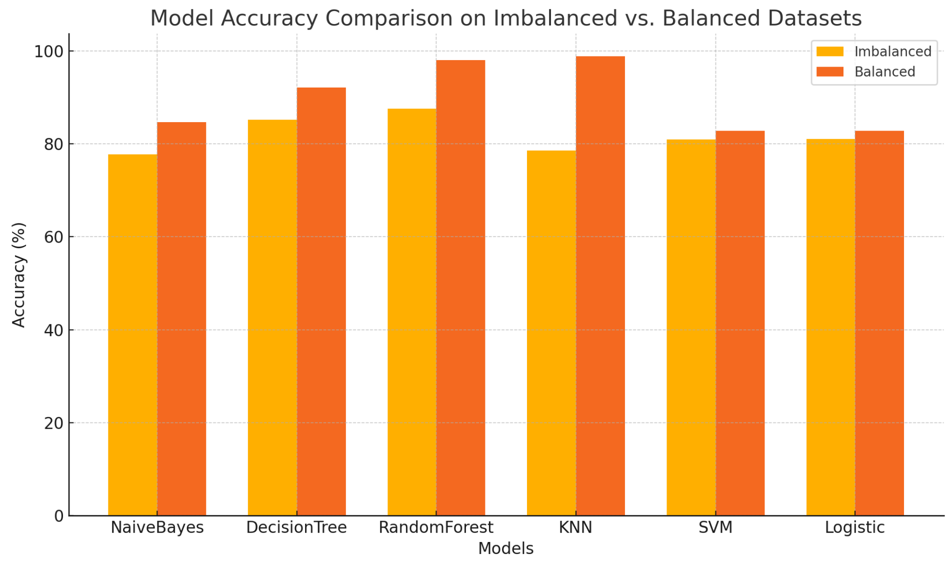

Figure 4.2: Model accuracy comparison between balanced and imbalanced datasets.

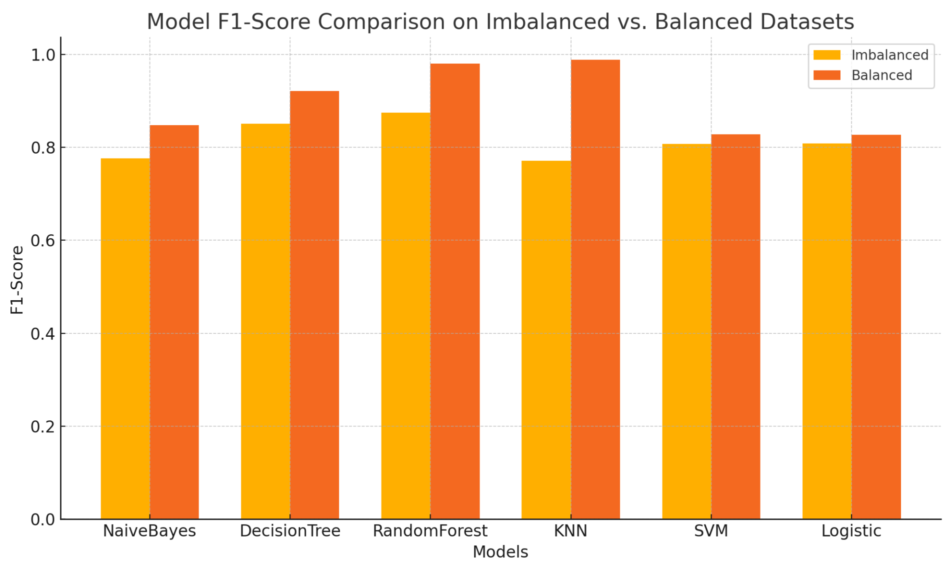

Figure 4.3: Model F1-Score comparison between balanced and imbalanced datasets.

These comparative findings highlight the varying strengths of the different algorithms, particularly the superior performance of nonlinear models in handling complex patterns. This understanding of individual model capabilities provides a crucial context for analyzing collective behavior within an ensemble framework.

4.2 Ensemble Performance

By leveraging the insights gained from the individual model comparisons, this section evaluates how these models collectively perform within the ensemble framework. The analysis examines both static and dynamic ensemble approaches, focusing on how the combination of models enhances the prediction capability beyond individual performance levels.

This section evaluates the effectiveness of the ensemble model across both imbalanced and balanced datasets. The performance of the ensemble was examined through dynamic weighting, where model weights were adjusted over time in response to the performance, and static weighting, where the model weights remained fixed. Additionally, the impact of individual models on the ensemble was analyzed by removing models individually and observing the subsequent changes in performance metrics, such as accuracy, precision, recall, and F1-Score. This analysis aimed to demonstrate the effectiveness of ensemble learning and to highlight the significance of individual model contributions.

Dynamic Ensemble Performance Over Time

The dynamic ensemble model was designed to adjust the weights of individual models adaptively as new folds of data were processed. This allowed the ensemble to prioritize the models that performed well on each fold, thereby optimizing the overall performance of the system over time. The performance of the dynamic ensemble was evaluated on both the imbalanced and balanced datasets, providing insight into how it adapted to different data distributions

Imbalanced Dataset On the imbalanced dataset, the dynamic ensemble faced the challenge of balancing predictions between the overrepresented majority class and the underrepresented minority class. Initially, the weights were distributed relatively evenly across all the models, reflecting the uncertainty of which model would perform best. However, as more data folds were processed, models such as Random Forest and Decision Tree quickly gained higher weights, owing to their ability to handle the complex relationships between features in imbalanced data. These models consistently contribute more to the ensemble decision-making process and improve the performance over time.

The performance of the dynamic ensemble fluctuated slightly as new folds were processed, reflecting the varying effectiveness of individual models on different data segments. Despite these fluctuations, the dynamic ensemble maintained a strong overall performance, achieving an average accuracy of 82.93%, precision of 80.66%, recall of 72.77%, and an F1-Score of 76.51%. These metrics indicate that the dynamic weighting mechanism allowed the ensemble to adapt to imbalanced data, particularly by leveraging models such as Random Forest and Decision Tree, which outperformed other models in handling skewed class distributions.

Balanced Dataset On the balanced dataset, where class distribution was balanced, the dynamic ensemble demonstrated even greater stability and higher performance across all metrics. With the class imbalance mitigated, models such as KNN and Random Forest were assigned significantly more weight because of their strong performance on balanced data. This shift in weighting allowed the dynamic ensemble to achieve much higher accuracy and generalization.

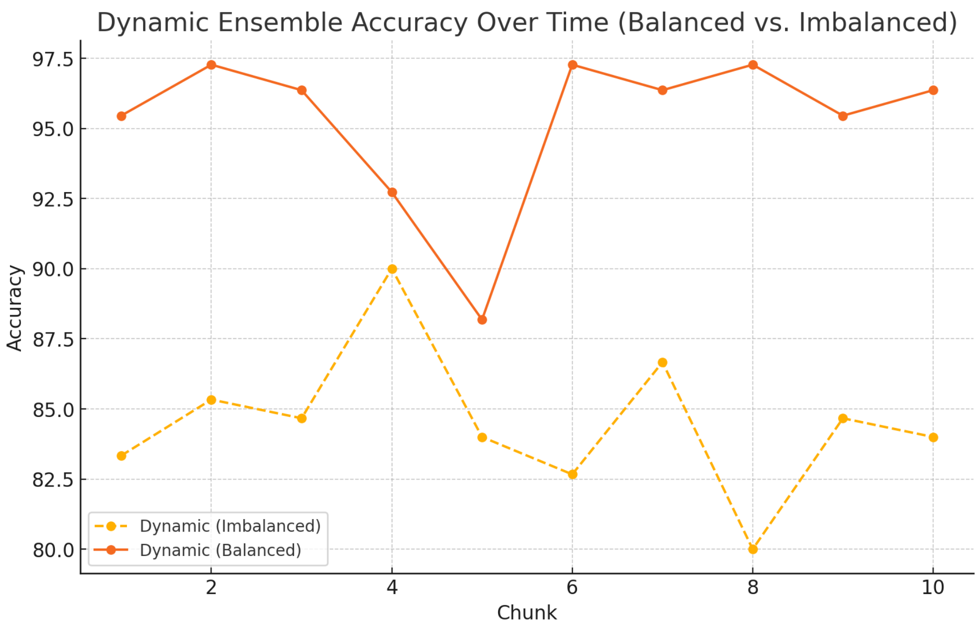

Figure 4.4: Dynamic ensemble accuracy over time (balanced vs. imbalanced datasets).

The dynamic ensemble on the balanced dataset achieved an average accuracy of 95.92%, precision of 97.71%, recall of 95.06%, and an F1-Score of 96.37%. These metrics represent a substantial improvement over the imbalanced dataset, highlighting the ability of the ensemble to adapt dynamically to a more equitable class distribution. The adaptability of the ensemble, driven by its ability to prioritize stronger models, was particularly evident in later data folds, where the performance consistently remained high as the ensemble was adjusted to evolving data patterns.

Figure 4.4 illustrates the evolution of the dynamic ensemble accuracy over time for both the imbalanced and balanced datasets. The figure shows that while the performance on the imbalanced dataset fluctuated more significantly owing to varying model contributions, the dynamic ensemble on the balanced dataset maintained a much more stable and upward trend in performance.

The dynamic ensemble on the balanced dataset achieved an average accuracy of 95.92%, precision of 97.71%, recall of 95.06%, and an F1-Score of 96.37%. These average metrics were calculated by calculating the mean of the accuracy, precision, recall, and F1-scores obtained across 10 evaluation folds, where each fold involved separate training and testing sets. This approach ensures that the reported metrics reflect the generalization capabilities of the ensemble across the entire dataset.

This substantial improvement over the imbalanced dataset highlights the ability of the dynamic ensemble to effectively adapt to a more equitable class distribution. The adaptability of the ensemble, driven by its capacity to prioritize stronger models, was particularly evident in later data folds, where the performance remained consistently high as the ensemble was adjusted to evolving data patterns.

Static Ensemble Performance Over Time

Unlike the dynamic ensemble model, the static ensemble model used fixed weights for each model throughout the evaluation process. Although this approach offers consistency, it lacks the flexibility to adapt to changing data patterns over time. As a result, the static ensemble's performance was more stable but less optimized compared to the dynamic ensemble, particularly on datasets where class distributions or feature relationships evolved

Imbalanced Dataset On the imbalanced dataset, the static ensemble achieved an average accuracy of 82.00%, precision of 84.75%, recall of 73.53%, and an F1-Score of 78.74%. Although these metrics were relatively strong, they were slightly lower than those of the dynamic ensemble. The fixed weighting prevented the static ensemble from fully capitalizing on stronger models, such as the Random Forest and Decision Tree, which performed well on imbalanced data. This limitation reduces the static ensemble's ability to adapt to the varying effectiveness of models across different folds of data, particularly when the performance of certain models, such as Naive Bayes, fluctuates more significantly.

Balanced Dataset On the balanced dataset, the static ensemble showed improved performance across all metrics, achieving an average accuracy of 95.45%, precision of 96.88%, recall of 95.38%, and F1-Score of 96.12%. These metrics are comparable to those of the dynamic ensemble, but the static ensemble still falls short in terms of its ability to optimize the contribution of the top-performing models over time. Although the static ensemble performed well on the balanced dataset, its lack of adaptability limited its capacity to fully exploit the strengths of models such as KNN and Random Forest as effectively as the dynamic ensemble.

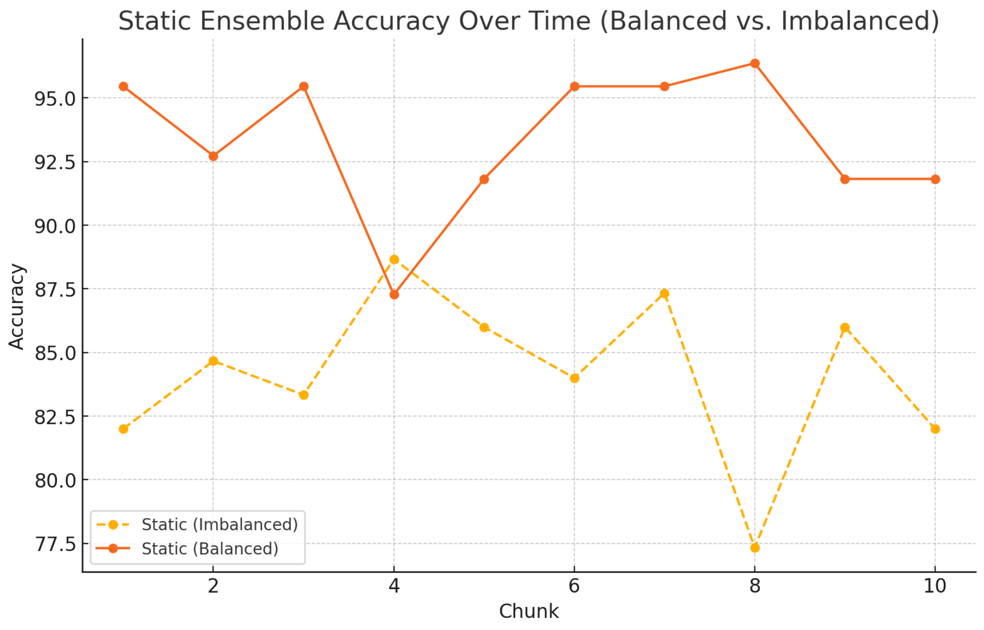

Figure 4.5: Static ensemble accuracy over time (balanced vs. imbalanced datasets).

Figure 4.5 depicts the accuracy of the static ensemble over time for both the imbalanced and balanced datasets. The figure highlights the relatively stable performance of the static ensemble but also underscores its inability to dynamically adjust to changes in data distribution and model effectiveness.

Overall, the static ensemble's fixed weighting strategy, which is effective in maintaining consistent performance, was outperformed by the dynamic ensemble, particularly in environments in which data conditions evolved over time. This highlights the importance of adaptability in ensemble learning, particularly when dealing with real-world datasets that can change incrementally.

The static ensemble, while effective, lacked the ability to dynamically fine-tune the contributions of the individual models. This highlights the importance of adaptability in ensemble learning, particularly when dealing with evolving datasets.

Model Impacts on Ensemble

The cognitive architecture framework used in this study introduces a unique advantage in its ability to dynamically manipulate model performance using codelets, a feature that allows models to be selectively activated or deactivated based on their performance. During the ensemble learning process, models encapsulated as codelets can be adjusted in real time, where poorly performing codelets may be temporarily shut down, and high-performing ones can be prioritized. This dynamic adaptability not only enhances the robustness of the ensemble but also provides insights into which models or combinations contribute most effectively to the overall system performance.

By manipulating the codelets dynamically, it is possible to test the ensemble under different configurations, thereby determining the optimal combination of the models. This experimentation allows the identification of which models can be deactivated without significantly affecting performance, which is crucial for maintaining high predictive accuracy. Through this mechanism, the architecture offers a flexible and responsive way to ensure that only the most valuable models contribute to real-time decision making, whereas others can be deactivated or reactivated based on evolving data patterns and model performance.

Imbalanced Dataset In scenarios where the dataset is imbalanced, such as the churn prediction context explored here, the removal of high-performing models such as Random Forest or Decision Tree resulted in a notable decline in the ensemble's accuracy. For instance, removing the Random Forest led to a decrease in accuracy from 82.93% to 82.00%, whereas removing the Decision Tree lowered the accuracy to 82.53%. This underscores the pivotal role of these models in handling complex and imbalanced data. Their capacity to model intricate, nonlinear relationships among features is essential for effectively identifying churn cases.

In contrast, models such as naïve Bayes and KNN had less impact when removed, suggesting that their contribution was not as critical in the imbalanced dataset. This could be due to their limitations in capturing the complexities of the minority class, where more sophisticated models such as the Random Forest excel. The ability of the cognitive architecture to deactivate these less impactful models without significantly hindering their performance further demonstrates its adaptive capability.

Balanced Dataset the architecture demonstrated its effectiveness on the balanced dataset. The removal of KNN and Random Forest had the most significant impact, with accuracies dropping from 95.92% to 90.93% and 91.02%, respectively. These models, which excel in nonlinear pattern recognition, are instrumental in maintaining high levels of accuracy for balanced class distributions. Their absence clearly illustrates their essential role in the ensemble's success with balanced datasets, in which distinguishing between classes is equally challenging.

In contrast, removing models, such as Naive Bayes, SVM, or Logistic Regression, had a minimal impact, reflecting their lower individual contribution in the context of this balanced dataset. Interestingly, removing Naive Bayes from the ensemble improved the performance, particularly on the balanced dataset, where the accuracy increased from 95.92% to 95.29%. This suggests that Naive Bayes may introduce noise or inconsistencies in predictions when more sophisticated models, such as KNN and Random Forest, are better suited to handle the complexity of the data.

Model Removed	Accuracy (Imbalanced)	Accuracy (Balanced)
Baseline (All Models)	82.93%	95.92%
Without Naive Bayes	85.60%	95.29%
Without Decision Tree	**82.53%**	**91.75%**
Without Random Forest	**82.00%**	**91.02%**
Without KNN	82.93%	90.93%
Without SVM	85.20%	95.92%
Without Logistic Regression	85.33%	95.92%

Table 4.3: Performance metrics with various models removed from the ensemble. The table highlights the significant drop in accuracy for both imbalanced and balanced datasets when Decision Tree and Random Forest are removed.

Table 4.3 summarizes the performance metrics when specific models are removed from the ensemble. The table highlights the significant contributions of models such as Random Forest and KNN, while also demonstrating that some models, such as Naive Bayes, may introduce noise or reduce the ensemble's overall accuracy.

Manipulating Codelets for Optimal Model Combinations

The ability to manipulate and deactivate codelets in real time based on their performance provides a distinct advantage in determining the best combination of models. By systematically deactivating underperforming codelets such as Naive Bayes, the cognitive architecture can focus on models such as Random Forest and KNN, which consistently show superior performance. This adaptability ensures that the ensemble remains responsive to changing data conditions, thereby improving overall accuracy, precision, and recall without being weighed down by less effective models.

The results underscore the flexibility of the cognitive architecture in managing ensemble performance. By leveraging the capacity of the architecture to activate or deactivate models dynamically, it is possible to maintain a highly efficient ensemble in which only the most impactful models contribute to real-time decision-making. This approach not only optimizes performance but also provides valuable insights into the contributions of individual models within the system.

The ensemble performance analysis demonstrated the potential benefits of combining multiple models. However, the effectiveness of this combination depends critically on the weighting and adjustment of the individual model contributions over time, which is examined in detail in the following section.

4.2.1 Dynamic Weight Application Over Time

Building on the ensemble performance analysis, this section examines how the dynamic weight adjustment mechanism influences the model contributions throughout the prediction process. The analysis focused on the system's ability to adapt weights in response to changing performance patterns, thereby providing insight into the architecture's real-time adaptation capabilities.

The dynamic ensemble approach employed in this study is characterized by a continuous adjustment of the model weights as new folds of data are processed. This methodology allows the ensemble to prioritize models that demonstrate superior performance over time, while reducing the influence of underperforming models. As the evaluation progressed, significant fluctuations were observed in the weights assigned to the individual models, with the adjustments driven by the relative performance of each model across the evolving dataset. Thus, the dynamic weight-adjustment mechanism offers a powerful way to ensure that the ensemble remains adaptable and responsive to changing data landscapes.

Imbalanced Dataset: When applied to the imbalanced dataset, the dynamic weight adjustment system initially distributed weights relatively evenly across all models. This approach reflects the uncertainty of which models excel during the early stages of data processing. However, as the system progressed through subsequent data folds, certain models began to distinguish themselves, resulting in weight adjustments that favored stronger models.

KNN, Random Forest, and SVM were the initial beneficiaries of slightly higher weights owing to their early performance on the imbalanced dataset. As the system processed more data, the weight distribution shifted dynamically, with Logistic Regression, Naive Bayes, and Random Forest maintaining their higher weights, owing to their ability to handle the complexities inherent in imbalanced data. Conversely, naive Bayes and SVM, despite showing initial promise, saw their weights reduce as they struggled to sustain consistent performance across different folds. These adjustments allowed the ensemble to adapt better to the imbalance in the data, ensuring that the models with stronger generalization capabilities contributed more significantly to the final predictions.

The continuous reassignment of weights based on performance allowed the ensemble to dynamically allocate more influence to models that are better at handling skewed data distributions. This mechanism significantly boosted the overall performance of the ensemble, as it was adapted to the imbalanced dataset. Figure 4.6 illustrates the evolution of the model weights over time for the imbalanced dataset, highlighting how the dynamic weighting system responds to performance variations across the data folds.

Balanced Dataset: A similar trend was observed when the dynamic ensemble approach was applied to the balanced dataset. Initially, the weights were distributed across all the models to reflect a balanced class distribution, allowing for an equitable start. However, as the system processed more data folds, specific models emerged as clear leaders in performance, leading to significant shifts in the weight distribution.

Models such as KNN and Decision Tree quickly gained dominance in the weight distribution, driven by their strong performance on the balanced dataset. These models demonstrate a remarkable ability to exploit a more equitable class distribution, thereby allowing them to achieve high levels of accuracy and generalization. In contrast, SVM and Logistic Regression consistently received lower weights as the dynamic ensemble was adjusted to favor more robust models. This weight adjustment pattern highlights the capacity of the system to prioritize models that are better suited to handling a balanced class distribution.

The adaptability of the dynamic ensemble proved to be particularly advantageous in the balanced dataset scenario, where class distributions were more equitable, and models were expected to handle both classes with a similar level of accuracy. As the ensemble dynamically adjusted its weights, it maintained a high level of performance, as evidenced by the stable upward trend in the accuracy, precision, and F1-scores across the data folds. 4.7 illustrates the evolution of model weights over time for the balanced dataset, showing the ability of the dynamic ensemble to optimize model contributions in response to changing data conditions.

The dynamic weight adjustment mechanism enabled the ensemble to adapt in real time, significantly boosting its overall performance compared with the static ensemble. This adaptability allows the dynamic ensemble to outperform the static ensemble in most cases, particularly on a balanced dataset, where its ability to allocate weights based on performance proved to be a major advantage.

Figure 4.7 illustrates the evolution of model weights over time for the balanced dataset

Comparative Analysis of Evaluation Metrics: Dynamic Ensemble vs. Static Ensemble Performance

A comparative analysis of the evaluation metrics for both the dynamic and static ensembles revealed key insights into the advantages of dynamic weight adjustment. By comparing metrics such as accuracy, precision, recall, and F1-score across multiple data folds, it was possible to observe the impact of dynamic weighting on the ensemble's performance and robustness.

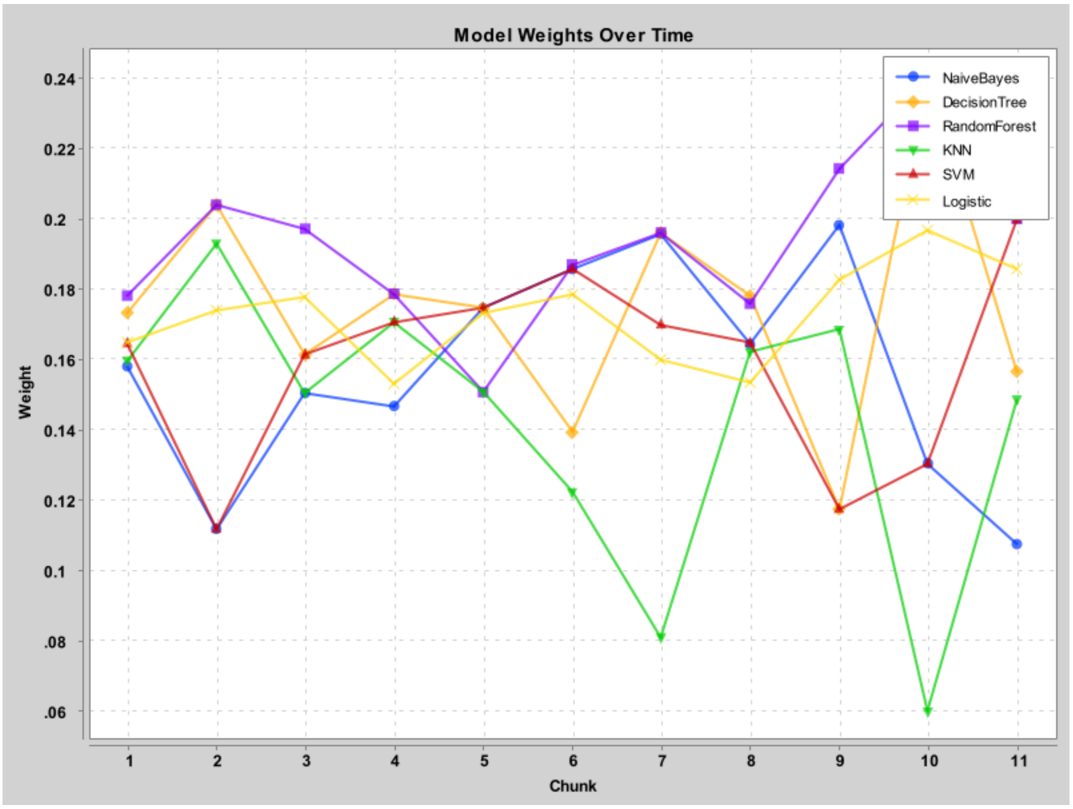

Figure 4.6: Model weights over time for the imbalanced dataset.

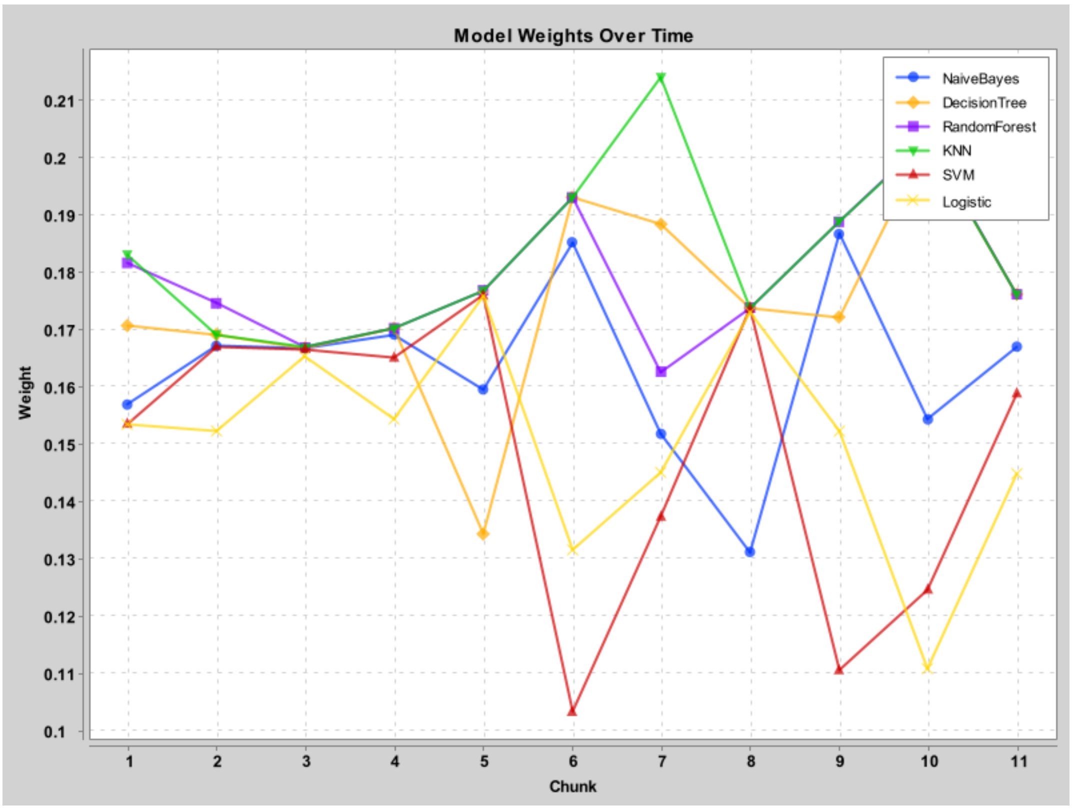

Figure 4.7: Model weights over time for the balanced dataset.

Accuracy: Across the imbalanced and balanced datasets, the dynamic ensemble consistently outperformed the static ensemble, particularly in the later data folds. This trend suggests that the dynamic approach effectively learned from the recent performance and adapted accordingly. On the imbalanced dataset, the static ensemble's accuracy remained relatively stable, but the dynamic ensemble showed a gradual improvement over time as it adjusted the weights to favor better-performing models. On the balanced dataset, the dynamic ensemble demonstrated a stronger upward trend in accuracy, benefiting from its ability to capitalize on the strengths of models such as KNN and Random Forest. Figure 4.8 illustrates the accuracy of both the dynamic and static ensembles over time for both datasets, highlighting the superior performance of the dynamic ensemble.

Precision: The dynamic ensemble also showed superior precision compared to the static ensemble on both the imbalanced and balanced datasets, particularly in later data folds. The ability of the dynamic ensemble to allocate more weight to models that excelled in precision allowed it to fine-tune its predictions and achieve higher levels of precision. This is particularly important in dynamic environments, where the ability to quickly adapt and improve precision is crucial for success. Figure 4.9 demonstrates the precision of both ensembles over time, showing the dynamic ensemble's ability to learn from recent performance and improve its precision over time.

F1-Score: One of the most significant findings of the analysis is the superior performance of the dynamic ensemble in terms of F1-Score on the balanced dataset. The

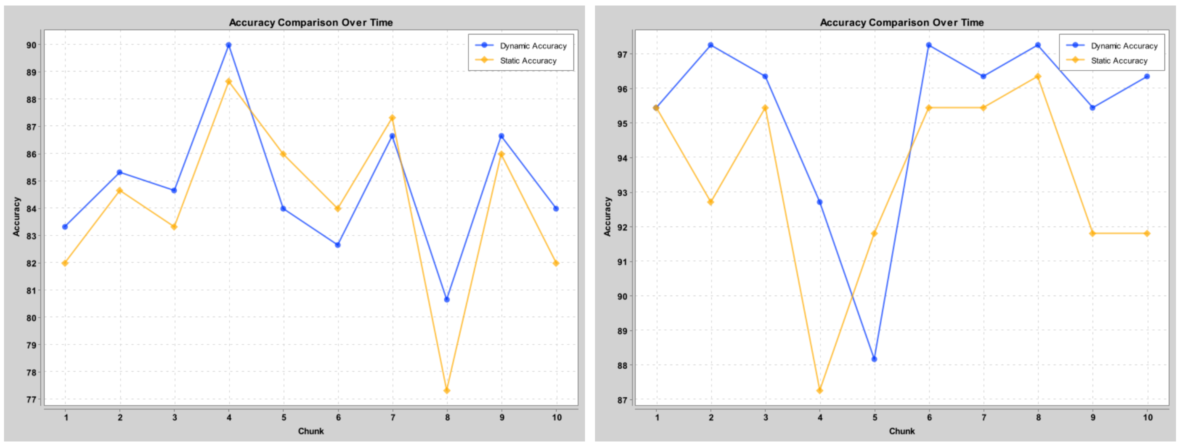

Figure 4.8: Accuracy across different folds, for the imbalanced (left) and balanced (right) datasets.

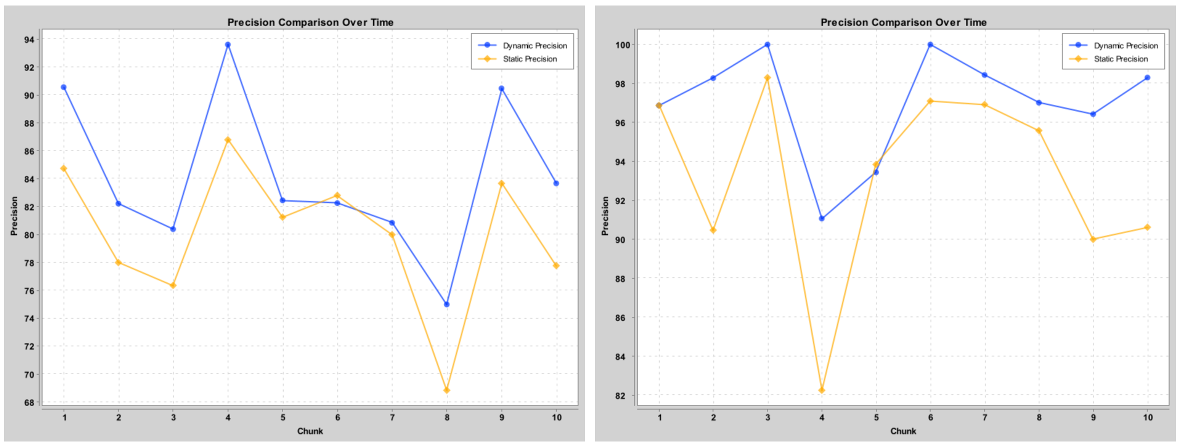

Figure 4.9: Precision across different folds, for the imbalanced (left) and balanced (right) datasets.

dynamic ensemble consistently outperformed the static ensemble across all the data folds, highlighting its ability to maintain a balance between precision and recall. This was particularly evident in the balanced dataset, where the adaptive weighting mechanism of the dynamic ensemble allowed it to achieve higher F1-Scores across all folds. In contrast, the static ensemble struggled to match the dynamic ensemble performance on the imbalanced dataset, where its fixed weighting limited its ability to optimize the balance between precision and recall. Figure 4.10 illustrates the F1-Scores for both ensembles over time, providing a clear comparison of their performance on both datasets.

Through this comparative analysis, it is evident that the ability of the dynamic ensemble to adjust weights based on recent performance and to apply forgetting mechanisms plays a crucial role in enhancing its ability to generalize and adapt to new data. The results suggest that incorporating these mechanisms into an ensemble model can significantly improve its performance on complex and evolving datasets, particularly in environments where data are processed incrementally in folds.

Overall, the dynamic ensemble approach offers a major advantage over static ensembles, particularly in environments characterized by evolving data conditions. The ability

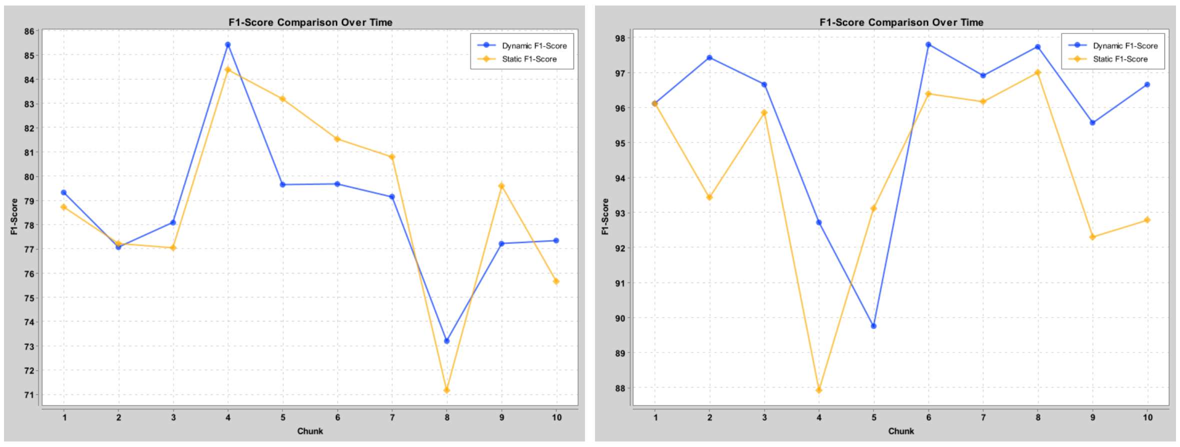

Figure 4.10: F1-Score across different folds, for the imbalanced (left) and balanced (right) datasets.

to adapt to changing data distributions and model performance allows the dynamic ensemble to achieve higher levels of accuracy, precision, recall, and F1-score, making it a more effective tool for real-time decision-making in adaptive machine learning systems.

Although dynamic weight analysis suggests improvements in prediction capability, these observations require statistical validation to confirm their significance. The following section describes the implementation of statistical testing to verify the observed performance differences.

4.2.2 Statistical Test Results

This section implements statistical hypothesis testing to validate the performance improvements observed through the dynamic weight adjustment. The analysis employs the Wilcoxon signed-rank test to evaluate whether the differences between the dynamic and static approaches represent statistically significant improvements in the prediction capability.

The results of statistical tests conducted to compare the performances of the dynamic and static ensemble methods are presented and interpreted. The Wilcoxon signed-rank test was employed to evaluate whether there was a statistically significant difference in the performance metrics (accuracy, precision, recall, and F1-score) across the ten data folds for both imbalanced and balanced datasets.

Hypothesis

- **Null Hypothesis (H_0):** There is no statistically significant difference in performance between the dynamic ensemble and the static ensemble across the evaluated metrics (accuracy, precision, recall, and F1-score).

- **Alternative Hypothesis (H_1):** There is a statistically significant difference in performance between the dynamic ensemble and the static ensemble across the evaluated metrics (accuracy, precision, recall, and F1-score).

These hypotheses aim to determine whether the dynamic ensemble's adaptive weighting mechanism significantly enhances the performance compared to the static ensemble.

Imbalanced Dataset

The Wilcoxon signed-rank test was applied to the performance metrics from the imbalanced dataset, with the following results.

Measure	**Score**	**p-value**
Accuracy	19	0.4316
Precision	3	0.0098
Recall	0	0.0020
F1-Score	23	0.6953

Table 4.4: Results of the performance measures using a significant threshold of 0.05.

- **Accuracy:** The p-value of 0.4316 suggests no statistically significant difference in accuracy between the dynamic and static ensembles. This indicates that both approaches performed similarly in terms of the overall classification accuracy. However, given that the architecture aims to enhance adaptability, the lack of significant improvement in accuracy suggests that, although the dynamic weighting mechanism may adjust model contributions, it does not necessarily enhance the global accuracy metric for imbalanced datasets.

- **Precision:** With a p-value of 0.0098, the difference in precision between the dynamic and static ensembles is statistically significant. The superior precision of the dynamic ensemble indicates its effectiveness in reducing false positives, which is crucial in scenarios where the cost of false alarms is high. This aligns with the objective of developing an architecture that can adaptively prioritize models that contribute to more accurate and reliable predictions.

- **Recall:** The test results show a statistically significant difference in recall (p-value = 0.0020), with the dynamic ensemble outperforming the static ensemble. This suggests that the dynamic ensemble is more effective at correctly identifying instances of the minority class, thereby addressing a common challenge in imbalanced datasets. This finding underscores the ability of the dynamic ensemble to improve the sensitivity of the architecture to true positives, thereby fulfilling the key objectives of this study.

- **F1-Score:** The p-value of 0.6953 indicates no significant difference in F1-scores between the two approaches. Although the dynamic ensemble showed improvements in precision and recall, these gains did not translate into a statistically significant difference in the F1-score, which balances both metrics. This suggests that, although the dynamic ensemble offers targeted improvements, the overall balance of precision and recall may still align with the static approach under certain conditions.

The results for the imbalanced dataset revealed significant differences in precision and recall but no significant differences in accuracy or F1-score. This suggests that while the dynamic ensemble performed better at identifying true positives and reducing false positives, the overall balance of precision and recall (F1-score) was similar to that of the static ensemble.

Balanced Dataset

For the balanced dataset, the Wilcoxon signed-rank test produced the following results.

Measure	Score	p-value
Balanced accuracy	6	0.0502
Precision	1	0.0109
Recall	0	0.3173
F1-Score	6	0.0506

Table 4.5: Results of the performance measures using a significant threshold of 0.05.

- **Accuracy:** The p-value of 0.0502 is on the borderline of statistical significance, suggesting that the dynamic ensemble may have a slight edge in accuracy over the static ensemble. This marginal improvement indicates that the dynamic weighting mechanism can contribute to a better overall performance when class distributions are balanced, aligning with the study's objective of developing an adaptable system that maintains high accuracy across varying data conditions.

- **Precision:** A p-value of 0.0109 confirms a statistically significant difference in precision, with the dynamic ensemble outperforming the static approach. This finding reinforces the ability of the dynamic ensemble to effectively minimize false positives, which is crucial in balanced datasets, where both classes are equally important. This result supports the study's goal of enhancing the prediction specificity through adaptive model contributions.

- **Recall:** The p-value of 0.3173 indicates no significant difference in recall between the two ensembles. Both methods performed similarly in identifying true positives, suggesting that, while the dynamic ensemble improves precision, recall is not compromised. This balance is critical for maintaining the robustness of the overall model in balanced scenarios in line with the objectives of the study.

- **F1-Score:** With a p-value of 0.0506, the F1-score difference is close to statistical significance. This suggests that the dynamic ensemble may balance precision and recall slightly better than the static ensemble, offering a more nuanced improvement in performance. This aligns with the study's aim to achieve a more balanced and adaptable predictive system.

The results for the balanced dataset showed that the precision metric was significantly different, with the dynamic ensemble outperforming the static ensemble. Although the accuracy and F1-score approached statistical significance, they did not meet the threshold, suggesting only marginal improvements in these areas for the dynamic ensemble. Recall showed no significant difference between the two ensembles, indicating similar performance in capturing true positives.

Summary of Statistical Test Results Statistical tests highlighted the strengths of the dynamic ensemble approach within the architecture, particularly in terms of precision and recall. The dynamic ensemble consistently outperformed the static ensemble in terms of precision across both datasets, demonstrating its ability to minimize false positives, which is essential in high-stake decision-making environments. In the imbalanced dataset, the superior recall of the dynamic ensemble further indicated its effectiveness in identifying minority class instances, thereby addressing a key challenge in churn prediction.

However, the lack of statistically significant differences in accuracy and F1-score suggests that the benefits of the dynamic ensemble may be more pronounced in specific metrics than across all performance measures. These findings underscore the potential of the architecture to enhance specific aspects of model performance through dynamic weighting. However, they also highlight the need for further refinement to achieve broader improvements across all metrics. Overall, the adaptive capabilities of the dynamic ensemble offer significant advantages in certain evaluation metrics, contributing to the development of a more effective and adaptable predictive system within the cognitive architecture framework.

Having established the statistical significance of our findings, it is crucial to ensure that these results are replicated and verified by other researchers. The following section details the measures implemented to ensure reproducibility of our experimental framework.

4.3 Model Performance Across Datasets

This section compares the performance of the individual models across three balanced datasets: Churn-in-Telecom, IBM Telco, and Churn-data-UCI. The comparisons included accuracy, precision, recall, F1-score, and ROC-AUC. By examining how each model performs across multiple datasets, this analysis aimed to demonstrate the consistency and adaptability of the model. Table 4.6 presents a summary of the individual model metrics across the three datasets.

Analysis of Model Performance

The results of the table 4.6 analysis of the individual model performance across the datasets revealed the following key insights.

1. **Random Forest, Decision Tree and KNN** emerged as the top performers across all three datasets, particularly for the Churn-in-Telecom dataset, where they achieved the highest accuracy and F1-scores.

    - Random Forest reached its peak performance on the Churn-in-Telecom dataset with an F1-score of 98.01%.

Model	Accuracy	Precision	Recall	F1-Score	ROC-AUC
Churn-in-Telecom Dataset					
Naive Bayes	84.68%	0.8597	0.8468	0.8475	0.8899
Decision Tree	92.11%	0.9211	0.9211	0.9211	0.9361
Random Forest	98.01%	0.9801	0.9801	0.9801	0.9982
KNN	98.82%	0.9883	0.9882	0.9882	0.9892
SVM	82.86%	0.8283	0.8286	0.8279	0.8216
Logistic	82.77%	0.8276	0.8277	0.8267	0.8976
IBM Telco Dataset					
Naive Bayes	89.33%	0.8938	0.8933	0.8934	0.9583
Decision Tree	93.17%	0.9317	0.9317	0.9317	0.9385
Random Forest	94.92%	0.9494	0.9492	0.9491	0.9875
KNN	90.23%	0.9026	0.9023	0.9024	0.9301
SVM	91.76%	0.9176	0.9176	0.9175	0.9169
Logistic	92.21%	0.9221	0.9221	0.9221	0.9773
Churn-data-UCI Dataset					
Naive Bayes	77.73%	0.7758	0.7773	0.7764	0.8501
Decision Tree	85.20%	0.8511	0.8520	0.8509	0.8701
Random Forest	87.60%	0.8763	0.8760	0.8744	0.9307
KNN	78.53%	0.7983	0.7853	0.7714	0.8198
SVM	80.93%	0.8076	0.8093	0.8077	0.7904
Logistic	81.07%	0.8089	0.8107	0.8090	0.8783

Table 4.6: Model performance comparison across different datasets (highlighted cells indicate top two performing models by accuracy for each dataset).

- KNN achieved an F1-score of 98.82% for the same dataset, highlighting its sensitivity to balanced data distributions.

2. **Naive Bayes, Decision Tree, SVM, and Logistic Regression** showed variability in performance across datasets, with Naive Bayes being more effective on the IBM Telco dataset compared to the Churn-data-UCI dataset.

3. **Logistic Regression and SVM** performed better on the **IBM Telco dataset** due to their linear nature, which could better capture the relationships within that dataset.

Dynamic vs. Static Ensemble Performance

Tables 4.7 and 4.8 present the performance metrics of both the dynamic and static ensembles over ten data folds for each dataset. A performance comparison between the static and dynamic approaches revealed dataset-specific patterns.

Fold	Dynamic Ensemble	Static Ensemble
1	94.35%	94.35%
2	**93.79%**	93.22%
3	**96.05%**	95.48%
4	**95.35%**	94.92%
5	**93.79%**	92.09%
6	96.61%	96.61%
7	**91.96%**	90.96%
8	**94.66%**	93.22%
9	**93.53%**	92.09%
10	**92.09%**	90.66%

Table 4.7: Dynamic vs. static ensemble accuracy over time for IBM Telco dataset. Values highlighted in green indicate where the dynamic ensemble achieved higher accuracy compared to the static ensemble.

Fold	Dynamic Ensemble	Static Ensemble
1	**90.33%**	90.10%
2	**94.32%**	92.67%
3	93.56%	94.33%
4	**96.28%**	95.67%
5	**95.45%**	93.00%
6	**93.67%**	90.00%
7	**94.67%**	94.33%
8	**95.20%**	93.33%
9	**94.67%**	91.00%
10	**92.00%**	90.40%

Table 4.8: Dynamic vs. static ensemble accuracy over time for Churn-data-UCI dataset.

4.4 Comparison with Existing Adaptive Ensemble Methods

To better understand the effectiveness of the proposed RTCP system, it was compared with the adaptive ensemble framework described in existing literature [24]. The comparison centers on accuracy across similar datasets used for churn prediction, providing insights into how the RTCP system performs against established adaptive ensemble methods.

Overview of the Adaptive Ensemble Framework

The adaptive ensemble framework described in the referenced study employs a stacking approach with a meta-learner to optimize the contributions of various base models, including XGBoost, LightGBM, LSTM, MLP, and SVM. This method is designed to improve predictive accuracy by dynamically adjusting the weights of individual models, thus enabling the ensemble to effectively adapt to different data distributions. The performance of this adaptive ensemble was evaluated using the IBM Telco, Churn-in-Telecom, and Orange Telecom datasets, demonstrating high accuracy and robust performance across these datasets.

Performance Comparison

Table 4.9 presents the accuracy achieved by both the adaptive ensemble and proposed RTCP system across three different datasets: IBM Telco, Churn-in-Telecom, and Orange Telecom.

Dataset	**Shaikshurab *et al.***	**This Work**
IBM Telco	96.65%	93.62%
Churn-in-Telecom	95.60%	95.92%
Orange Telecom	99.89%	94.52%

Table 4.9: Accuracy comparison between the adaptive ensemble and the proposed RTCP dynamic ensemble across different datasets.

The results indicate that the adaptive ensemble achieves a slightly higher accuracy on the IBM Telco dataset (96.65% vs. 93.62%) and Orange Telecom dataset (99.89% vs. 94.52%). However, the RTCP system demonstrated comparable accuracy on the Churn-in-Telecom dataset (95.92% vs. 95.60%), demonstrating its competitiveness, particularly in handling balanced datasets.

It is important to note that the adaptive ensemble was implemented within a scikit-learn environment that provided optimized implementations of models such as XGBoost, LightGBM, and other gradient-boosting learners. These models often benefit from more efficient algorithms and better integration within Python's machine learning ecosystem. In contrast, the cognitive architecture RTCP system was developed using the Java-based WEKA library, where certain models, especially complex ones, such as XGBoost or LightGBM, are not directly available. This limitation in the Java environment contributes to the marginally lower performance of the RTCP system. The implementation challenges faced in WEKA, which lacks some of the more sophisticated algorithms available in scikit-learn, may explain the performance gap observed for some datasets.

Discussion of Findings

This comparison reveals several key findings.

1. **Accuracy:** The adaptive ensemble achieves higher accuracy on the IBM Telco and Orange Telecom datasets, indicating its potential advantage in handling complex datasets. However, the RTCP system performed competitively on the Churn-in-Telecom dataset, demonstrating its robustness in balanced data scenarios.

2. **Simplified Architecture:** The RTCP system uses a simpler cognitive architecture that dynamically adjusts model weights without the need for a meta-learner. This makes it easier to implement and maintain.

3. **Real-Time Adaptability:** The RTCP system adapts well to evolving data patterns, making it suitable for real-time churn prediction tasks. Dynamic weight adjustments ensure optimal contributions from individual models, allowing the RTCP to adjust more effectively to change data distributions.

Although the adaptive ensemble achieved marginally higher accuracy on some datasets, mainly because of the advantages of the scikit-learn environment, the RTCP system offers a more adaptable solution for real-time churn prediction. Its comparable accuracy and adaptability highlight its potential as an alternative to more complex ensemble methods, particularly when deployed in environments with limited model support. Despite its challenges, the RTCP system demonstrates strong potential for real-time streaming data applications.

4.5 Final Remarks

Integrating dynamic ensemble learning within the cognitive architecture offers significant performance enhancements over static ensemble methods, particularly when handling imbalanced and balanced datasets. The ability of the dynamic ensemble to adaptively adjust model weights based on real-time performance metrics ensures robust and reliable churn prediction, demonstrating the effectiveness of the architecture in optimizing individual model contributions. Furthermore, the adaptability of the architecture to changing data patterns over time underscores its suitability for real-time, evolving environments, fulfilling the study's objectives of developing an effective and adaptable predictive system.

Chapter 5

Conclusion

This study does not focus solely on the performance improvements of dynamic ensemble learning but rather on the potential of integrating a cognitive architecture into ensemble learning. The key contribution of this study lies in harnessing the power of cognitive architectures to improve modularity, real-time adaptability, scalability, and the management of memory objects through codelets features that have not been explored in the domain of supervised learning. This conclusion synthesizes the results, emphasizing the conceptual innovations brought about by the cognitive architecture framework and its broader implications for future machine learning systems.

5.1 Answering Research Questions

With the results reported in the previous chapter, we can now address how these findings specifically answer our research questions. The following section synthesizes the experimental results to provide clear responses to the study's primary research objectives.

Addressing Research Question 1

How does dynamic ensemble learning within the proposed cognitive architecture improve prediction accuracy in real-time churn prediction?

The experimental results demonstrate that dynamic ensemble learning enhances real-time churn prediction performance compared to static ensemble methods on the two datasets. The reported metrics are the averages calculated across all the data folds, ensuring a comprehensive performance evaluation.

- **Dynamic Ensemble:** Achieved an average accuracy of 95.92%, precision of 97.71%, recall of 95.06%, and an F1-Score of 96.37%.
- **Static Ensemble:** Showed an average accuracy of 95.45%, precision of 96.88%, recall of 95.38%, and an F1-Score of 96.12%.

The ability of the dynamic ensemble to dynamically adjust model weights allowed it to leverage high-performing models, such as KNN and Random Forest, more effectively. This resulted in higher accuracy and better generalization on the balanced dataset. The improvement highlights the advantage of dynamic ensemble methods in optimizing model contributions based on real-time performance, thereby enhancing overall predictive accuracy and reliability (Figures 4.6, 4.7, 4.8, 4.9, and 4.10).

Addressing Research Question 2

How effectively does the proposed cognitive architecture adapt to real-time changes in data patterns?

The cognitive architecture demonstrated strong adaptability to changing data patterns over time on balanced datasets, as evidenced by the dynamic ensemble performance metrics and weight distribution analyses. The reported metrics are averaged across all folds, emphasizing the sustained performance and adaptability of the system.

- The dynamic ensemble achieved stable and upward trends in performance metrics, with an average accuracy of 95.92%, precision of 97.71%, recall of 95.06%, and an F1-Score of 96.37%.
- The weight distribution dynamically shifted to favor models excelling under balanced conditions, such as KNN and Random Forest, ensuring responsiveness to evolving data patterns (Figures 4.6, 4.7, 4.8, 4.9, and 4.10).

Statistical test results confirmed the adaptability of the architecture. For both datasets, the dynamic ensemble showed statistically significant improvements in precision and recall compared with the static ensemble (Tables 4.4 and 4.5), highlighting its capability to adapt to model contributions based on recent performance. These findings demonstrate that the proposed cognitive architecture effectively adapts to changing data patterns and optimizes the model weights in real time to sustain and enhance predictive performance.

5.2 Key Findings

The critical insight from this research is not just in achieving higher predictive performance, but also in demonstrating how cognitive architectures provide a flexible and scalable framework for managing dynamic ensemble learning in complex environments. Key findings include:

1. **Modularity and Real-Time Adaptability:** By implementing the dynamic ensemble learning within a cognitive architecture, we introduced a modular approach where individual models act as components that can be dynamically weighted, adjusted, or replaced as the system processes data in real-time. The use of codelets allows specialized processing tasks that adaptively contribute to decision-making. This architecture ensures that the system can prioritize high-performing models while diminishing the influence of underperforming models as new data become available. The emphasis on real-time adaptability was critical because it allowed the system to continually learn and adjust, enhancing its accuracy and resilience to changing data conditions.

2. **Scalability and Flexibility:** The cognitive architecture's modular design inherently supports scalability, allowing the system to grow and evolve as more models, data types, and processing capabilities are introduced. This makes the architecture suitable for a wide range of applications requiring real-time decision making and the ability to integrate new models and data streams without significant reconfiguration. Scalability also extends to managing memory objects, where the architecture can dynamically store and retrieve important information based on the context and task requirements, further enhancing its flexibility.

3. **Memory Objects and Codelets:** The cognitive architecture's use of memory objects and codelets enabled efficient management of information and tasks. Memory objects store essential knowledge from models and processed data, allowing the system to recall and utilize past experiences for better decision-making. Codelets are lightweight, task-oriented processes that handle specific responsibilities such as model weight adjustments, data fold processing, and performance evaluations. This design provides an organized and efficient way of handling complex, evolving datasets, which traditional machine learning approaches may struggle to manage in real time.

4. **Improved Generalization and Robustness:** The cognitive architecture demonstrated superior generalization capabilities through dynamic weight adjustments and real-time processing. Even though performance metrics such as accuracy and F1-score were strong, the key achievement was how the architecture maintained robust decision-making under varying conditions. This adaptability ensures that the system can not only respond to current data patterns but also anticipate and adjust to future trends in data, enhancing its longevity and application across various real-world scenarios.

5.3 Implications of Findings

The findings of this research have broad implications, particularly in the areas of adaptive machine learning systems and real-time decision making.

- **Transforming Ensemble Learning:** By shifting the focus from static models to dynamic, cognitive-driven ensemble systems, this research opens new pathways for enhancing the effectiveness of machine learning systems in environments characterized by constant change. The cognitive architecture's ability to manage modularity, process real-time data, and scale effectively allows for a more robust and adaptive ensemble learning system that can significantly improve machine learning.

- **Applications in Real-Time Systems:** The use of cognitive architectures in this research makes a compelling case for their application in real-time systems where adaptability and modularity are critical, different domains where decisions must be made rapidly and based on evolving data streams, stand to benefit significantly from this type of approach. The capability of the architecture to dynamically adjust model weights based on performance offers a more nuanced system for decision-making tasks.

- **Long-Term Learning and Adaptation:** The integration of memory objects and codelets ensures that the system is responsive to immediate data and capable of long-term learning and adaptation. This continuous learning process allows the architecture to maintain high levels of performance and reliability over time, even as data evolve or drift. Long-term adaptability is essential for systems that operate in unpredictable or rapidly changing environments.
- **Scalable and Modular Systems:** The architecture allows for a scalable and modular approach to machine learning, where new models or data streams can be incorporated seamlessly. This flexibility extends the usefulness of ensemble methods beyond static datasets, allowing for dynamic interactions between models, real-time data processing, and scalable growth as the system complexity increases.

5.4 Broader Applications

The cognitive architecture approach demonstrated in this research has broad potential applications beyond the specific use cases of this study. The cognitive architecture framework can provide a solution that leverages modularity and adaptability in fields that require constant adaptation to changing environments and data, such as real-time analytics and large-scale automated decision-making.

- **Real-Time Financial Analysis:** Financial markets operate in highly dynamic environments where data streams constantly shift. A cognitive architecture can enable real-time adaptation to market conditions, thereby improving the accuracy of predictions related to stock performance, risk assessment, and fraud detection.
- **Healthcare:** In healthcare, where patient data is continuously being updated, cognitive architectures can help build adaptive systems that make real-time predictions and learn from past patient outcomes to improve future diagnostics and treatment recommendations.

5.5 Limitations and Future Research Directions

Despite these promising results, limitations must be acknowledged as they provide an important context for interpreting the findings and offer guidance for future research.

5.5.1 Limitations

One of the primary limitations of this study is the use of cognitive architectures in a domain in which the typical application of reinforcement learning is not the central focus. Cognitive architectures are traditionally better suited for environments where autonomous decision-making and continuous learning are critical, such as in reinforcement learning tasks. However, this research adapted their use for a supervised learning task, specifically customer churn prediction, which required additional design considerations to simulate adaptability and dynamic decision-making. This adaptation, although successful in certain areas, may not fully exploit the capabilities of cognitive architectures in tasks that align more closely with their inherent strengths.

Another key limitation is the reliance on Java-based machine learning libraries in this study. While cognitive architecture frameworks such as the Cognitive System Toolkit (CST) are essential for modular implementation, the use of Java libraries like Weka presented challenges in scalability and efficiency. These limitations are implementation-specific and not inherent to CST as a framework. For instance, Weka lacks support for advanced machine learning algorithms such as gradient boosting, which are more readily available in Python libraries like scikit-learn. This disparity imposed constraints on the optimization and performance of the models. Transitioning to more robust libraries could significantly enhance the scalability and computational efficiency of future implementations.

In addition, the research relied on a static dataset for training and evaluation, which does not fully capture the complexities of real-time customer churn prediction. Although the system was designed to simulate real-time adaptability, the absence of live-streaming datasets introduces a gap between the proposed framework and real-world applications, where continuous data streams and real-time adjustments are critical. This highlights the need for further evaluation of the cognitive architecture in live environments to validate its full potential for real-time decision-making.

5.5.2 Transitioning to a Fully Real-Time System

Acknowledging Simulation Limitations

This study simulates real-time prediction by processing static datasets incrementally to mimic the behavior of a real-time system. While this approach provides valuable insights into the potential adaptability and scalability of the proposed framework, it does not fully replicate the complexities of a true real-time environment. Specifically, the system lacks live-streaming data integration, real-time feedback mechanisms, and infrastructure optimized for continuous data processing. These limitations highlight the need for further development to transition the framework into a fully operational real-time system.

Roadmap for Transitioning to Real-Time Prediction

To address these limitations, the following steps are proposed for future development:

1. **Integration of Live-Streaming Datasets**
    - Replace static datasets with live data streams from relevant sources, such as customer transaction logs, real-time user interactions, or IoT sensors.
    - Incorporate tools and platforms like Apache Kafka or AWS Kinesis to manage and process continuous data streams efficiently.

2. **Implementation of Real-Time Feedback Mechanisms**

   - Develop mechanisms to capture real-time system performance metrics, such as latency, accuracy, and response time, to enable immediate system adjustments.
   - Introduce adaptive learning techniques that update model weights dynamically based on feedback from recent predictions and actual outcomes.

3. **Dynamic Data Preprocessing**

   - Implement automated data preprocessing pipelines that operate in real time, addressing issues such as missing data, outliers, and class imbalances on the fly.

4. **Validation in Real-World Scenarios**

   - Test the fully real-time system in real-world environments to assess its performance under realistic conditions, such as diverse customer behaviors.
   - Collaborate with industry partners to deploy the system in operational settings and gather feedback for iterative improvement.

5.5.3 Expected Benefits

Transitioning to a fully real-time system would enable the proposed framework to handle live, continuous data streams, making it more effective in addressing real-world challenges in churn prediction. Real-time adaptability and feedback mechanisms would enhance the system's responsiveness, while infrastructure upgrades would ensure scalability and reliability in dynamic environments.

5.5.4 Future Research Directions

Future work could explore integrating CST with advanced machine learning libraries in Python to overcome the scalability and efficiency limitations observed in this study's Java-based implementation. By leveraging Python's optimized libraries, such as scikit-learn and TensorFlow, the system's performance and scalability could be significantly improved. Additionally, transitioning to live-streaming datasets would provide a more realistic assessment of the framework's real-time adaptability and its capacity to handle continuously evolving data.

Another promising direction would involve expanding the cognitive architecture's functionality to include reinforcement learning techniques. This would allow the system to capitalize on the natural strengths of cognitive architectures in autonomous decision-making and dynamic learning environments. Furthermore, future studies could investigate integrating additional variables, such as external economic indicators, into the churn prediction model to improve its predictive accuracy and broader applicability.

Finally, exploring alternative cognitive architecture toolkits that are designed specifically for real-time data handling could enhance the modularity and adaptability of future implementations. These efforts would further validate the framework's potential to address the challenges of real-time customer churn prediction in dynamic and complex environments.

In conclusion, although this study makes meaningful contributions to real-time churn prediction by integrating cognitive architectures with ensemble learning, these limitations highlight areas where future research can build upon these foundations to achieve more adaptive, scalable, and effective solutions. By addressing these constraints, future studies can further refine the application of cognitive architectures in dynamic, real-time environments, and push the boundaries of customer churn prediction.

5.6 Final Thoughts

This study presented an ensemble learning method by introducing a new approach to adaptive machine learning systems. By leveraging the strengths of cognitive architectures, such as modularity, real-time adaptability, scalability, and memory management, our approach offers significant advancements in the design, implementation, and deployment of machine learning systems in a churn prediction context. Integrating cognitive principles into machine learning frameworks is a critical step forward in developing more intelligent, responsive, and effective learning systems with the potential for widespread applications in real-world environments where data are constantly evolving. This investigation laid the foundation for future research on cognitive architectures in supervised learning, presenting exciting opportunities to further enhance the adaptability and performance of machine learning systems in various complex and dynamic domains.

Bibliography

[1] V. Nagesh, "Customer churn prediction," *International Journal of Scientific Research in Engineering and Management*, vol. 7, no. 12, pp. 1–6, December 2023.

[2] A. H. M, B. T, S. Tanisha, S. B, and C. C. Shanuja, "Customer churn prediction using synthetic minority oversampling technique," in *2023 4th International Conference on Communication, Computing and Industry 6.0 (C2I6)*. IEEE, December 2023, pp. 01–05.

[3] B. Huang, M. T. Kechadi, and B. Buckley, "Customer churn prediction in telecommunications," *Expert Systems with Applications*, vol. 39, no. 1, pp. 1414–1425, 2012.

[4] M. Saghir, Z. Bibi, S. Bashir, and F. Khan, "Churn prediction using neural network based individual and ensemble models," in *2019 16th International Bhurban Conference on Applied Sciences and Technology (IBCAST)*. IEEE, 2019, pp. 634–639.

[5] S. K. Wagh, A. A. Andhale, K. S. Wagh, J. R. Pansare, S. P. Ambadekar, and S. Gawande, "Customer churn prediction in telecom sector using machine learning techniques," *Results in Control and Optimization*, vol. 14, p. 100342, March 2024.

[6] N. Tamuka and K. Sibanda, "Real time customer churn scoring model for the telecommunications industry," in *2020 2nd International Multidisciplinary Information Technology and Engineering Conference (IMITEC)*. IEEE, 2020, pp. 1–9.

[7] A. Chakraborty, V. Raturi, and S. Harsola, "Bbe-lswcm: A bootstrapped ensemble of long and short window clickstream models," in *Proceedings of the 7th Joint International Conference on Data Science & Management of Data (11th ACM IKDD CODS and 29th COMAD)*, 2024, pp. 350–358.

[8] C. Lebiere, E. A. Cranford, M. Martin, D. F. Morrison, and A. Stocco, "Cognitive architectures and their applications," in *2022 IEEE 8th International Conference on Collaboration and Internet Computing (CIC)*. IEEE, December 2022, pp. 55–60.

[9] R. Wolniak, "Analyzing customer behavior—employing business analytics within industry 4.0 ecosystems," *Scientific Papers of Silesian University of Technology*, vol. 195, no. 39, June 2024.

[10] S. S. Gokulnath and M. Revathi, "Leveraging data analytics in azure for effective churn management," in *2024 Second International Conference on Emerging Trends in Information Technology and Engineering (ICETITE)*, February 2024.

[11] Z. Hasan and D. Stablichenkova, “Design of long short term memory based deep learning model for customer churn prediction in business intelligence,” *International Journal of Advances in Applied Computational Intelligence (IJAACI)*, vol. 5, no. 1, pp. 56–64, 2024.

[12] V. Nguyen, A. Fermanian, A. Guilloux, and A. Barbieri, “Flash: a fast joint model for longitudinal and survival data in high dimension,” *arXiv preprint arXiv:2309.03714*, 2023, proposes a model for real-time churn prediction using longitudinal features.

[13] M. Mallegowda and R. Sanjana, “A comparative analysis of serial and parallel data mining approaches for customer churn prediction in telecom,” *International Research Journal on Advanced Science Hub*, 2023, compares data mining approaches for real-time churn prediction.

[14] L. Nalla and V. Reddy, “Machine learning and predictive analytics in e-commerce: A data-driven approach,” *Journal of E-commerce Studies and Technology*, 2024.

[15] A. Soni, J. Mishra, and M. Dixit, “Comparative study of bank customers churn prediction using ai/ml,” in *2024 IEEE 13th International Conference on Communication Systems and Network Technologies (CSNT)*. IEEE, 2024, pp. 1359–1365.

[16] N. Alboukaey, A. Joukhadar, and N. Ghneim, “Dynamic behavior based churn prediction in mobile telecom,” *Expert Systems with Applications*, vol. 162, p. 113779, 2020.

[17] A. Chakraborty, V. Raturi, and S. Harsola, “Ricon: A ml framework for real-time and proactive intervention to prevent customer churn,” *arXiv preprint arXiv:2203.16155*, 2022.

[18] X. Zhu, J. Li, J. Ren, J. Wang, and G. Wang, “Dynamic ensemble learning for multi-label classification,” *Information Sciences*, 2021. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0020025521001980

[19] D. Seo and Y. Yoo, “Improving shopping mall revenue by real-time customized digital coupon issuance,” *IEEE Access*, vol. 11, pp. 7924–7932, 2023.

[20] A. L. Paraense, K. Raizer, S. M. de Paula, E. Rohmer, and R. R. Gudwin, “The cognitive systems toolkit and the cst reference cognitive architecture,” *Biologically Inspired Cognitive Architectures*, vol. 17, pp. 32–48, 2016.

[21] A. L. O. Paraense, K. Raizer, and R. R. Gudwin, “A machine consciousness approach to urban traffic control,” *Biologically Inspired Cognitive Architectures*, vol. 15, pp. 61–73, 2016.

[22] F. Grassiotto, E. Colombini, A. d. S. Simões, and R. R. Gudwin, “Cogtom-cst: An implementation of the theory of mind for the cognitive systems toolkit,” in *Proceedings of the 14th International Conference on Agents and Artificial Intelligence (ICAART)*. Setubal, Portugal: SCITEPRESS, January 2022, pp. 462–469. [Online]. Available: http://hdl.handle.net/11449/237536

[23] J. Vallverdú, M. Talanov, S. Distefano, M. Mazzara, A. Tchitchigin, and I. Nurgaliev, "A cognitive architecture for the implementation of emotions in computing systems," *Biologically Inspired Cognitive Architectures*, vol. 15, pp. 34–40, 2016.

[24] M. A. Shaikhsurab and P. Magadum, "Enhancing customer churn prediction in telecommunications: An adaptive ensemble learning approach," *arXiv preprint arXiv:2408.16284*, 2024.

[25] B. Krawczyk, L. L. Minku, J. Gama, J. Stefanowski, and M. Woźniak, "Ensemble learning for data stream analysis: A survey," *Information Fusion*, vol. 37, pp. 132–156, 2017. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1566253516302025

[26] L. Yang, D. Li, and Y. Lu, "Prediction modeling and analysis for telecom customer churn in two months," *arXiv preprint*, vol. cs.IR, p. 22, 2019. [Online]. Available: https://arxiv.org/abs/1911.00558

[27] V. Umayaparvathi and K. Iyakutti, "Applications of data mining techniques in telecom churn prediction," *International Journal of Computer Applications*, vol. 42, no. 20, pp. 5–9, March 2012.

[28] J. Rabbah, M. Ridouani, and L. Hassouni, "A new churn prediction model based on deep insight features transformation for convolution neural network architecture and stacknet," *International Journal of Web-Based Learning and Teaching Technologies (IJWLTT)*, vol. 17, no. 1, pp. 1–18, 2022.

[29] S. Çelik and S. T. Tayalı, "Resampling and ensemble strategies for churn prediction," *Bilişim Teknolojileri Dergisi*, vol. 16, no. 4, pp. 263–273, 2023.

[30] F. Merchie and D. Ernst, "Churn prediction in online gambling," *arXiv preprint arXiv:2201.02463*, 2022, 14 pages, 3 figures. Submitted to Expert Systems with Applications. [Online]. Available: https://arxiv.org/abs/2201.02463

[31] N. Mustafa, L. S. Ling, and S. F. Abdul Razak, "Customer churn prediction for telecommunication industry: A malaysian case study," *F1000Research*, vol. 10, p. 1274, 2021, version 1; peer review: 1 approved. [Online]. Available: https://doi.org/10.12688/f1000research.73597.1

[32] S. Youngjung, "Machine learning based customer churn prediction in home rental business," *Journal of Big Data*, vol. 10, no. 1, p. 41, 2023. [Online]. Available: https://doi.org/10.1186/s40537-023-00721-8

[33] F. E. Usman-Hamza, A. O. Balogun, R. T. Amosa, L. F. Capretz, H. A. Mojeed, S. A. Salihu, A. G. Akintola, and M. A. Mabayoje, "Sampling-based novel heterogeneous multi-layer stacking ensemble method for telecom customer churn prediction," *Scientific African*, vol. 24, p. e02223, 2024, this is an open access article under the CC BY-NC-ND license. Available online 3 May 2024. [Online]. Available: http://creativecommons.org/licenses/by-nc-nd/4.0/

[34] J. Rabbah, M. Ridouani, and L. Hassouni, “New approach to telecom churn prediction based on transformers,” in *The 3rd International Conference on Artificial Intelligence and Computer Vision (AICV2023).* Springer, 2023, pp. 565–574, first Online: March 1, 2023.

[35] J. C. Turiho, W. Cheruiyot, A. Kibe, and I. Mungwarakarama, “Survey of data mining techniques used for real time churn prediction,” *International Journal of Advanced Research in Computer Science and Software Engineering*, vol. 7, no. 3, pp. 219–222, March 2017.

[36] L. Almuqren *et al.*, “An empirical study on customer churn behaviours prediction using arabic twitter mining approach,” *Future Internet*, vol. 13, no. 7, p. 175, 2021. [Online]. Available: https://www.mdpi.com/1999-5903/13/7/175

[37] M. Almeida, M. Z. Mota, W. F. de Souza, M. Nicolau, E. J. S. Luz, and G. Moreira, “A temporal approach to customer churn prediction: A case study for financial services,” in *Proceedings*, November 27 2022.

[38] E. G. Castro and M. S. Tsuzuki, “Churn prediction in online games using players’ login records: A frequency analysis approach,” *IEEE Transactions on Computational Intelligence and AI in Games*, vol. 7, no. 3, pp. 255–265, 2015.

[39] P. Bertens, A. Guitart, and Á. Periáñez, “Games and big data: A scalable multi-dimensional churn prediction model,” in *2017 IEEE conference on computational intelligence and games (CIG).* IEEE, 2017, pp. 33–36.

[40] S. Renjith, “B2c e-commerce customer churn management: Churn detection using support vector machine and personalized retention using hybrid recommendations,” *International Journal on Future Revolution in Computer Science & Communication Engineering (IJFRCSCE)*, vol. 3, no. 11, pp. 34–39, 2017.

[41] P. Lalwani, M. K. Mishra, J. S. Chadha, and P. Sethi, “Customer churn prediction system: A machine learning approach,” *Computing*, vol. 104, pp. 271–294, 2021.

[42] S. Tamuka, A. Bhattacharjee, and S. Mitra, “Dynamic churn prediction in online social platforms,” *Information Sciences*, vol. 543, pp. 135–151, 2020.

[43] R. Almeida, A. Ferreira, and L. Santos, “A machine learning framework for customer churn prediction in cloud-based services,” *Expert Systems with Applications*, vol. 168, pp. 114–121, 2021.

[44] D. M. M, S. R, and S. Ramineni, “A comparative analysis of serial and parallel data mining approaches for customer churn prediction in telecom,” *International Research Journal on Advanced Science Hub*, vol. 5, no. 12, pp. 413–419, 2023.

[45] O. J. Ogbonna, G. I. Aimufua, M. U. Abdullahi, and S. Abubakar, “Churn prediction in telecommunication industry: A comparative analysis of boosting algorithms,” *Dutse Journal of Pure and Applied Sciences*, 2024.

[46] A. Sussman and J. B. Hollander, *Cognitive Architecture: Designing for How We Respond to the Built Environment.* Elsevier, 2014.

[47] S. Hélie and R. Sun, "Cognitive architectures and agents," in *Springer Handbook of Computational Intelligence.* Springer, 2015, pp. 683–696.

[48] J. F. Sowa, "Cognitive architectures for conceptual structures," in *International Conference on Conceptual Structures.* Springer Berlin Heidelberg, 2011, pp. 35–49.

[49] R. Gudwin, A. Paraense, S. M. de Paula, E. Froes, W. Gibaut, E. Castro, V. Figueiredo, and K. Raizer, "The multipurpose enhanced cognitive architecture (meca)," *Biologically Inspired Cognitive Architectures*, vol. 22, pp. 20–34, 2017.

[50] J. R. Anderson, D. Bothell, M. D. Byrne, S. Douglass, C. Lebiere, and Y. Qin, "An integrated theory of the mind," *Psychological Review*, vol. 111, pp. 1036–1060, 2004.

[51] J. E. Laird, "Extending the soar cognitive architecture," in *Proceedings of the Artificial General Intelligence Conference.* Memphis, TN: IOS Press, 2008.

[52] P. Langley, K. Cummings, and D. Shapiro, "Hierarchical skills and cognitive architectures," in *Proceedings of the Twenty-Sixth Annual Conference of the Cognitive Science Society*, Chicago, IL, 2004, pp. 779–784.

[53] G. Morais, I. Loron, L. F. Coletta, A. A. Da Silva, A. Simões, R. Gudwin, P. D. P. Costa, and E. Colombini, "Cst-godot: Bridging the gap between game engines and cognitive agents," in *2022 21st Brazilian Symposium on Computer Games and Digital Entertainment (SBGames).* IEEE, 2022, pp. 1–6.

[54] F. Grassiotto, E. Colombini, A. d. S. Simões, and R. R. Gudwin, "Cogtom-cst: An implementation of the theory of mind for the cognitive systems toolkit," in *Proceedings of the 14th International Conference on Agents and Artificial Intelligence.* SCITEPRESS, January 2022, pp. 462–469.

[55] J. K. Kalita, D. K. Bhattacharyya, and S. Roy, *Fundamentals of Data Science: Theory and Practice*, 1st ed. Elsevier, November 2023.

[56] D. Sharma, A. Garg, and A. Kumar, "Ensemble learning in machine learning: Integrating multiple models for improved predictions," *International Journal of Applied Research*, vol. 4, no. 7, pp. 61–65, July 2018. [Online]. Available: https://www.allresearchjournal.com

[57] F. Huang, G. Xie, and R. Xiao, "Research on ensemble learning," in *2009 International Conference on Artificial Intelligence and Computational Intelligence (AICI)*, vol. 3. IEEE Computer Society, 2009, pp. 249–252.

[58] P. Gurram and H. Kwon, "Ensemble learning based on multiple kernel learning for hyperspectral chemical plume detection," pp. 649–659, 2010.

[59] M. P. Sesmero, A. I. Ledezma, and A. Sanchis, "Generating ensembles of heterogeneous classifiers using stacked generalization," *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, vol. 5, no. 1, pp. 21–34, 2015.

[60] P.-N. Tan, M. Steinbach, and V. Kumar, *Introduction to Data Mining*, 1st ed. New Delhi, India: Pearson Education India, 2016.

[61] R. Polikar, "Ensemble based systems in decision making," *IEEE Circuits and Systems Magazine*, vol. 6, no. 3, pp. 21–45, 2006.

[62] X. Wang, K. Nguyen, and B. P. Nguyen, "Churn prediction using ensemble learning," in *Proceedings of the 4th International Conference on Machine Learning and Soft Computing*, 2020, pp. 56–60.

[63] D. H. Wolpert, "Stacked generalization," *Neural Networks*, vol. 5, no. 2, pp. 241–259, 1992.

[64] K. Kim and E. B. Bartlett, "Error estimation by series association for neural network systems," *Neural Computation*, vol. 7, no. 4, pp. 799–808, 1995.

[65] L. Todorovski, H. Blockeel, and S. Dzeroski, "Ranking with predictive clustering trees," in *Proceedings of the European Conference on Machine Learning.* Springer, 2002, pp. 444–455.

[66] I. H. Witten and E. Frank, "Data mining: practical machine learning tools and techniques with java implementations," *ACM SIGMOD Record*, vol. 31, no. 1, pp. 76–77, 2002.

[67] P. L. Bras, F. Sévellec, P. Tandeo, J. Ruiz, and P. Ailliot, "Selecting and weighting dynamical models using data-driven approaches," *Nonlinear Processes in Geophysics*, 2024.

[68] A. Catto, N. Jiang, A. Salleb-Aouissi, and A. Raja, "M-dew: Extending dynamic ensemble weighting to handle missing values," 2024.

[69] X. Zhang, Y. Zhou, Z. Lin, and Y. Wang, "Ensemble learning with dynamic weighting for response modeling in direct marketing," *Electronic Commerce Research and Applications*, vol. 64, p. 101371, 2024.

[70] H. Chen, T. Wang, Y. Zhang, Y. Bai, and X. Chen, "Dynamically weighted ensemble of geoscientific models via automated machine-learning-based classification," *Geoscientific Model Development*, vol. 16, no. 19, pp. 5685–5701, 2023.

[71] I. AlShourbaji, N. Helian, Y. Sun, A. G. Hussien, L. Abualigah, and B. Elnaim, "An efficient churn prediction model using gradient boosting machine and metaheuristic optimization." *Scientific Reports*, vol. 13, no. 1, p. 14441, 2023.

[72] G. Cleotilde, "Building human-like artificial agents: A general cognitive algorithm for emulating human decision-making in dynamic environments." *Perspectives on Psychological Science*, vol. 19, no. 5, pp. 860–873, 2024.

[73] N. Rane, S. P. Choudhary, and J. Rane, "Ensemble deep learning and machine learning: applications, opportunities, challenges, and future directions," *Studies in Medical and Health Sciences*, vol. 1, no. 2, pp. 18–41, 2024.

[74] S. Moro, R. Laureano, and P. Cortez, "Using data mining for bank direct marketing: An application of the crisp-dm methodology," in *Proceedings of the European Simulation and Modelling Conference - ESM'2011*, P. N. et al., Ed. Guimaraes, Portugal: EUROSIS, Oct. 2011, pp. 117–121.