

**DESIGN OF WEB PERSONALIZATION
TECHNIQUES
USING SOFT COMPUTING**

A Thesis Submitted
in the Partial Fulfillment of the Requirements
for the Degree of

DOCTOR OF PHILOSOPHY

by

Nipun Bansal

(Roll No. 2K18/PhDIT/09)

Under the Supervision of

Dr. Manju Bala

Associate Professor

Department of Computer Science

I.P. College for Women, University of Delhi

Delhi, India

Dr. Kapil Sharma

Professor

Department of Information Technology

Delhi Technological University

Delhi, India



to the

DEPARTMENT OF INFORMATION TECHNOLOGY

DELHI TECHNOLOGICAL UNIVERSITY

Delhi-110042, India

August, 2024

CERTIFICATE

It is certified that the work contained in this thesis entitled “*Design of Web Personalization Techniques using Soft Computing*”, by **Nipun Bansal**, has been carried out under our supervision and that this work has not been submitted elsewhere for a degree. The thesis embodies results of original work, and studies are carried out by the student himself and the contents of the thesis do not form the basis for the award of any other degree to the candidate or to anybody else from this or any other University/Institution.



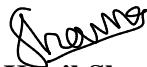
Dr. Manju Bala

Associate Professor

Department of Computer Science

I.P. College for Women, University of Delhi

Delhi, India



Dr. Kapil Sharma

Professor

Department of Information Technology

Delhi Technological University

Delhi, India

CERTIFICATE BY STUDENT

I, **NIPUN BANSAL**, hereby certify that the work which is being presented in the thesis entitled “*Design of Web Personalization Techniques using Soft Computing*” in partial fulfillment of the requirements for the award of the Degree of Doctor of Philosophy, submitted in the Department of Information Technology, Delhi Technological University is an authentic record of my own work carried out during the period from August 2018 to June 2024 under the supervision of **Dr. Kapil Sharma**, Professor of Information Technology Department, Delhi Technological University, Delhi, India and **Dr. Manju Bala**, Associate Professor at I.P. College for Women, University of Delhi, Delhi, India.

The matter presented in the thesis has not been submitted by me for the award of any other degree of this or any other Institute.

A rectangular box containing a handwritten signature in black ink that reads "Nipun".

Candidate's Signature

ACKNOWLEDGEMENTS

In servitude to God, I express my deepest gratitude for His almighty grace, which has guided and supported me throughout the course of this thesis.

First and foremost, I am sincerely grateful to my advisors, **Prof. Kapil Sharma** and **Dr. Manju Bala**, for their invaluable guidance, support, and encouragement. Their expertise and insights have been instrumental in shaping this research and bringing it to fruition.

I am deeply indebted to my mother, **Dr. Raj Rani Bansal**, and my father, **Sh. Vinod Kumar Bansal**, for their unwavering love and belief in me throughout this PhD journey. None of this would have been possible without their unconditional love and trust. Many thanks to my sister, **Dr. Megha Bansal**, for her constant support, understanding, and prayers. A special mention goes to my little niece, **Aadya Goel**, who brings so much joy into my life. I can never thank you all enough; your belief in me has been a profound source of strength and motivation.

I would also like to express my heartfelt gratitude to the **Hon'ble Vice Chancellor** of Delhi Technological University and the **Head of the Department** of Information Technology, Delhi Technological University, for their unwavering support and encouragement.

Special thanks go to my friends **Dr. Yashasvi Bansal** and **Ashish Madaan** for their insightful comments and stimulating discussions. Their camaraderie and encouragement have made this journey more enjoyable and fulfilling.

Thank you all for being a part of this journey.

Nipun Bansal

August, 2024

List of Publications

Journals

1. Nipun Bansal, Manju Bala, and Kapil Sharma, “FuzzyBandit: An autonomous personalized model based on contextual multi-arm bandits using Explainable AI,” *Defence Science Journal*, (2024). 74. [doi:496-504. 10.14429/dsj.74.19330](https://doi.org/10.14429/dsj.74.19330).
2. Nipun Bansal, Manju Bala, and Kapil Sharma, “Hybrid-Neuro Bandit: A bandit model for online recommendation,” *Defence Science Journal*, (2024).[Accepted]

International Conferences

1. Nipun Bansal, Manju Bala, and Kapil Sharma, “A systematic review on Web Personalization recommendation system,” in the *7th International Joint Conference on Computing Sciences (ICCS-2023)*.
2. Nipun Bansal, Manju Bala, and Kapil Sharma, “Web Personalization with Large Language Models: Challenges and Future Trends,” in the *Proceedings of Fourth International Conference on Paradigms of Communication, Computing and Data Analytics*.



वक्रतुण्ड महाकाय सूर्यकोटि समप्रभ ।
निर्विघ्नं कुरु मे देव सर्वकार्येषु सर्वदा ॥

Abstract

The advancement of personalized recommendation systems is essential in today's digital landscape, where user-specific content and services play a pivotal role in enhancing the user experience. However, these systems face significant challenges, particularly in balancing the trade-off between exploration and exploitation of user preferences and ensuring explainability in decision-making algorithms. This thesis addresses these challenges by proposing several novel solutions, namely FuzzyBandit, Hybrid-Neuro Bandit, and Contextual-POI-Bandit, aimed at improving web personalization.

The FuzzyBandit model optimizes decision-making through a dynamic feedback mechanism. This model adjusts parameters based on the relevance and diversity of features to maximize rewards while maintaining interpretability by generating explanations for its decisions. A novel trust score framework is also developed, assessing the reliability of the model's outputs, thus enhancing user trust and enabling the detection of potential errors in the system's reasoning.

Further advancing the field, this research presents the Hybrid Neuro Bandit (HNB) model, which integrates the most effective expert advice from existing recommendation models while discarding those that underperform. This approach demonstrates advantages across various datasets and scenarios, addressing the variability in model performance. Additionally, the Contextual-POI-Bandit framework is introduced, which integrates Social, Geographical, Temporal, and Categorical (SGTC) contextual influences into a unified user vector. This framework significantly enhances the predictive accuracy of personalized Point of Interest (POI) recommendations by closely analyzing user behavior and anticipating future visits.

Furthermore, all the solutions have been empirically evaluated on benchmark datasets, comparing them with state-of-the-art models across performance metrics such as recall, precision, and accuracy, offering substantial improvements in the effectiveness, reliability, and interpretability of the developed personalized recommendation models.

0. List of Publications

Contents

List of Publications	v
List of Figures	xi
List of Tables	xiii
1 Introduction	1
1.1 General	1
1.2 Background	5
1.3 Research Gaps	8
1.4 Research Objectives	9
1.5 Contributions	10
1.6 Thesis Organization	13
2 Literature Review: Personalization in Web Applications using Advanced Machine Learning Models	15
2.1 Introduction	15
2.2 Web Personalization using Advanced Machine Learning (ML) Models . .	17
2.2.1 Neural Network Models	17
2.2.2 Web Personalization through Large Language Models (LLMs) and its Challenges	18
2.2.3 Web Personalization using Contextual Multi-Arm Bandits (CMAB) Models	22
2.3 Why CMAB for Personalization?	24
3 Explainable AI based FuzzyBandit Model using Contextual Bandits	27
3.1 Introduction	27
3.2 Explainable AI (XAI) based FuzzyBandit Model	28

CONTENTS

3.2.1	FuzzyBandit Ranking	28
3.2.2	XAI	32
3.3	Results and Discussion	34
3.4	Conclusion	44
4	Hybrid-Neuro Bandit Model	47
4.1	Introduction	47
4.2	Development of Hybrid-Neuro Bandit (HNB) Model	48
4.2.1	HNB Model	48
4.2.2	HNB Ranking Algorithm	52
4.3	Results and Discussion	53
4.3.1	Experimental Results	54
4.4	Conclusions	59
5	Contextual-POI-Bandit: A Contextual Multi-Arm based Framework for next-Points-of-Interest (POI) Recommendation	61
5.1	Introduction	61
5.1.1	Key Factors for Next POI Recommendation Model	62
5.1.2	Overview of Contextual-POI-Bandit	66
5.2	Development of Contextual-POI-Bandit Framework	68
5.2.1	Mathematical Modelling	68
5.2.2	Definitions	70
5.2.3	Proposed Methodology	72
5.3	Datasets and Evaluation Metrics	86
5.3.1	Datasets	86
5.3.2	Evaluation Metrics	88
5.4	Results and Discussion	90
5.5	Conclusions	99
6	Conclusion and Future Scope	103
6.1	Introduction	103
6.2	Future Scope	106
	References	107

List of Figures

1.1	Exploring potential layout of music app Wynk	3
3.1	FuzzyBandit model	29
3.2	FuzzyBandit ranking algorithm	31
3.3	XAI model for FuzzyBandit	32
3.4	Comparison of contextual bandit models on adult dataset	39
3.5	Comparison of contextual bandit models on covertime dataset	41
3.6	Comparison of contextual bandit models on mushroom dataset	43
3.7	Comparison of contextual bandit models on statlog dataset	44
4.1	HNB model	48
4.2	Instance of weight matrix updation in HNB model	50
4.3	HNB ranking algorithm	52
4.4	Performance metric analysis of contextual multi-arm bandit algorithms on mushroom and adult datasets	55
4.5	MCC index	58
5.1	Trapezoidal membership function	75
5.2	FNN model	76
5.3	Performance analysis of Contextual-POI-Bandit vs other models for Gowalla dataset	92
5.4	GeoSoCa algorithm performance for different permutations of SGTC con- text on Yelp Dataset	94
5.5	FCFKDEAMC algorithm performance for different permutations of SGTC context on Yelp dataset	95
5.6	PFMMGM algorithm performance for different permutations of SGTC context on Yelp dataset	96

LIST OF FIGURES

5.7	GeoSoCa algorithm performance for different permutations of SGT context on Gowalla dataset	97
5.8	FCFKDEAMC algorithm performance for different permutations of SGT context on Gowalla dataset	98
5.9	PFMMGM algorithm performance for different permutations of SGT context on Gowalla dataset	99

List of Tables

2.1	Comparison of LLMs and CMABs algorithms for personalization	25
3.1	Performance analysis of contextual bandit models on the mushroom database	35
3.2	Performance analysis of contextual bandit models on the adult database . .	37
4.1	Description of datasets used by HNB model	53
4.2	Confusion matrix	54
5.1	Description of datasets for next POI recommendation. $ U $: number of users, $ S $: number of social links, $ C $: number of categories, $ \text{checkins} $: number of check-ins, $ \text{POIs} $: number of POIs.	88
5.2	Data for recommended POIs and actual POIs visited by user	89
5.3	Confusion matrix	90
5.4	Empirical evaluation of Contextual-POI-Bandit for New York, Tokyo, and Gowalla datasets	91

LIST OF TABLES

Nomenclature

ANFIS	Adaptive Neuro-Fuzzy Inference System	EXP-4	Exponential-weight algorithm for Exploration and Exploitation using Expert advice
AR	Augmented Reality	FB-ATU	FuzzyBandit Arm Tuning Unit
AutoML	Automated Machine Learning	FB-U	FuzzyBandit User
BLBF	Bandit Feedback	FBA	FuzzyBandit Arm
C	Categorical	FM	Fowlkes–Mallows
CB	Contextual Bandit	FNN	Fuzzy Neural Network
CMAB	Contextual Multi-Arm Bandits	G	Geographical
CRM	Counterfactual Risk Minimization	GNN	Graph Neural Network
CRS	Conversational Recommender Systems	HNB	Hybrid Neuro Bandit
CSI	Critical Success Index	IPS	Inverse Propensity Scoring
DNN	Deep Neural Network	KG	Knowledge Graph
DRO	Distributionally Robust Optimization	LBSNs	Location-Based Social Networks
EXP-3	Exponential-weight algorithm for Exploration and Exploitation	LinUCB	Linear Upper Confidence Bound Disjoint
		LMM	Large Language Models
		MCC	Matthews Correlation Coefficient
		MLP	Multi-Layer Perceptron
		MO-MAB	Multi-objective Multi-armed Bandit
		MRR	Mean Reciprocal Rank

NOMENCLATURE

NDCG	Normalized Discounted Cumulative Gain	SDM	Sequential Decision-Making
POIs	Points-of-Interest	ST-LSTM	Spatial-Temporal Long-Short Term Memory
RL	Reinforcement Learning	T	Temporal
RLSE	Recursive Least Square Estimator	TS	Thompson Sampling
RNN	Recurrent Neural Networks	TSV	Tab-Separated Values
RS	Recommendation Systems	UTC	Coordinated Universal Time
S	Social	XAI	Explainable AI

Chapter 1

Introduction

1.1 General

In today's digital era, personalization is vital to enhancing the user experience, with vast amounts of data being generated and disseminated every second. Recommendation Systems (RS) address the challenge of personalization by leveraging advanced algorithms to analyze extensive data, such as user preferences, past interactions, search history, ratings, reviews, and demographic information. These systems provide personalized content, products, and services tailored to individual needs, effectively cutting through the noise of information overload. By delivering targeted recommendations, these systems enhance user engagement and experiences on various online platforms, including e-commerce websites, streaming services, and social media. At the same time, RS help businesses increase conversions and build stronger customer relationships.

Thus, a sound RS provides recommendations that are personalized, diverse, non-repetitive (not recommending the same item repeatedly), and relevant (only available items). RS can be broadly categorized into three types: content-based filtering, collaborative filtering, and hybrid approaches.

A content-based filtering RS predicts users' preferences based on their previous interactions with items, such as their viewing and purchasing history. For example, if a user has already read a book from a particular author or purchased a product from a specific brand, the content-based filtering system will recommend a book from that author or a product from that brand next. However, one downside of this approach is that the recommended items are often too similar to what the user has already seen or interacted with, potentially missing opportunities for diversification.

In contrast, collaborative filtering RS make recommendations based on the behavior and preferences of similar users or similar items. Lastly, hybrid recommendation systems combine multiple recommendation techniques to provide diverse and personalized recommendations.

Despite their effectiveness in many scenarios, these systems often fall short in addressing the Sequential Decision-Making (SDM) problem under uncertainty. SDM under uncertainty involves making a series of interdependent decisions where each choice influences future options and outcomes, all within a context of incomplete or imperfect information. This process is complex because the decision-maker agent must consider not only the immediate consequences of each action but also how these actions will impact future decisions and the overall objective.

In practice, this problem occurs in scenarios where decisions need to be made by an agent in an environment with limited information available, and new information arrives after each decision. Uncertainty adds an additional layer of difficulty, as the agent often lacks full knowledge of the environment or the outcomes of actions.

For instance, consider an online music streaming service app, such as Wynk. The app consists of more than 15 different modules on the homepage, ranging from programmatically curated modules such as “new releases”, “top shows”, “editorial picks”, etc., to recommendation modules like “recommended artists”, “top picks for you”, and a wide range of topics like “Bollywood romantic”, “Party”, etc. The challenge is to rearrange the modules to reflect the streaming habits of a user. For example, if a user mostly listens to podcasts and new releases, these modules should appear in the first few positions in the app. Further, as the user’s interests change over time, the solution must be able to adapt to these changing interests.

This represents a classic tradeoff between exploitation and exploration. Exploitation involves leveraging known information to maximize immediate rewards by choosing the best-known option based on past experiences. In contrast, exploration seeks to gather more information about unknown options, potentially leading to even greater rewards in the future. The tradeoff arises because focusing too much on exploitation can lead to sub-optimal results in dynamic environments. If a system only exploits known information, it might miss out on potentially better options that have not yet been discovered. Over time, this can lead to stagnation and a lack of adaptability to changing user preferences or market conditions. Conversely, excessive exploration can also be detrimental. Constantly trying new and uncertain options can lead to inconsistent performance and missed

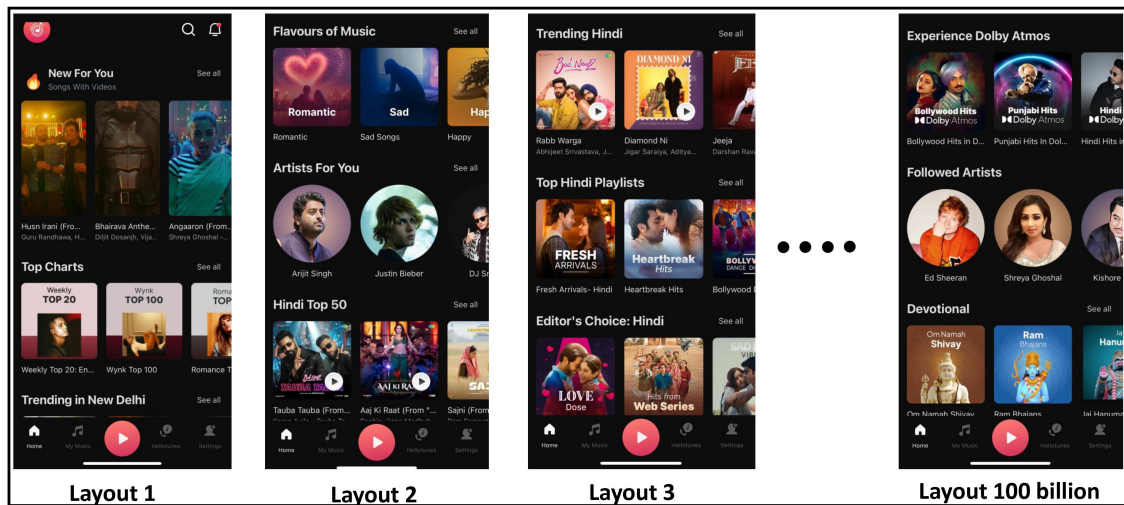


Figure 1.1: Exploring potential layout of music app Wynk

opportunities to capitalize on well-established, high-performing choices. It can also result in user frustration if recommendations are too varied and do not align with the user's interests. Thus, balancing exploration and exploitation is crucial to achieving optimal performance, which traditional recommender systems fail to achieve.

Another common approach to address this challenge, as described in Fig. 1.1, is A/B testing. A/B testing compares multiple versions/layouts of an app and determines which layout is best based on user interaction and predefined metrics. A/B testing consists of two phases. The first phase is the exploration phase, where an equal number of experimental units are generated for different possible layouts/variants of the app, and the response rate of each layout is measured statistically. The second phase is the exploitation phase, where only the successful layout is picked, and the rest of the layouts are discarded. However, A/B testing fails in this scenario for several reasons:

- There are an exponentially large number of combinations (nearly 130 billion!) to try out, making it complex to manage and analyze these many A/B tests, as each A/B test demands significant time and resources to implement and interpret. Thus, such a huge computational cost and time complexity make A/B testing impractical.
- A/B testing evaluates static changes between two or more layouts and doesn't account for dynamic user preferences and continuous real-time adjustments based on user interactions.

- In the exploration phase of A/B testing, there is a cost to try out each layout, and an experimenter, after gaining sufficient confidence, might not want to gather more information about certain layouts, which is not possible.
- A particular layout/variant might be the best one in the exploration phase; however, it may not remain the same over time. This is because A/B testing makes a discrete jump from pure exploration to pure exploitation, whereas the experimenter may wish to continuously transition between the two.

To overcome these shortcomings, Contextual Multi-Armed Bandits (CMAB) algorithms are proposed. CMAB algorithms are a type of reinforcement learning algorithm that uses a reward system to learn iteratively and adapt to the user's choices, providing an efficient way to balance exploitation and exploration to minimize cumulative regret. Specifically, CMAB algorithms are presented with K different options (also known as arms) to choose from. Each option is associated with a reward that is unknown to the algorithm and is revealed after choosing the arm. At the time of selecting an arm, additional information, also known as context, is available. Context includes historical data about each customer, such as clicks on the website, past purchases, and opened emails, as well as data from their ongoing session, such as recent searches, which helps in personalizing the customer's experience. The algorithm chooses an option and displays it to the customer/user. Then the action happens—either the customer converts or doesn't. This could be a click on a banner, a newsletter subscription, email engagement, or a purchase later on—essentially any other action the algorithm is optimizing for. The contextual “bandit” takes this as feedback. Over time, it learns to balance exploration and exploitation in such a way that it tries to learn the best choice to make while spending a minimum number of trials exploring the options. Due to the smooth transition between exploration and exploitation, contextual bandit algorithms are preferable in situations where expected rewards could change or new variants are added over time. Additionally, due to their adaptive nature, they can be fed with new hypotheses to test without getting stuck with suboptimal variants. Thus, contextual bandits:

1. Smoothly decrease the amount of exploration over time instead of requiring the experimenter to make an abrupt jump, as in the case of A/B testing.
2. Naturally determine the best action to take under new circumstances, as there are no absolute winners at all times.

3. Focus the experimenter's resources on the optimal variants instead of wasting time on suboptimal variants which otherwise would have been over-explored in an A/B experiment.

1.2 Background

The multi-armed bandit problem [1] was first introduced as a sequential decision problem with statistical assumptions over the distribution of rewards over each arm to establish worst-case lower bounds for bandit experiments. It is recognized as an exploration/exploitation trade-off problem, maximizing the user's satisfaction by selecting the best arm (i.e., exploitation), while exploring new choices/arms for uncertainties in the user's interests. Epsilon-greedy [2] and Epoch Greedy [3] are classic algorithms for random exploration. Thompson Sampling [4] [5] is a heuristic algorithm that handles the exploration-exploitation trade-off by maintaining probability distributions for each arm and then sampling from them on every trial to choose the one that predicts better rewards. Linear Upper Confidence Bound Disjoint (LinUCB) [6] [7] summed a linear relationship between the expected reward and the context. However, Chapelle and Li [4] showed that Thompson Sampling (TS) outperforms UCB. With time, variants of the initial problem with different practical scenarios and constraints, such as non-stationary data (where both data distributions and rewards may change over time), personalization on a per-user basis, and no assumptions on how the rewards are generated, were introduced. In adversarial bandits [8], an adversary controls the rewards, while in the stochastic bandit formulation [9], arms' reward distribution is given by a well-behaved stochastic process instead of the statistical assumption as in the originally defined multi-arm bandit problem. Various optimal solutions using stochastic formulations [10] [9], adversarial formulations [11] [12], and Bayesian formulations [13] [14] have been provided in the literature. To provide personalized services, CMAB [15] [3] uses a context feature vector (which consists of the user's profile and choices) on each iteration and predicts the best choice out of the possible options for user satisfaction and interest. Recently, many solutions have been developed around contextual bandits to represent real-world problems. In the policy elimination algorithm [16], only good policies are kept in the working set, and the epoch-greedy algorithm is used for exploration. However, it is difficult to keep track of good policies and difficult to implement. If an optimal policy is removed by mistake, the algorithm can never recover. The Exponential-weight algorithm for Exploration and Exploitation

(EXP-3) [11] used weights for each arm, and weights are incremented exponentially for choosing the best arm, whereas, the Exponential-weight algorithm for Exploration and Exploitation using Expert advice (EXP-4) [11], used advice from multiple experts to explore the connection between the context vector and the rewards of each arm, making it suitable for non-stationary data. Banditron [17] employed a perceptron [18] to model expected rewards. It maintained weight vectors for each arm and outputs a prediction to the arm with the highest score. When Banditron is coupled with upper confidence bound techniques, it is called Confidit [19], which provides better performance than its base algorithm, Banditron. Linear algorithms lack representational power, and to overcome this shortcoming, deep neural networks have become popular. Many Deep Neural Networks (DNN) use TS as an exploration technique where a context is drawn at each round, and the posterior distribution is updated with the result of the action (i.e., the feedback). In [20], an empirical comparative study on how different posterior approximations by various algorithmic approaches/models, such as the dropout model and neural linear model, affect decision-making performance via TS is presented. In DNN approaches, such as Neural Bandits [21], a neural model is maintained for each arm to facilitate adding and removing arms. At each step, the context vector is taken as input for each neural model, and a score is obtained. The model chooses the arm with the highest score. Epsilon-Greedy is used as an exploration technique in this case. However, there are various limitations. Firstly, it is a very daunting task to train different architectures and find optimal hyperparameters for each architecture, which requires significant computational power. Secondly, a neural network requires a large training dataset, which is infeasible in real-time applications where quick responses are essential. The dropout [22] model addresses these issues by dropping out the hidden layers in the network, i.e., randomly zeroing out the output of a neuron in the forward pass with a probability p . This prevents both overfitting and the computational power required to train the model. However, the number of hyperparameters to tune and the training time required will still be much higher compared to linear models.

Mathematical Modelling

The mathematical modeling of a CMAB algorithm is as follows:

Assume a class of contexts: $\mathcal{C} \subseteq \mathbb{R}^d$ (where \mathcal{C} can also be a finite set), and a set of arms $[K]$. For $t = 1, \dots, T$, each round t proceeds as follows:

- Observe context $x_t \in \mathcal{C}$ (this can be adversarially or randomly chosen).

- Choose arm $a_t \in [K]$ (where a_t is the action that the algorithm takes, e.g., an article recommended to the user with features x_t).
- Receive a reward r_t (a function of both the context x_t and the action a_t). Alternatively, receive loss $l_{i,t}$. Assume that $l_{i,t}$ and r_t are bounded within $[0, 1]$.
- Update the training data set with $\{(r_t, x_t, a_t)\}$ in order to improve future estimates of the value, Q , of an action. The value of an action given an arm is chosen and is denoted by $Q(a_t) = \mathbb{E}[r_t | a_t]$.
- Compute regret as the difference between the expected reward rate of the optimal arm, a_t^* , and the arm of choice, a_t , in a particular trial. The cumulative regret can then be defined as:

$$R_T = \sum_{t=1}^T [Q(a_t^*) - Q(a_t)] \quad (1.1)$$

The goal of the contextual bandit algorithms is to minimize the cumulative regret over the time horizon T , where,

- K : Number of arms
- T : Total number of time steps
- C : The set of contexts (this can be a finite set, or an infinite set such as a vector space \mathbb{R}^d). For example, this could be information about a particular user, such as their browsing history, location, age, etc.
- a_t : The arm chosen at time t . For example, this could be an article chosen for recommendation.
- r_t : The reward received at time t . In our example, this is the rating given by user x_t for article a_t .
- $l_{i,t}$: The loss incurred by pulling arm i at time t .

1.3 Research Gaps

With the advent of machine learning algorithms, numerous solutions have been proposed to recommend items or services, but there has been limited progress in the realm of contextual personalization. Some of the common challenges are:

1. **Overloading of options:** There is a vast range of choices available to be recommended to a user, but it is highly unlikely that a user wants to try out a large fraction of products. Instead, only a few items are demanded by many users, while many items are only requested by a few.
2. **Data sparsity:** For a RS to work effectively, a substantial dataset consisting of past interactions between items and users is required. This allows algorithms to gain deep insights and patterns about user choices and preferences. In online settings, such as live music streaming or food ordering apps like Zomato, where limited information is available, this becomes a serious limitation. This issue not only hampers the training of machine learning models but also impacts the accuracy of item recommendations according to user taste.
3. **Cold start problem:** Most RS struggle to recommend items about which they have little to no information. This usually happens when a new user signs up or a new item is added. New users have no browsing history or known preferences, and newly added items have no past interactions or ratings.
4. **Biased recommendations:** One major challenge faced by various real-life recommendation systems is their tendency to promote a particular set of items. For example, a tour and travel app like MakeMyTrip might recommend or list certain hotels despite the user's past disliking. Another possible issue is inhibiting or suppressing certain items from the user. For instance, a news app RS might hide certain relevant articles and highlight others to sway public opinion or promote propaganda, serving the vested interests of corporate giants.
5. **Dynamic user taste and likings:** Items previously admired by the user may no longer be liked for unknown reasons. RS struggle to adapt to this changing behavior over time. This primarily happens because new suggestions are made based on previously interacted data fed into machine learning models, without incorporating new, unbiased, and random datasets through random recommendations for re-training the recommender models.

6. **Accurate, but diverse predictions:** This is a major bottleneck for traditional recommender systems such as matrix factorization, which involves decomposing a user-item matrix to find latent factors that explain ratings or other hybrid approaches. Over time, the recommendations become stale, and no new items matching the user's preferences are suggested. There is a lack of experimentation with new choices added to the platform, and algorithms continue to exploit tried and tested recommendations. This approach restricts users to a narrow set of similar items consumed in the past. Simultaneously, suggesting diverse choices risks reduced efficiency. Thus, a balance is needed between exploiting existing choices and exploring new options.
7. **High computational complexity:** RS can be computationally intensive and require significant processing power. This poses a problem for large-scale RS or for systems needing real-time recommendations.
8. **Limited personalization:** Many content-based RS rely solely on the characteristics of the items being recommended. As a result, they do not account for user preferences, interests, and behavior, which can lead to a lack of personalization and a poor user experience.
9. **Data biasness:** Many RS are vulnerable to biases in the dataset used for making recommendations. For example, if the data is skewed towards certain items or consists mainly of interactions with a specific set of items, the recommendations generated will also be biased towards these items.
10. **Lack of transparency:** RS often act as black boxes, providing little to no insight into how or on what criteria an item is suggested. Additionally, there is no accountability for the suggestions made by these algorithms, leaving users with no choice but to accept recommendations with caution.

1.4 Research Objectives

Keeping in view the above considerations, we defined the following objectives of the thesis:

- To perform a comprehensive literature review and analysis for web personalization.

- To design and develop a holistic framework for web personalization.
- To design and develop web personalization solutions based on user preferences and their behavior using Soft Computing.

1.5 Contributions

The major contributions of this thesis are:

Development of FuzzyBandit: An Autonomous Personalized Model based on Contextual Multi-Arm Bandits using Explainable Artificial Intelligence (XAI)

In today's digital era, many practical applications require providing relevant content and personalized services tailored to users based on knowledge about their preferences and behavior. While many solutions have been proposed to recommend items or services, progress in personalization remains limited. Contextual bandit algorithms address this issue by solving the exploration versus exploitation dilemma to provide customized solutions according to user preferences. However, many machine decisions remain poorly understood, and a high level of accountability is required. There is a need to understand the underlying mechanisms of the black-box nature of contextual bandit algorithms.

To address this, an XAI-based FuzzyBandit model was designed and developed. Each arm in FuzzyBandit mimics an Adaptive Neuro-Fuzzy Inference System (ANFIS) to address the ambiguous nature of arm selection in contextual bandits. The model uses a feedback mechanism to adjust its parameters based on the relevance and diversity of the features to maximize reward generation. The FuzzyBandit model optimizes decisions at each trial based on previous observations and generates explanations for its decisions. This makes the model interpretable, allowing users to understand the rationale behind the decisions made.

The FuzzyBandit model has been empirically compared with seven popular models from the literature on four benchmark datasets over nine criteria: recall, specificity, precision, prevalence, F1 score, Matthews correlation coefficient (MCC), Fowlkes–Mallows index (FM), Critical Success Index (CSI), and accuracy. Additionally, a coherent mathematical framework is provided to calculate a trust score α for the FuzzyBandit model, which helps induce trust in the decisions taken and detect any erroneous reasoning.

Thus, the developed model not only provides personalized choices for real-world applications but also offers reasoning for the decisions made.

Development of Hybrid Neuro Bandit: A Contextual Multi-Arm Bandit Model for Online Recommendation and Personalization

Consider an online advertisement model, like YouTube, where a user visits the website, queries a request, and is rendered an advertisement along with the requisite content on the web page. The system aims to show the most relevant ad to maximize the likelihood of the user clicking on it. If the user clicks on the ad, the system receives a reward. Feedback for the system is the action taken by the user. The system cannot observe user actions if other advertisements are shown. Therefore, the system must balance the exploration versus exploitation dilemma: either continue showing the ad previously clicked by the user (exploitation) or explore new ads from the available set (exploration).

This scenario can be modeled as a Contextual Multi-Arm Bandit Model (CMAB) problem, where each trial represents a user visit to the website and each arm is the ad displayed. For each trial, the context is announced, and the agent selects an arm from the set of predefined arms $[k] = \{1, 2, \dots, k\}$ and observes the reward associated with the selected arm. Over several iterations, the agent finds a relation between the context and rewards, i.e., arm reward distribution. The goal is to maximize cumulative rewards over n trials or improve the accuracy of predicting the correct arm in each trial by updating the model with user feedback. This involves exploring new ads to cater to stochastic user behavior over time. For instance, if the system shows a relevant ad and the user clicks on it but no longer wants to see the same ad, the system needs to adapt to this change in user behavior.

Contextual bandits can provide personalized recommendations for online ad streaming platforms. Similar applications include clinical trials to reduce patient losses by finding the best medicine for a given set of symptoms, marketing optimization to improve click-through rates, website layout optimization, adaptive routing to minimize network delays, etc. However, challenges such as cold start, scalability, contextual inference, and non-stationarity must be addressed.

To meet these challenges, a novel Hybrid Neural Bandit (HNB) model was developed. The HNB model combines expert advice from existing CMAB models into one unit to exploit their respective merits and provide personalized recommendations. The HNB model integrates decisions from bandit experts by assigning weights to each expert. These

weights are fine-tuned through a neural network with user feedback used for training. This approach enhances the overall user experience and provides relevant recommendations. The HNB model has been empirically compared with existing state-of-the-art contextual bandit models over nine performance metrics: recall, specificity, precision, prevalence, F1 score, MCC, FM, CSI, and accuracy.

Development of Contextual-Bandit-POI: Personalized Next Point-Of-Interest Recommendation using Contextual Bandits

In today's era, there is a growing trend of check-in behavior among Generation Z users, who frequently share their experiences at various Points-of-Interest (POIs) like cafés, restaurants, and hotels on location-based social networks (LBSNs). These check-ins include valuable contextual data such as timestamps, comments, and GPS coordinates. The massive amount of check-in data collected on platforms like Foursquare provides an opportunity to analyze user behavior over time and predict their future visits to POIs. However, the effectiveness of these predictions is challenged by issues like data sparsity, where user interactions with POIs are limited, and cold start problems, which affect new users and POIs lacking historical data.

To improve the next POI recommendations, it's crucial to consider both spatial-temporal intervals (the time and distance between check-ins) and the user's long-term and short-term preferences. Traditional models, including those based on Recurrent Neural Networks (RNNs), often fail to capture the full range of user behaviors, particularly over long periods. Advanced models, like those incorporating LSTM or GRU, attempt to address these shortcomings but still face limitations, particularly in personalizing recommendations to individual users' preferences and adapting to their dynamic behaviors.

Thus, there is a need for innovative algorithms that consider various contextual factors—social, geographical, temporal, and categorical—in making more accurate and personalized POI recommendations. Additionally, it is important to use online learning settings so that models can adapt in real-time to changes in user preferences, as opposed to the traditional offline settings that rely on static data.

In this chapter, Contextual-POI-Bandit framework is designed and developed, which integrates CMAB to provide personalized recommendations by balancing exploration (suggesting new POIs) and exploitation (recommending familiar POIs). This framework is flexible, allowing the integration of current and future CMAB models, and introduces a novel Fuzzy Neural Network (FNN) algorithm that combines neural networks and fuzzy

logic to handle uncertainty and improve decision-making in next POI recommendations. The framework is empirically evaluated using benchmark datasets to optimize and validate its performance across various metrics.

1.6 Thesis Organization

Finally, the thesis work is organized in the following six chapters. The brief summary of each chapter is given as follows,

Chapter 1: Introduction

This chapter briefly introduces personalized recommendation and the use of CMAB algorithms leveraging reinforcement learning to balance exploration and exploitation by using user-specific contextual data. These algorithms personalize user experiences, particularly in dynamic environments by optimizing decision-making and minimizing cumulative regret over time. The chapter then discusses the research gaps and outlines the motivation behind the research and work done to achieve them.

Chapter 2: Literature Review: Personalization in Web Applications using Advanced Machine Learning Models

This chapter describes the personalized recommender systems that effectively match users' preferences by leveraging various types of information to suggest relevant items, thereby mitigating the problem of information overload. A detailed study on achieving personalization in various domains using advanced machine learning models ranging from reinforcement learning to Large Language models (LLMs) along with challenges has been done in this chapter.

Chapter 3: Explainable AI Based FuzzyBandit Model using Contextual Bandits

This chapter focuses on the design and development of an autonomous personalized model, FuzzyBandit, based on contextual multi-arm bandits using XAI. The FuzzyBandit model is empirically compared with seven existing contextual bandit models on four benchmark datasets across nine criteria: recall, specificity, precision, prevalence, F1 score, Matthews correlation coefficient (MCC), Fowlkes–Mallows index (FM), critical success index (CSI), and accuracy. Additionally, a coherent mathematical framework is

provided to calculate a trust score α for the proposed FuzzyBandit model, which helps induce trust in the user regarding the decisions made and detects any erroneous reasoning in the model.

Chapter 4: Hybrid Neuro Bandit

This chapter is dedicated to the development of a new contextual multi-arm bandit model, Hybrid Neuro Bandit (HNB), for web personalization. The HNB model integrates expert advice from existing CMAB models into a unified unit. These bandit experts are assigned weights, which are fine-tuned through a neural network, with user feedback used to train the overall system.

Chapter 5: Contextual-POI-Bandit: A Contextual Multi-Arm based Framework for Next Point-Of-Interest (POI) Recommendation

This chapter focuses on the application of CMAB algorithms developed in previous chapters to study user behaviour closely and predict future personalized POI visits, which holds practical implications for applications like targeted advertising and traffic management. Next, POI recommendation is bewitched with challenges such as data sparsity - wherein users interact with only a small subset of available POIs and the cold start problem for new users and POIs, complicates accurate recommendation efforts. Traditional models often struggle with these issues, underscoring the need for innovative approaches that can better account for temporal and geographical dynamics in user behavior. Advanced models, such as those incorporating Recurrent Neural Networks (RNNs) and their extensions like LSTM and GRU, have been developed to address these challenges, but they often fall short in capturing long-term user preferences and non-linear relationships between POIs. Furthermore, most models fail to offer personalized recommendations that align with individual user contexts, which can vary based on factors like current activity, emotional state, or social surroundings. To overcome these limitations, a Contextual-POI-Bandit framework has been proposed that user contextual information to render the next PoI as per user liking.

Chapter 6: Conclusion and Future Scope

This chapter summarizes the conclusions drawn from this research work and highlights potential future work in this area.

Chapter 2

Literature Review: Personalization in Web Applications using Advanced Machine Learning Models

2.1 Introduction

Personalization is the skill of customizing experiences based on individual preferences and serves as a connection between humans and robots in today's technology-driven society. Web personalization recommender systems adapt to individual preferences, enhancing user interactions across digital platforms like entertainment [23], e-commerce [24], and job matching [25]. For instance, in movie recommendation platforms like Internet Movie Database (IMDB) and Netflix, users receive suggestions based on their past interactions and the content of movies, facilitating the discovery of new films matching their interests. These systems rely on user-item interactions and associated textual information (such as item descriptions, user profiles, and reviews) to predict the likelihood of a user liking a particular item [26]. They offer content recommendations and customize user interfaces and communication styles. As artificial intelligence progresses, personalization becomes more sophisticated, necessitating advanced techniques to handle diverse user intents. The pursuit of improved personalization stems from understanding users' evolving needs. Hence, web personalization Recommendation Systems (RS) seamlessly integrates human-machine interactions into daily life for personalized experiences and grows exponentially as the technology evolves. Some of the real-time web personalization applications are described as follows,

- **Targeted advertising:** Online platforms frequently display targeted ads that utilize real-time data to ensure that the right kind of content is shown to users based on their past or current behavior. By analyzing user profiles and interactions in real-time, advertisers can make ad placements more effective for greater user engagement and conversion.
- **Personalized search results:** Search engines and websites benefit from real-time personalization which adjusts search results using information about what users have just searched for, clicked on, or interacted with. This ensures that users receive the most current results pertinent to their immediate interests and needs, thereby improving the overall search experience.
- **Dynamic content recommendations:** To provide users with content recommendations tailored specifically to them, such as articles, products, or videos based on their browsing history and ongoing interactions, dynamic personalization needs to be incorporated into content recommendation systems. This approach increases user involvement by presenting materials that align with individual preferences and tastes.
- **Dynamic pricing strategies:** Pricing can be adjusted based on the behavior, location, or engagement level of a user through real-time personalization. For instance, prices may be altered or discounts offered dynamically by e-commerce platforms depending on the user's browsing history or purchasing behaviors, thus optimizing revenue and user satisfaction.
- **Contextual health recommendations:** Real-time customization in health and wellness platforms allows for the adaptation of content and recommendations to fit current health metrics or activity levels. For example, workout suggestions or nutritional advice provided by a fitness app may change according to data from wearable devices at that time.
- **Interactive virtual try-ons:** Augmented reality (AR) technology is used in real-time virtual fitting rooms where users can try on clothing virtually. These applications personalize the shopping experience by adjusting the virtual fit and appearance of garments based on users' body dimensions and preferences in real-time, making fashion recommendations more accurate and appealing.

2.2 Web Personalization using Advanced Machine Learning (ML) Models

Extensive research and numerous studies have been conducted on personalized recommendations across various domains, reflecting a significant advancement in this field. To simplify the current landscape, recent studies can be categorized into three primary areas: advanced machine learning techniques utilizing Deep Neural Networks (DNNs), Reinforcement Learning (RL) methodologies, and Large Language Models (LLMs). DNNs work structures to figure out intricate patterns about user preferences and behaviors hence producing unique accurate results. RL is another notable progress where algorithms optimize decision-making through trial and error. For instance, this approach works well on dynamic environments whereby it learns continuously while adapting based on real-time feedback to enhance relevance as well as effectiveness of recommendations. LLMs can generate fine-grained and contextually sensitive personalization by analyzing huge volumes of textual data. They can process vast amounts of text because they understand complex written information.

2.2.1 Neural Network Models

DNNs [27], [28] have gained widespread adoption in enhancing recommender systems thanks to their exceptional representation learning capabilities. They excel in modelling user-item interactions through various architectures. Recurrent Neural Networks (RNNs) are highly efficient in handling sequential data, allowing for the identification of complex relationships in user interaction sequences [29], [30]. Graph Neural Networks (GNNs) are sophisticated techniques for acquiring representations in graph-structured data, such as users' online behaviors [31]. They acquire efficient models of users and items. Additionally, DNNs are adept at encoding side information. For example, methods based on Bidirectional Encoder Representations from Transformers (BERT) extract and utilize textual reviews from users, further enhancing the capabilities of recommender systems [32]. However, there exist several limitations. Firstly, traditional DNN-based models as well as pre-trained language models such as BERT, have constraints in capturing comprehensive textual information about users and items. Consequently, they exhibit inferior natural language understanding capabilities, leading to less-than-optimal performance in various recommendation scenarios. Secondly, many existing RS are tailored to specific tasks and lack robust generalization abilities to handle unseen recommendation tasks effectively.

For instance, an algorithm trained on a user-item rating matrix to predict movie ratings may struggle to provide top-K movie recommendations with explanations. This limitation arises because these recommendation architectures are heavily reliant on task-specific data and domain knowledge, making them less adaptable to different recommendation scenarios. Thirdly, while DNN-based recommendation methods excel in tasks like rating prediction or top-K recommendations, which require straightforward decisions, they encounter challenges in complex decision-making processes, such as trip planning recommendations, where the system needs to consider popular tourist attractions based on the destination, arrange a suitable itinerary, and recommend a travel plan tailored to specific user preferences like cost and time constraints.

2.2.2 Web Personalization through Large Language Models (LLMs) and its Challenges

LLMs have been recognized as significant developments in the field of AI today. Their proficiency in comprehending and generating human language, along with their strong capabilities in generalization and reasoning, has greatly enhanced their performance. Additionally, their ability to adapt to new tasks and domains highlights their versatility. Consequently, there is a growing interest in utilizing LLMs to transform recommender systems, with the objective of offering customized and top-notch recommendations to the user. LLMs have shown impressive generalization and reasoning abilities [33] [34] as they are trained on vast data from diverse sources. This allows transformer-based architectures [35], categorized as architectures comprising only encoders, for example, BERT [36]; architectures such as GPT, which consist of both encoders and decoders; and lastly architectures like T5 [37], consisting only of decoders, to perform remarkably well in various unseen problems and domains without extensive fine-tuning. Advanced techniques like as prompting and in-context learning significantly improve their performance, making LLMs highly promising for revolutionizing recommender systems. LLMs have a crucial function in recommender systems, specifically in forecasting user ratings for items through the analysis of previous interactions and preferences. This enhances recommendation accuracy [38] [39]. LLMs are also utilized in sequential recommendations, such as TALLRec [40], M6-Rec [41], PALR [42], and P5 [43], which predict users' next preferences based on interaction sequences. Next, the obstacles and issues associated with leveraging LLM for personalization in several web applications are examined as follows,

Knowledge Graphs

Knowledge graphs, with nodes as entities and edges as relations, enhance recommender performance as side information and act as the common format of knowledge bases. LLMs have demonstrated remarkable proficiency in retrieving factual knowledge, akin to explicit knowledge bases [44], [45], [46], [47], [48], [49], [50], [51], [52]. This ability enables the construction of more extensive knowledge graphs for the recommender systems. However, despite the promise offered by LLMs, existing methods [53] [54] in knowledge graph construction face challenges in handling incomplete knowledge graphs and integrating textual corpus data. Researchers [55] [56] have commenced exploring how LLMs can address these challenges, particularly through knowledge completion and construction tasks. In the realm of knowledge graph completion, efforts are directed towards leveraging LLM models such as MTL-KGC [57], MEMKGC [58], StAR [59], GenKGC [60], TagReal [61], and AutoKG [62] to encode text or generate missing facts within knowledge graphs. This involves enhancing the completeness of knowledge graphs by inferring and adding missing information. On the other hand, knowledge graph construction involves the structured representation of knowledge, including entity discovery [63], [64], coreference resolution [65], [66], and relation extraction [67], [62]. LLMs offer potential solutions for each of these subtasks, allowing for more accurate and comprehensive knowledge graph construction. Moreover, LLMs show promise in enabling end-to-end construction [68],[69] wherein they directly build knowledge graphs from raw text data. This holistic approach streamlines the process of constructing a knowledge graph, eliminating the need for intermediate steps and enhancing efficiency. However, LLMs due to their inherent nature may introduce ambiguity or inaccurate information can manifest as extraneous information or noise in the recommendation process [57], leading to responses that lack informative context or relevance, despite being syntactically correct.

Direct Recommendations and Automated Reasoning

The advancement of LLMs has revealed their remarkable reasoning abilities when sufficiently scaled, akin to human intelligence in decision-making and problem-solving [70]. Through techniques like "chain of thoughts" prompting, LLMs can exhibit emergent reasoning skills, enabling them to draw conclusions based on evidence or logic [71]. In the realm of recommender systems, these reasoning capabilities empower LLMs to enhance user interest mining, thereby improving overall performance. Additionally, LLMs demonstrate "step by step" reasoning, leveraging prompts that include intermediate steps

to tackle complex tasks effectively [71]. For instance, Wang and Lim [72] introduce NIR, a three-step prompt designed to capture user preferences, filter items, and re-rank recommendations. Moreover, Automated Machine Learning (AutoML) is increasingly utilized in recommender systems to streamline manual setup processes, particularly in optimizing embedding sizes [58] [59] [60] [61] and other facets like feature selection and model architecture. However following challenges needs to addressed effectively by LLMs for significant improvements in automated learning approaches effective recommendation algorithms and systems:

- **Complex search space:** The search space in recommender systems is highly intricate, involving a wide range of kinds and encountering problems related to volume. This complexity poses challenges for successful exploration and optimization.
- **Lack of foundation:** Recommender systems do not have a solid understanding of the important elements in the search space, especially when it comes to successful interactions between advanced features. This sets them apart from other sectors that have well-established network structures. Further, This knowledge gap is compounded by the diverse and domain-specific nature of recommender systems, operating across various scenarios.

Conversational Recommender Systems (CRS)

CRS personalize recommendations through dialogue, allowing real-time adjustments based on user feedback. CRS mainly comprises of dialogue and recommendation modules; where the dialogue module is crucial for effective user-system interactions and preference understanding. The dialogue system is further classified into chit-chat and task-oriented categories. Task-oriented dialogue systems are preferred due to their ability to assist users in accomplishing certain tasks. The two most common approaches used are response generation pipelines and end-to-end methods [73] [74]. The former one generates responses in four components namely, dialogue understanding [75] [76], dialogue state tracking [77], dialogue policy learning [78] and natural language generation [79] [80] but however suffers from scalability issues and lacks sync between the different components. The latter one handles all the processing steps collectively using an encoder-decoder model but requires substantial supervised data for training. LLMs such as Bard and Large Language Model Meta AI (LLaMA) have demonstrated remarkable performance in conversational abilities. However, the first and most prominent challenge faced is regarding private domain

data as these LLM models predominantly rely on publicly available internet sources for training data, resulting in a potential blind spot concerning data within closed information platforms. Further, massive high-quality dialogue data is required to train such complex models. Also, it is difficult to generate such high-quality dialogue data as the available dataset mainly comprises explicit or implicit interactions between users and items and lacks conversational context.

Web Personalized Content Creator

Traditionally, recommender systems have functioned by offering existing products to users, taking advantage of their preferences and past behavior. However, as technology evolves and content creation platforms become more sophisticated, there has been a notable shift towards personalized content creation. This personalized approach entails generating content tailored to individual user interests and preferences, particularly prevalent in online advertising contexts. Such content [81], [82] encompasses a variety of visual and semantic elements, including titles, abstracts, descriptions, copywriting, ad banners, thumbnails, and videos. Text ad generation, in particular, has garnered significant attention, focusing on crafting personalized ad titles and descriptions. While early approaches [83, 84, 85] relied on predefined templates to streamline the process, they often fell short of fully satisfying user preferences. More recent advancements [86] [87] [88] [89] have seen the emergence of data-driven methods, integrating user feedback within RL frameworks to guide the content generation process. In addition, the integration of pre-trained language models has dramatically improved the ability to generate content across different forms of content [90] [91] [92]. By leveraging these language models, content generation models can be refined to better align with user preferences, thereby improving overall effectiveness in delivering personalized content experiences.

Challenges in LLMs for Personalization

Personalizing services for users with LLMs poses various crucial issues that require attention. Firstly, effective personalization requires a deep understanding of user preferences, often extending beyond the general knowledge LLMs acquire through training. This indicates that adapting LLMs to cater effectively to personalized services is a significant unresolved issue. Additionally, there are privacy concerns associated with using LLMs for personalization. Since LLMs can remember users' confidential information to offer

personalized services, there is a valid concern about protecting user privacy. It's crucial to ensure that LLMs maintain privacy while providing personalized experiences to build trust with users. Moreover, LLMs trained on internet data may exhibit exposure bias, potentially resulting in unfair predictions for minority groups. This underscores the importance of mitigating biases in LLMs to ensure fair outcomes for all users. To address these challenges, the research community requires comprehensive benchmarks and evaluation datasets. However, the current availability of such resources is limited, indicating a need for collaborative efforts within the research community to bridge this gap. Furthermore, to fully utilize the potential of LLMs for personalization, it's vital to establish systematic methodological and experimental frameworks. These frameworks should encompass various aspects, including understanding user preferences, addressing privacy concerns, mitigating biases, and accurately evaluating model performance. In essence, addressing the challenges of personalization with LLMs necessitates a multidimensional approach involving domain-specific knowledge, privacy protection measures, bias mitigation strategies, and the development of robust evaluation frameworks. Collaboration within the research community is essential for advancing research in this area and realizing the full potential of personalized services powered by LLMs.

2.2.3 Web Personalization using Contextual Multi-Arm Bandits (CMAB) Models

The Contextual Bandit (CB) framework, also known by terms such as the partial label problem, associative bandit problem, and bandits with side information, enhances the traditional multi-armed bandit model by integrating contextual information into the decision-making process. In this framework, before an action is chosen, the bandit observes contextual data, which influences the potential reward associated with each action. This context-aware approach necessitates assumptions about the context and reward structures to manage the added complexity. For instance, when the context is a finite set, the model can be viewed as a collection of independent bandits indexed by context. In cases where the context is a vector space, common assumptions include linear [7] reward functions. It is understood that the multi-armed bandit problem [93] is an exploration/exploitation trade-off problem where the user's happiness is maximized by choosing the best arm or exploitation while investigating the new options/arms for uncertainty in the user's interests. While TS [94] [95] is the heuristic algorithm that has handled the exploration-exploitation trade-off by keeping probability distributions for each arm and then sampling

from them on each trial to choose the one that predicts greater rewards, Epsilon-greedy [96] and Epoch Greedy [97] are famous algorithms for random exploration. A linear link between the predicted reward and the situation was presupposed by LinUCB [98] [99]. Nevertheless, Chapelle and Li [94] demonstrated that TS defeats UCB.

The value of innovation and diversity is increasingly being incorporated into evaluation practices [100] [101]. Many contexts, including movies [102], tags [103], and adverts [104], have seen the successful application of multi-objective recommender systems. Multi-objective Multi-armed Bandit (MO-MAB) algorithms-based techniques merit our attention when handling online settings. In MO-MAB, depending on the set of objectives, several arms (items) are candidates for the best solution [105, 106, 107]. Ranked Bandits [108, 109] represent an extension of the traditional multi-armed bandit problem, focusing on generating and selecting a ranked list of actions rather than choosing a single optimal action. Unlike classic bandit problems where the objective is to maximize rewards from individual actions, ranked bandits aim to produce a ranking of available actions based on their expected rewards. The effectiveness of ranked bandit strategies is often assessed using metrics like Mean Reciprocal Rank (MRR) and Normalized Discounted Cumulative Gain (NDCG), which evaluate the quality of the ranked results. This framework enhances decision-making processes by providing a structured approach to ranking actions in scenarios where presenting a ranked list is crucial. The CB framework has significant applications in areas such as recommender systems, mobile health, and clinical trials. It involves a repeated interaction between a decision-maker and an environment that provides actions and contextual information, with the goal of discovering an effective strategy for selecting the best action given the context. The literature on CB optimization is divided into online and offline methods. Online methods focus on balancing exploration and exploitation to minimize regret, while offline methods, such as Batch Learning from Bandit Feedback (BLBF), utilize historical data to develop better policies. The challenge in offline policy learning includes addressing biases introduced by previous policies and managing high-variance estimates.

Approaches like Counterfactual Risk Minimization (CRM) and Distributionally Robust Optimization (DRO) have been developed to address these challenges, though they face difficulties related to optimizing non-convex objectives and handling large-scale data efficiently. Off-policy methods leverage logged data from previous actions to inform new decision-making strategies. This approach allows for evaluation of various strategies without the need for real-time experimentation, which can be costly and impractical. Techniques like estimating the reward function from logged data or employing Inverse

Propensity Scoring (IPS) are used to address limitations of bandit feedback. Combining these methods can mitigate their individual shortcomings. Alternatively, the offset-tree approach transforms logged data into a weighted classification problem, enabling the application of diverse machine learning techniques for improved feature selection and classification.

2.3 Why CMAB for Personalization?

Traditional machine learning algorithms often depend on historical data to make predictions, leveraging established patterns to optimize performance. However, these models primarily focus on exploiting known information and may neglect the dynamic nature of user preferences and interactions. Recent advancements in deep learning such as RNN based DNN models emphasize modeling historical user-item interactions to predict preferences, but they often fail to incorporate real-time feedback effectively. Consequently, these methods may not adapt swiftly to evolving user preferences, leading to less accurate recommendations. In contrast, the interactive nature of recommendation tasks—where users provide feedback on recommended items—highlights the necessity for RL based models that can adapt dynamically. CMAB demonstrate several advantages over RL in personalized recommendation tasks. The primary difference lies in how each method handles decision-making and adaptation. RL involves a sequential decision-making process where each action influences future states and rewards, creating a complex dependency between actions and outcomes. This sequential nature requires sophisticated models to manage these dependencies, which can be computationally intensive and may slow down the adaptation to new user behaviors. On the other hand, CMAB treats each interaction independently, even when multiple states are involved. This independence allows CMAB to adapt in real-time to user feedback without the need for managing the intricate dependencies seen in RL. As a result, CMAB is particularly well-suited for scenarios where rapid adaptation to user preferences is crucial, such as in personalized recommendations. It can quickly incorporate feedback and adjust recommendations accordingly, making it more responsive and efficient in dynamic environments compared to RL, which might require more complex and time-consuming processes to achieve similar levels of personalization. Table [2.1](#) shows how CMABs are better suited for personalization over LLMs.

Table 2.1: Comparison of LLMs and CMABs algorithms for personalization

Aspect	Large Language Models (LLMs)	Contextual Multi-Armed Bandits (CMABs)
Personalization Approach	LLMs achieve personalization by fine-tuning on user-specific data or through prompt engineering, where the model adapts its responses based on the context provided in the input. LLMs can generate highly personalized text by understanding and mimicking the user's language style, preferences, and context.	CMABs achieve personalization by employing a decision-making strategy that dynamically balances the exploration of new options with the exploitation of known user preferences in real-time, considering contextual factors such as time, location, and user activity to provide personalized outputs.
Data Utilization	LLMs leverage vast amounts of text data to understand and generate human-like language. However, their personalization is typically based on static data and may not adapt to changes in user behavior in real time.	CMABs focus on real-time learning from interactions. They utilize user-specific contextual data (such as time, location, and past preferences) to make personalized decisions dynamically, continually updating recommendations based on new user feedback.
Adaptability	LLMs personalize based on patterns learned from large-scale datasets during pre-training and fine-tuning phases. While they can generate contextually relevant outputs, their ability to adapt to new, unseen contexts without additional training is limited.	CMABs are inherently adaptable, as they balance exploration (trying new recommendations) and exploitation (using known preferences) in real-time. This adaptability allows CMABs to continuously refine personalization strategies based on the most recent user interactions.

Aspect	Large Language Models (LLMs)	Contextual Multi-Armed Bandits (CMABs)
Decision-Making	LLMs excel in understanding and generating text, making them suitable for tasks like content generation and conversational AI. They can provide personalized responses or recommendations, but they do not inherently focus on decision-making processes.	CMABs are designed specifically for decision-making under uncertainty. They aim to optimize the selection of personalized actions (such as recommending the next POI) by weighing multiple contextual factors, making them more effective in environments where user preferences evolve over time.
User Context Handling	LLMs can handle complex contexts and generate nuanced responses, but they often require substantial computational resources to process and integrate multiple contextual layers simultaneously. Their handling of context is more generalized and less targeted toward specific decision-making scenarios.	CMABs are designed to efficiently handle and integrate multiple contextual factors (e.g., social, temporal, geographical data) into their decision-making process. This focused approach allows personalized recommendation in applications, such as next POI recommendation.

Chapter 3

Explainable AI based FuzzyBandit Model using Contextual Bandits

3.1 Introduction

As discussed in Chapter 1 and 2, context-aware decision-making problems have attracted significant attention in today's era of artificial cognizance. The contextual bandit addresses these problems by solving the exploration versus exploitation dilemma to provide customized solutions as per the user's liking. However, a high level of accountability is required, and there is a need to understand the underlying mechanism of the black box nature of the contextual bandit algorithms. In this regard, this chapter discusses an explainable AI (XAI) based FuzzyBandit model that maximizes the cumulative reward by optimizing the decision at each trial based on the rewards received in previous observations and, at the same time, generates explanations for the decision made. The developed model uses an Adaptive Neuro-Fuzzy Inference System (ANFIS) to address the vague nature of arm selection in contextual bandits. It uses a feedback mechanism to adjust its parameters based on the relevance and diversity of the features to maximize reward generation. The efficacy of the developed FuzzyBandit model is established by empirically comparing it with the existing seven most popular art of literature models on four benchmark datasets over nine criteria, namely recall, specificity, precision, prevalence, F1 score, Matthews correlation coefficient (MCC), Fowlkes–Mallows index (FM), critical success index (CSI) and accuracy.

The rest of the chapter is organized as follows. Section 3.2 elaborates the developed XAI-based FuzzyBandit model. Section 3.3 provides a detailed analysis of the bench-

mark datasets and the simulation results. At last, Section 3.4 draws conclusions from the presented approach.

3.2 Explainable AI (XAI) based FuzzyBandit Model

A novel XAI based FuzzyBandit model, an autonomous decision system, is developed that not only optimizes and personalizes decisions for every situation based on the previous observations but at the same time generates explanations for the decisions made. For every decision, the model generates a confidence score α which enables the end user to easily understand and trust the decisions taken by the model. The developed scheme is comprised of two main steps which are elaborated as follows.

1. FuzzyBandit Ranking
2. XAI

3.2.1 FuzzyBandit Ranking

3.2.1.1 Ranking Model

At first, a contextual multi-arm bandit (CMAB) model settings is simulated where it inputs a finite n -dimension user context feature vector, $x(t) = \{x_1(t), x_2(t), \dots, x_n(t)\}$ and chooses an action, $a(t)$ from an alternate number of choices/actions for each trial t . Each action is associated with a reward that is unknown to the model and is revealed after the action is chosen.

The developed model observes a binary reward, $r_{a(t)}(t)$ i.e., +1 if the rendered action is accepted by the user; otherwise 0. For each trial, t , let $a^*(t)$ be the optimal arm/choice which would yield the highest reward, $r_{a^*(t)}(t)$. Then, the regret(t) is defined as the difference between the maximum reward for trial t and the observed reward by the model, i.e.,

$$\text{regret}(t) = r_{a^*(t)}(t) - r_{a(t)}(t) | x(t) \quad (3.1)$$

At the end of the T trials, the model aims to minimize the regret or maximize the total reward observed. This CMAB property allows the developed FuzzyBandit model to simulate real-world problems, for e.g., in an online food ordering app, each time the user logs into the app, the model uses the information about the user and renders a serviceable restaurant (action) to the user from a set of possible restaurants partner advertisement

(modelled as k arms). The context feature is the information about the user: user’s past order history, favourite restaurants, average order values, device used to order food, etc. The user feedback i.e., click/ no click on the advertisement, will act as a reward.

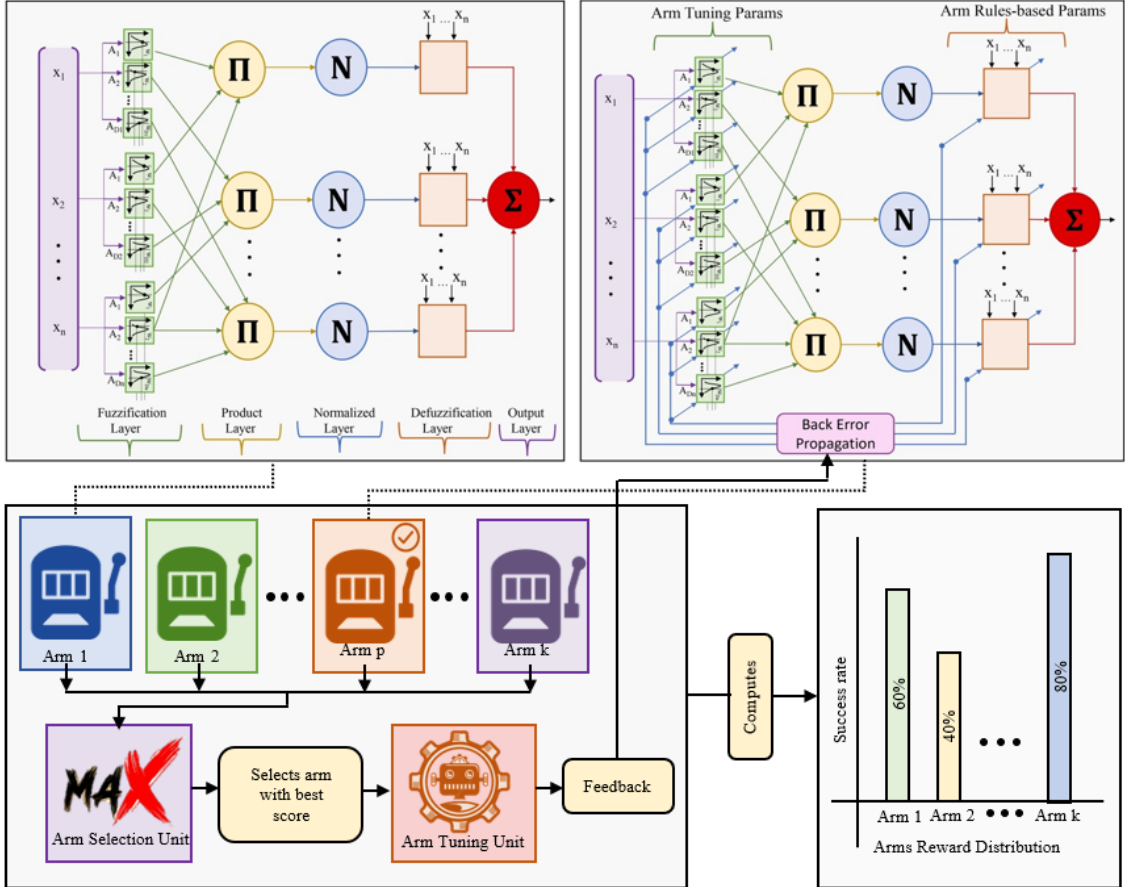


Figure 3.1: FuzzyBandit model

As depicted in Fig. 3.1, the developed model consists of k fuzzybandit arms where each FuzzyBandit Arm (FBA) corresponds to an action from an alternate number of choices/actions possible. Each FBA represents a standalone ANFIS model developed using the Sugeno fuzzy model [110][111] and inputs an n -dimension user context feature $x(t)$. Since the inputted context feature vector is vague or imprecise, $x(t)$ is represented with more than one fuzzy set with membership functions, $A_{11}, A_{21}, \dots, A_{Dn}$, to accommodate the possibility of more than one linguistic variable associated with the feature. For each membership function, \mathcal{F} maps the element of $x(t)$ to a value between 0 and 1 i.e. $\mathcal{F}(x(t)) \rightarrow [0, 1]$. We have used the gaussian membership function for each node as

shown in the fuzzification layer in Fig. 3.1 and is defined as:

$$O_{i,j}^1 = e^{-\frac{(x_i - a_{i,j})^2}{2b_{i,j}^2}} \quad \forall i \in [1, n], j \in [1, m] \quad (3.2)$$

where, $\{a_{i,j}, b_{i,j}\} \in R \times R$ are arm-tuning parameters associated with each node and get updated with each iteration. n is the number of features of the user context feature vector and m represents the number of membership function associated with each feature. The number of membership functions for each feature is data-dependent and is calculated experimentally. Next the real-time relevance scoring, $FBA_i(t)$ for each action associated with FBA is calculated using the FuzzyBandit Ranking Algorithm. The $FBA_i(t)$ score is calculated as:

$$FBA_i(t) = \sum_{j=1}^m W \cdot \beta \quad (3.3)$$

where, β will be $\sum_{i=0}^n (q_{j,i} \cdot x_i(t))$, $q_{j,i}$ are arm-rules based parameters and W is the output from the previous layer for each FBA. The arm-rule based parameters for each FBA is computed as the best solution for equation (3) which is minimizing the square error $\|Z - W\beta\|$. It is given as :

$$\beta^* = (W^T W)^{-1} W^T Z \quad (3.4)$$

where, W^T is the transpose of W and $(W^T W)^{-1} W^T$ is the pseudo inverse of W . The $FBA_i(t)$ score calculated in equation (4) is inputted into the FBA Selection Unit (FB-ASU). FB-ASU ranks each action, $a_i(t)$ corresponding to the arm, i based on the $FBA_i(t)$ score in the descending order i.e., the arm with the highest $FBA_i(t)$ the score gets the lowest rank. The lowest ranked arm is then considered as the best possible arm for the given $x(t)$ by the developed model and fed into the FBA Tuning Unit (FB-ATU). For each trial, FB-ATU checks whether the selected arm is the optimal arm or not and generates relevant feedback for network training. The feedback is the difference in the score calculated by the proposed FuzzyBandit model and the actual score using the recursive least square estimator (RLSE) method. In case the selected arm is the optimal arm, a positive feedback is generated and backpropagated to the selected arm. This is done to ensure that next time the model is presented with the same user context feature vector, the previously selected arm $FBA_i(t)$ score is higher than its peer arms. Hence, exploiting the same action/choice as by the user in the past. Also, if the arm outputted by the FB-ATU is not the desired

arm, negative feedback is not generated to minimize the score of the selected arm. Arm Tuning based parameters are trained using the derivation as proposed by Jang [112].

3.2.1.2 Ranking Algorithm

Algorithm 1 FuzzyBandit Ranking Algo

Input: Given data-set of the training sample $\{x(t), t\}_{t=1}^n \in \{R^d \times R\}$

```

1: Initialize  $ANFIS_1(a_1, b_1) \dots ANFIS_k(a_k, b_k)$  for each arm  $\triangleright a_k, b_k$  are arm-tuning parameters
2: for  $t = 1, 2, \dots, T$  do
3:   for  $arm, k = 1, 2, \dots, K$  do
     // FBA fuzzification
4:     for  $i = 1, 2, \dots, n$  do
5:       for  $j = 1, 2, \dots, m$  do
6:          $O^1[i][j] \leftarrow \exp[-(x_i - a_{i,j})^2 / 2 * b_{i,j}^2]$ 

     // FBA production
7:     for  $j = 1, 2, \dots, m$  do
8:       for  $i = 1, 2, \dots, n - 1$  do
9:          $O^2[j] \leftarrow \text{product}(O^1[i][j], O^1[i + 1][j])$ 

     // FBA normalization
10:    for  $j = 1, 2, \dots, m$  do
11:       $O^3[j] \leftarrow \text{divide}(O^2[j], \text{totalsum}(O^2))$ 

     // FBA defuzzification
12:    for  $j = 1, 2, \dots, m$  do
13:       $O^4[j] \leftarrow \text{product}(O^3[j], F(x_1, x_2, \dots, x_n, q_0, q_1, \dots, q_n))$   $\triangleright q_0, q_1, \dots, q_n$  are arm-rule based
      parameters and F is linear function
14:       $FBA\_score[k] \leftarrow FBA\_score[k] + O^4[j]$ 
15:       $k = \text{argmax}_{1 \leq i \leq K} (FBA\_score[i])$ 
16:      display action  $a(t)$  corresponding to  $k$  to the user; record clicks
17:      if user clicked  $a(t)$  then
18:        update ( $ANFIS_k$ , reward  $\leftarrow 1$ )
19:      else
20:        continue

```

Figure 3.2: FuzzyBandit ranking algorithm

The developed ranking algorithm, [3.2], is explained in three stages, viz., FBA_fuzzification, FBA_integration and FBA_defuzzification. In the first stage, FBA_fuzzification computes fuzzy membership value associated with the user contextual feature vector for each FBA and stores it into the matrix, O^1 , of dimension $n \times m$ where n is the number of features of the user contextual feature vector and m is the number of membership functions associated with each feature. The second stage, FBA_integration, first combines the mem-

bership of all features computed from the previous stage, by multiplying the membership values computed in the matrix, O^1 , and then normalizing the membership of each feature as each feature is represented with more than one membership function. Lastly, the FBA_defuzzification stage returns the crisp output score FBA_score for each FBA. To calculate FBA_score , the normalized weights calculated in FBA_normalization stage are multiplied with a function, F , and stored in a matrix O^4 . F is a linear function of arm-rule based parameters and the user context feature vector. The same can be visualized in the defuzzification layer (as shown in **Algorithm 1**) which multiplies the inference of normalized firing strength of each node and the first-order polynomial of user contextual feature vector and arm-rule based parameters. The score for each FBA is then the summation of matrix O^4 and stored in FBA_score . Each FBA is ranked in descending order of the FBA_score . The arm with the lowest rank is selected and the action corresponding to the selected arm is displayed to the user. If the user clicks on the displayed action, then the action selected by the model is correct and positive feedback is sent back i.e. exploiting the choice for the next iteration. If the selected arm is not clicked by the user, then no feedback is sent back which allows the proposed model to explore new actions in the next iteration.

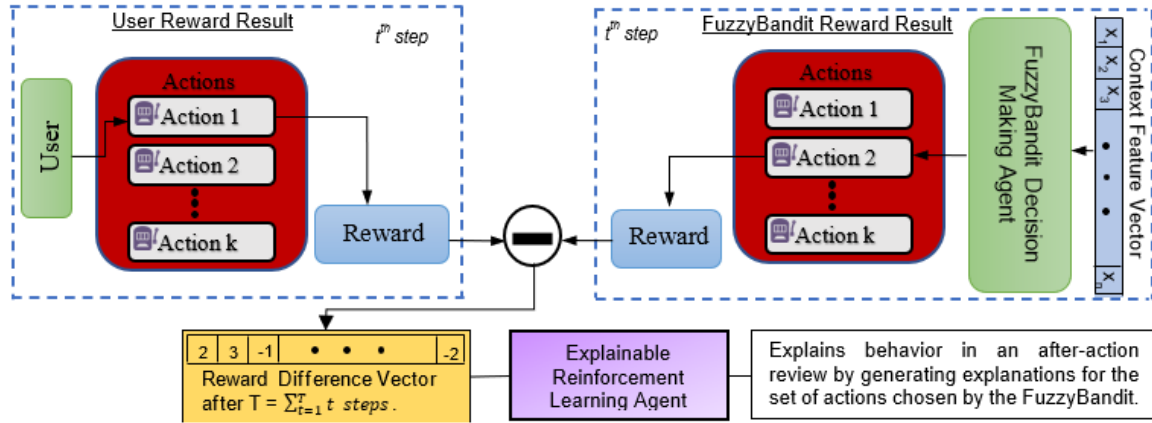


Figure 3.3: XAI model for FuzzyBandit

3.2.2 XAI

After computing FuzzyBandit used reward difference, the developed scheme incorporates XAI that the model learns by itself without any human intervention and a user can also easily understand the decisions made by the model. This enables the proposed model to

be transparent and perform autonomously. Thus the model's decisions in the real-world environment can be trusted by the user; thereby allowing the model to be deployed at scale.

Fig. 4.1 shows how the explanations are generated by the model using FuzzyBandit-User ("FB-U") reward difference. The FuzzyBandit model inputs an n -dimensional context vector $b_i(t) \in \mathcal{R}^d$ associated with each arm i at time t and yields a random reward $r_a(t)$ with unknown distribution $\theta_i(t)$ for the corresponding action $a_i(t)$ chosen. If the user disapproves of the action chosen by the model or seeks an explanation as to why $a_i(t)$ is chosen, then the user chooses an action $\tilde{a}_i(t)$ preferred over the action $a_i(t)$ by the user and observes the reward, $\tilde{r}_{(\tilde{a}_i(t))}(t)$.

Let the optimal arm at time t be $a^*(t) = \operatorname{argmax}_{(1 \leq i \leq N)} \{\theta_i(b_i(t))\}$. The FB-U reward difference vector can be computed as

$$\delta(r, a_i(t), \tilde{a}_i(t)) = r_{(a_i(t))}(t) - \tilde{r}_{(\tilde{a}_i(t))}(t) | \{b_i(t)\}_{(i=1)}^T \quad (3.5)$$

$$= \theta_t(b_{(a_i(t))}(t)) - \theta_t(b_{(\tilde{a}_i(t))}(t)) \quad (3.6)$$

$$FB - U = \delta(r, a(t), \tilde{a}(t)) \quad \forall t \in \{1, 2, \dots, T\}$$

For $t = 1, 2, \dots, T$ steps, the total reward difference $R(T)$ can be calculated as

$$R(T) = \sum_{t=1}^T r_{(a_i(t))}(t) - \tilde{r}_{(\tilde{a}_i(t))}(t) | \{b_i(t)\}_{(i=1)}^T \quad (3.7)$$

If the total reward difference, $R(T)$ is greater than zero, that means the total reward generated by the FuzzyBandit model is more than that of the reward generated by the user. A high positive magnitude of $R(T)$ shows the decisions made by the developed FuzzyBandit model are much higher reward yielding as compared to the decisions made by the user and thus enables humans to understand the decisions or predictions made by the proposed model. In order to quantify this, each component Δ_c (FB-U) can take numerically either 0, -1, or +1. Logically, each component of FB-U signifies either a positive or negative reason for choosing action $a_i(t)$ over user-generated action $\tilde{a}_i(t)$. Since the FB-U vector is computed over T steps, it can become overwhelming for the human to visualize each action preference individually, so in order to simplify this, two sets of positive and negative reasons i.e., $FB-U^+$ and $FB-U^-$, is computed. Mathematically,

$$FB-U^- = \sum_c I[\Delta_c(\delta(r, a(t), \tilde{a}(t))) < 0] |\Delta_c(\delta(r, a(t), \tilde{a}(t)))| \quad (3.8)$$

$$FB-U^+ = \sum_c I[\Delta_c(\delta(r, a(t), \tilde{a}(t))) > 0] \Delta_c(\delta(r, a(t), \tilde{a}(t))) \quad (3.9)$$

where, I is the identity function. The confidence score, α , is computed using FB-U reward difference which signifies by how much percentage the user can trust the FuzzyBandit model decisions in real-time settings. Alternatively, α represents the total percentage of the FuzzyBandit model's decisions (or actions chosen) better than the user's decisions. Similarly, β is the total percentage of the user's decisions better than that of the Fuzzy-Bandit model. They can be computed as follows:

$$\alpha = \frac{\text{FB-U}^+}{(\text{FB-U}^+ + |\text{FB-U}^-|)} \quad (3.10)$$

$$\beta = \frac{\text{FB-U}^-}{(\text{FB-U}^+ + |\text{FB-U}^-|)} \quad (3.11)$$

In the above equations, α and β act as the trust factor for the user as they quantify that by how much value the FuzzyBandit model's decisions are better or worse than the user's decisions. This enables developed FuzzyBandit model to be deeply coupled with explainable AI to yield insight into complex model decisions and deployed autonomously without any human interventions.

3.3 Results and Discussion

The developed model is validated on four publicly available datasets, namely the Adult Income, Forest Coverttype, Mushroom, and Statlog (Shuttle) from the UCI Machine Learning Repository [113]. These datasets have been widely used in the literature as benchmark datasets to measure the performance of various contextual multi-arm bandit algorithms. The objective of this experiment is to test the suitability of the FuzzyBandit model by combining various criteria for the datasets.

Nine criteria [114], namely recall, specificity, precision, prevalence, F1 score, Matthews correlation coefficient (MCC), Fowlkes–Mallows index (FM), critical success index (CSI), accuracy are considered, as summarized in Table 3.1 and Table 3.2. These criteria [114] are important as the number of observations for each arm/choice varies significantly in the dataset, and thereby, relying alone on accuracy can be misleading. For example, in the mushroom dataset, there are 3916 instances where the mushroom is edible and 4084 instances where the mushroom is poisonous, and the algorithm has to predict out of the two choices, i.e., mushroom is either edible or poisonous. A particular algorithm/model might predict all the observations as poisonous, giving an overall accuracy of 51%, but in more detail, the algorithm has 100% sensitivity for the poisonous class but a 0% sensitivity for the edible class.

Table 3.1: Performance analysis of contextual bandit models on the mushroom database

Parameters	Uniform Sampling	RMS	Dropout	BootRMS	ParamNoise	BBAAlphaDiv	FuzzyBandit	Banditron
Recall	0.502	0.829	0.735	0.873	0.739	0.818	0.851	0.669
Specificity	0.504	0.756	0.695	0.826	0.708	0.834	0.853	0.668
Precision	0.521	0.785	0.722	0.843	0.731	0.841	0.862	0.684
Prevalence	0.518	0.518	0.518	0.518	0.518	0.518	0.518	0.518
F score	0.511	0.809	0.727	0.856	0.735	0.829	0.855	0.674
MCC	0.006	0.588	0.431	0.701	0.4469	0.652	0.703	0.337
FM	0.260	0.650	0.530	0.733	0.540	0.690	0.730	0.450
CSI	0.340	0.680	0.570	0.751	0.581	0.710	0.748	0.510
Overall Accuracy	50.31	79.39	71.59	85.03	72.39	82.57	85.18	66.84

Table 3.1 provides a detailed comparison of several contextual bandit models based on various performance metrics when applied to the mushroom database. Each row of the table presents a different metric, while each column lists the performance of a specific model. The models compared include Uniform Sampling, RMS, Dropout, BootRMS, ParamNoise, BBAAlphaDiv, FuzzyBandit, and Banditron. The performance metrics are discussed in detail individually as follows,

Recall, also known as sensitivity, measures the model's ability to correctly identify positive cases. The BootRMS model achieves the highest recall at 0.873, indicating that it is the most sensitive in detecting positive instances within the mushroom dataset. This suggests that BootRMS has a strong capacity to avoid false negatives, making it reliable for situations where missing a positive case could have significant consequences. On the other hand, Uniform Sampling shows the lowest recall at 0.502, which means it struggles significantly in identifying positive cases, possibly leading to many false negatives.

Specificity measures the model's ability to correctly identify negative cases. Here, the FuzzyBandit model performs best with a specificity of 0.853. This high value indicates that FuzzyBandit is highly effective in identifying true negatives, making it useful in scenarios where false positives need to be minimized. Dropout, however, has a lower specificity of 0.695, suggesting it may incorrectly label some negative cases as positive, thereby having a higher false positive rate.

Precision assesses the accuracy of the positive predictions made by the model, showing the proportion of true positive predictions out of all positive predictions. FuzzyBandit again emerges as a top performer with a precision score of 0.862, closely followed by BBAAlphaDiv at 0.841. This implies that when these models predict a positive case, they are correct over 80% of the time, reflecting their reliability in making positive predictions. Conversely, Uniform Sampling has the lowest precision at 0.521, indicating that nearly

half of its positive predictions are incorrect, which could lead to significant inefficiencies or errors depending on the application.

Prevalence, which remains constant across all models at 0.518, indicates that in the mushroom database, positive cases constitute approximately 51.8% of the total cases. This uniform prevalence suggests that the dataset is balanced in terms of positive and negative cases, providing a consistent basis for comparing the models.

The F1 score, which is the harmonic mean of precision and recall, provides a single metric to assess the balance between these two aspects of performance. BootRMS and FuzzyBandit exhibit high F1 scores of 0.856 and 0.855, respectively, indicating that these models strike a good balance between precision and recall. They are effective both at identifying positive cases and at making accurate positive predictions. Uniform Sampling, however, shows a significantly lower F1 score of 0.511, underlining its poor overall balance between recall and precision.

The MCC is a comprehensive measure that considers true and false positives and negatives, giving a more balanced view of the model's performance. FuzzyBandit leads with an MCC of 0.703, demonstrating its robustness in classification tasks. This high MCC suggests that FuzzyBandit is well-calibrated, with a good balance between all four types of outcomes. In contrast, Uniform Sampling has an MCC close to zero (0.006), indicating a near-random performance where it fails to provide meaningful predictions.

The FM index measures the geometric mean of precision and recall, offering another perspective on the trade-off between these two metrics. BootRMS achieves the highest FM index at 0.733, reinforcing its strong performance in maintaining a balance between sensitivity and precision. On the lower end, Uniform Sampling has an FM index of 0.260, reflecting its poor capability to balance recall and precision effectively.

The CSI measures the ratio of true positives to the sum of true positives, false negatives, and false positives. BootRMS and FuzzyBandit again score highly with CSIs of 0.751 and 0.748, respectively, suggesting that these models are highly efficient in capturing true positives while minimizing false positives and negatives. Uniform Sampling's CSI of 0.340 further highlights its inefficiency in these aspects.

Overall Accuracy provides a direct measure of the percentage of correctly predicted cases out of the total cases. FuzzyBandit has the highest overall accuracy at 85.18%, followed closely by BootRMS at 85.03%. This confirms their effectiveness across the board, making them the most reliable models for this dataset. On the other hand, Uniform Sampling has an overall accuracy of 50.31%, barely better than random guessing, underscoring its inadequacy in handling this classification task.

It can be concluded from Table 3.1 that BootRMS and FuzzyBandit are the leading models for the mushroom database, with high scores across almost all metrics. These models demonstrate a balanced performance with strong capabilities in identifying both positive and negative cases accurately, while also maintaining high overall accuracy. Uniform Sampling, by contrast, performs poorly in nearly every metric, suggesting it is not suitable for tasks requiring reliable predictions in this context.

Table 3.2: Performance analysis of contextual bandit models on the adult database

Parameters	Uniform Sampling	RMS	Dropout	BootRMS	ParamNoise	BBAAlphaDiv	FuzzyBandit	Banditron
Recall	0.499	0.758	0.773	0.770	0.747	0.673	0.742	0.613
Specificity	0.504	0.756	0.770	0.768	0.745	0.679	0.729	0.611
Precision	0.754	0.905	0.910	0.910	0.900	0.870	0.893	0.828
Prevalence	0.753	0.753	0.753	0.753	0.753	0.753	0.753	0.753
F1 score	0.599	0.823	0.836	0.834	0.816	0.759	0.809	0.704
MCC	0.002	0.459	0.487	0.482	0.438	0.308	0.420	0.194
FM	0.374	0.682	0.704	0.701	0.672	0.586	0.663	0.508
CSI	0.429	0.702	0.719	0.715	0.690	0.609	0.681	0.544
Overall Accuracy	50.02	75.74	77.2	76.93	74.67	67.48	73.87	61.23

Table 3.2 mirrors the structure of Table 3.1 but evaluates the performance of the same set of models on a different dataset, the adult database. The performance metrics are examined in detail, one by one, as follows:

In the adult database, Dropout emerges as the leader with a recall of 0.773, indicating it is the most effective model in identifying positive cases. This high recall suggests that Dropout is less likely to miss positive instances, making it valuable in contexts where capturing as many positive cases as possible is crucial. Uniform Sampling, with a recall of 0.499, performs poorly once again, struggling to detect positive cases effectively.

Specificity remains a critical metric for evaluating a model’s ability to correctly identify negative cases. Both Uniform Sampling and RMS achieve the highest specificity at 0.756, indicating that they are equally effective in minimizing false positives. Banditron, with the lowest specificity of 0.611, may have a higher tendency to incorrectly classify negative cases as positive, leading to potential overestimation of positive predictions.

Precision in the adult database is highest for Dropout at 0.910, followed closely by RMS at 0.905. This indicates that when these models predict a positive outcome, they are correct over 90% of the time, making them highly reliable in their positive predictions. ParamNoise, while still performing well with a precision of 0.900, does not match the precision of Dropout and RMS. Banditron, on the other hand, has a lower precision of 0.828, indicating a higher rate of false positives compared to the top models.

Similar to the mushroom database, the prevalence metric is consistent across all models in the adult database, at 0.753. This reflects that positive cases make up approximately 75.3% of the total cases in the dataset, providing a baseline for evaluating the models.

The F1 score, which balances precision and recall, is highest for Dropout at 0.836, indicating it maintains a strong balance between sensitivity and accuracy of positive predictions. RMS also performs well with an F1 score of 0.823, reinforcing its robustness across multiple metrics. In contrast, Uniform Sampling's F1 score of 0.599 reflects its continued struggle to effectively balance precision and recall.

MCC remains an important metric for assessing overall classification quality. Dropout achieves the highest MCC at 0.487, indicating it has a well-balanced performance across true positives, false positives, true negatives, and false negatives. Uniform Sampling, with an MCC of 0.002, suggests near-random performance, highlighting its inability to effectively differentiate between classes.

The FM index for Dropout is the highest at 0.704, suggesting that this model is well-suited for tasks requiring a strong balance between recall and precision. RMS follows closely with an FM index of 0.682, indicating consistent performance. Banditron, with the lowest FM index of 0.508, highlights its relative inefficiency in maintaining this balance.

CSI values are highest for RMS and Dropout, both scoring 0.719, indicating that these models are efficient in correctly identifying true positives while minimizing errors. Banditron, with a CSI of 0.544, is less efficient, reflecting its weaker performance in accurately identifying positive cases while controlling for false positives and negatives.

Overall accuracy in the adult database is highest for Dropout at 77.2%, making it the most reliable model in this context. RMS also shows strong performance with an accuracy of 76.74%. Uniform Sampling, with an overall accuracy of 50.02%, again highlights its inadequacy, performing no better than random guessing.

All the experiments are conducted in online settings, where the context is fed into the model and action is recorded. The model is then updated with the feedback observed. The performance of the developed FuzzyBandit model is tested on various criteria with the existing seven contextual bandit models. The first one is the uniform sampling model, a random policy model which ignores the context vector and chooses an arm randomly for each iteration. The second model is the Banditron model, which uses a simple perceptron model for exploitation and epsilon greedy as an exploration policy for each iteration. The third model is the Dropout model, a variation of the Neural Model (which uses each arm as a neural model for exploiting the best arm and epsilon greedy policy for exploration) and uses the dropout neural network to predict the best arm in each trial. Next is the RMS

model, which trains a neural network and chooses the action based on the highest score predicted, i.e., acts greedily for the current context. BootRMS model uses Bootstrap [115] to offer significant performance gain with respect to its parent RMS model.

The remaining models, namely BBAAlphaDiv [116] and ParamNoise [117], are non-linear models and use Thompson Sampling for exploitation. The results under various datasets are given below.

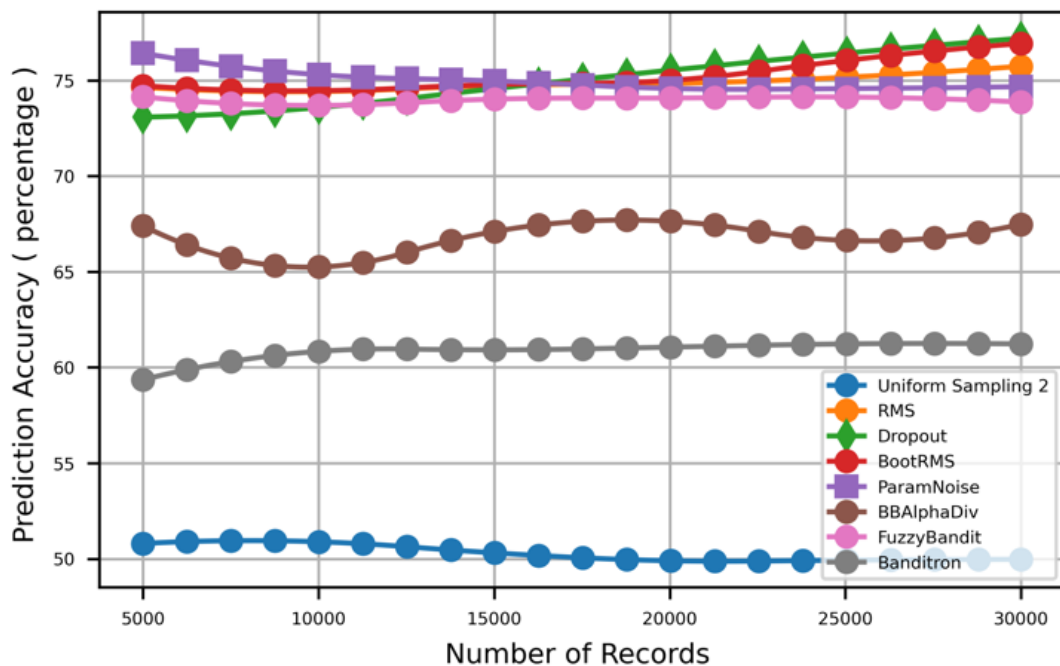


Figure 3.4: Comparison of contextual bandit models on adult dataset

- (a) **Adult income dataset:** Fig. 3.4 illustrates the prediction accuracy achieved by various contextual bandit models on adult dataset. The x-axis represents the number of records, ranging from 5,000 to 30,000, while the y-axis displays the prediction accuracy in percentage terms. Each curve on the graph in Fig. 3.4 corresponds to a different CMAB algorithm. Uniform Sampling 2 model shows the lowest prediction accuracy across all record counts, consistently hovering around 50%. The accuracy does not exhibit any significant improvement as the number of records increases, indicating that this method is not particularly effective at leveraging additional data to enhance prediction performance. RMS model starts at a higher prediction accuracy of approximately 70%. The curve is relatively stable, with only minor fluctuations, suggesting that this model maintains consistent performance as

the number of records increases, but it does not show substantial gains with more data. Dropout model demonstrates a similar pattern to RMS, with its prediction accuracy starting around 75%. The model maintains its accuracy level with only slight variations as the number of records increases. BootRMS model, similar to Dropout model, begins with a prediction accuracy close to 75%. BBAAlphaDiv model starts with the highest accuracy, about 78% but later its accuracy gradually fall to around 75%. FuzzyBandit begins with an accuracy close to 75% and maintains this level across all record sizes. This stability underscores its effectiveness in maintaining high accuracy regardless of the data volume. Banditron model starts with an accuracy of around 60% and shows a slight upward trend as the number of records increases. Although it does improve with more data, its performance is consistently lower than that of the higher-performing models like FuzzyBandit model. Fig. 3.4 can summarised as follows,

- (i) Models such as ParamNoise, FuzzyBandit, Dropout, and BootRMS consistently demonstrate high prediction accuracy, with little to no degradation in performance as the data size increases. These models appear to be well-suited for applications where maintaining high accuracy is crucial.
 - (ii) Uniform Sampling 2 is the least effective, with accuracy plateauing at around 50%.
- (b) **Forest covertype dataset:** Fig. 3.5 depicts the prediction accuracy (expressed as a percentage) of contextual bandit models observed on the number of records of Mushroom dataset. Each curve in Fig. 3.5 represents a different CMAB algorithms, with the x-axis indicating the number of records (ranging from 5,000 to 40,000) and the y-axis showing the prediction accuracy. It can be observed, Uniform Sampling 2 model consistently demonstrates the lowest prediction accuracy across all numbers of records. Its performance remains flat and does not show any significant improvement, staying around 10% regardless of the number of records. The RMS model starts with a prediction accuracy of approximately 15% with 5,000 records. As the number of records increases, the accuracy improves steadily, reaching about 55% at 40,000 records. Dropout model exhibits a similar pattern to RMS model, starting at around 23% accuracy. It shows a consistent upward trend, achieving close to 50% accuracy at 40,000 records. BootRMS model starts at a lower accuracy level than Dropout model but follows a similar trend. Its accuracy also improves with more

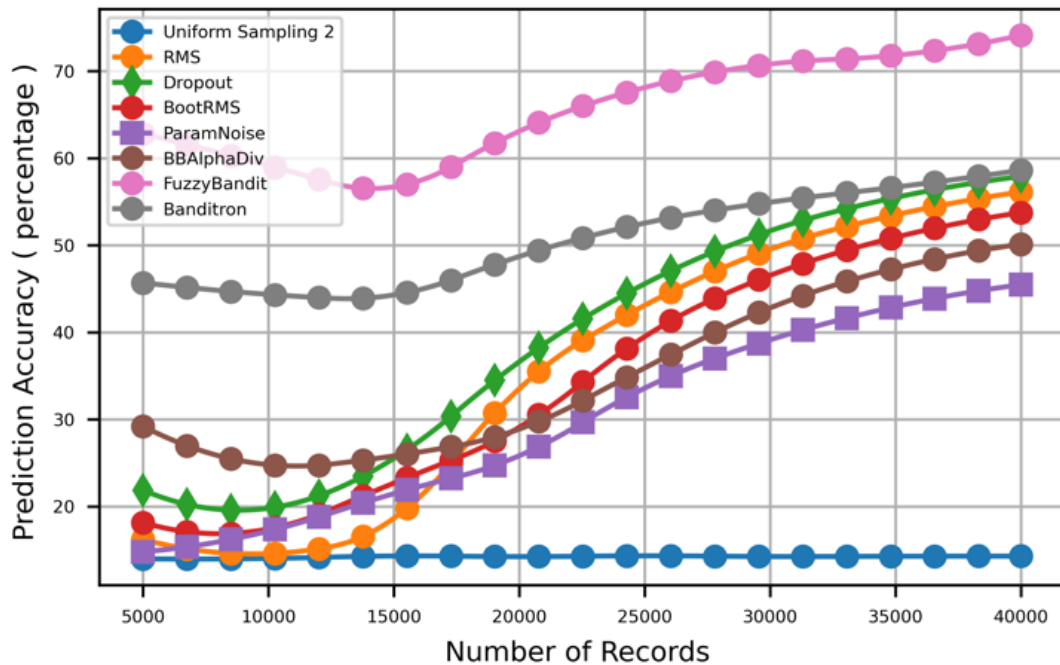


Figure 3.5: Comparison of contextual bandit models on covertype dataset

records, reaching approximately 53% at 40,000 records. ParamNoise model starts with lower accuracy compared to RMS and BootRMS. Its performance steadily improves, reaching around 45% accuracy at 40,000 records. BBAAlphaDiv model starts with a higher initial accuracy (around 30%) compared to the previously mentioned methods. It continues to improve with more data, reaching about 55% at 40,000 records, which is among the higher performing methods. FuzzyBandit model shows the best performance, starting at around 65% accuracy with 5,000 records and steadily increasing to about 74% at 40,000 records. This method consistently outperforms the others as more records are added. Banditron model starts at around 40% accuracy and exhibits a slower rate of improvement compared to FuzzyBandit but eventually reaches about 58% accuracy with 40,000 records. It performs better than most methods except FuzzyBandit. To summarize,

- (i) The FuzzyBandit model exhibits the highest performance, indicating its superior capability to leverage more data effectively. This suggests that FuzzyBandit is particularly well-suited for contexts where a large volume of data is available.

- (ii) Uniform Sampling 2 model's flat performance suggests it does not benefit significantly from additional data, indicating its potential limitations in scenarios requiring high accuracy.
- (iii) BBAlphaDiv and Banditron show competitive performance, especially as the number of records increases, making them viable alternatives depending on the specific application context.
- (iv) Most models show improvement as the number of records increases, highlighting the importance of data volume in enhancing prediction accuracy.
- (v) The banditron model, surprisingly, is the second-best performing model with an accuracy of around 48% – 52%.
- (vi) For applications where accuracy is critical, FuzzyBandit model appears to be the best choice among the methods compared, particularly as data availability increases.
- (vii) Fig. 3.5 clearly demonstrates that FuzzyBandit model consistently outperforms other CMAB models across varying data volumes, indicating its robustness and adaptability in improving prediction accuracy as more data becomes available.

(c) **Mushroom Dataset:**

Fig. 3.6 shows the prediction accuracy achieved by various contextual bandit algorithms on mushroom dataset. The x-axis represents the number of records ranging from 1,000 to 8,000, while the y-axis indicates the prediction accuracy in percentage. Each line in the graph corresponds to a different model, as indicated by the legend. Uniform Sampling 2 model consistently shows the lowest prediction accuracy, starting at around 50% and barely increasing throughout the range of records. The RMS model starts at an accuracy of about 52% and shows a steady increase as the number of records grows, peaking at around 88% near the end. Dropout model begins at lowest accuracy of approx 45% but then shows a pronounced upward trend, reaching around 78% accuracy with 7,500 records. This indicates that dropout model is particularly effective at improving prediction accuracy as more data becomes available. BootRMS model follows a similar pattern to Dropout, starting at around 60% and gradually increasing to approximately 92%. ParamNoise starts with about 70% accuracy and experiences a steady increase, reaches highest

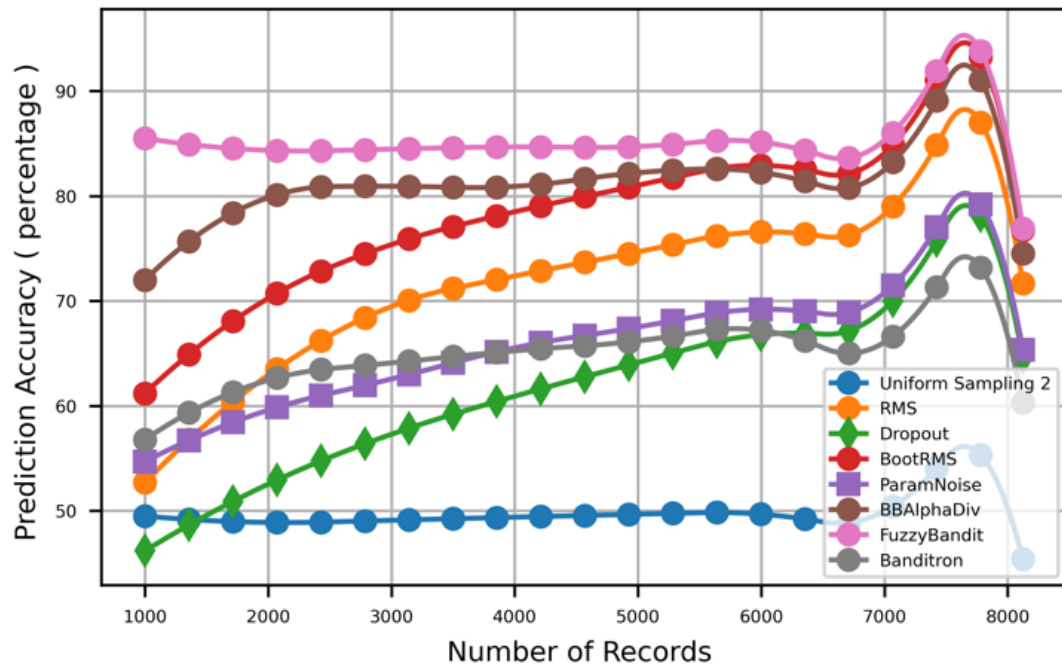


Figure 3.6: Comparison of contextual bandit models on mushroom dataset

accuracy of about 90% at 7,800 records. FuzzyBandit model begins with a high accuracy of around 85% among all the models and maintains this level throughout the range, peaking at nearly 95%. Fig. 3.6 can be summarized as follows,

- (i) All the models follow a similar trend. They initially shows strong performance, reach a peak and then suffers a noticeable drop in accuracy as the number of records exceeds 7,000. This decline can be due to overfitting or imbalanced dataset.
 - (ii) models like Dropout and RMS show a huge variation in accuracy as the more data is available. On the other hand, Uniform sampling 2 model shows least variation in accuracy as more number of records are processed and plateaus around 50% accuracy.
 - (iii) FuzzyBandit model starts with the highest accuracy and its accuracy gradually increases as the more data is processed, thereby making FuzzyBandit as the best model even for applications where less data is available.
- (d) **Statlog dataset:** The prediction accuracy of different CMAB models on statlog dataset can be seen in Fig. 3.7. X-axis shows the number of number of records

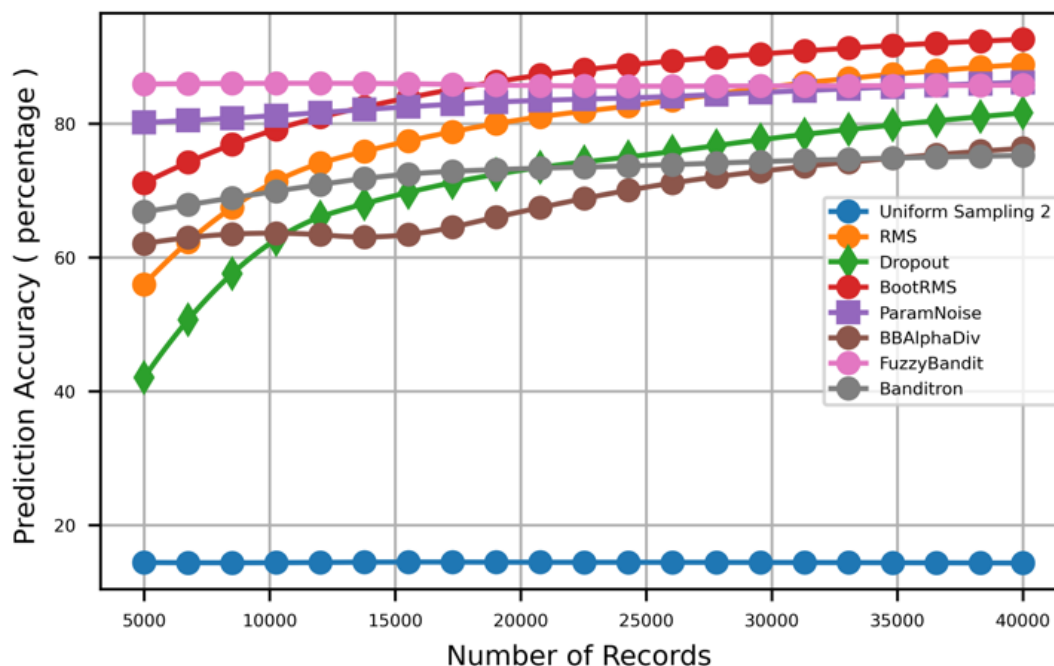


Figure 3.7: Comparison of contextual bandit models on statlog dataset

in statlog dataset, ranging from 5,000 to 40,000. Each curve represents a CMAB model in the Fig. 3.7. Uniform Sampling 2 model maintains a consistent prediction accuracy of around 10%, unaffected by the increase in the number of records. RMS model shows a steady increase in accuracy as the number of records increases, starting around 57% and reaching close to 85%. BootRMS model achieves one of the highest accuracies, quickly reaching around 87% as more records are available for processing. Paramnoise model starts at 80% accuracy and maintains this steady accuracy rate. FuzzyBandit model performs similar to Paramnoise model but with higher accuracy of about 85%.

3.4 Conclusion

This chapter presents a new FuzzyBandit model for the contextual multi-arm bandit problem. Each arm in the FuzzyBandit model mimics Adaptive Neuro-Fuzzy Inference System (ANFIS) independently by adjusting arm tuning and arm-rules based parameters according to user choice. The model can easily tune all its parameters by backpropagation and can be successfully trained in both offline and online settings for real-time applica-

tions where the rewards and data distribution can be dynamic. A comparative study of the contextual bandit's models is conducted on nine criteria. It is observed that none of the models has outperformed the other remaining models on different criteria in all four datasets. However, the developed Explainable artificial intelligence (XAI) FuzzyBandit model has shown promising results on all the datasets.

Chapter 4

Hybrid-Neuro Bandit Model

4.1 Introduction

Contextual Multi-Arm Bandit (CMAB) is a popular framework for sequential decision-making problems where an agent must repeatedly choose among multiple actions, each with an unknown reward distribution. The CMAB agent aims to maximize its cumulative reward over a finite or infinite horizon. At the same time, a high level of accountability is required, and there is a need to understand the underlying mechanism so that the user can trust the model's decisions. Therefore, in Chapter-4, a novel Hybrid Neuro Bandit (HNB) model is developed, that infuses the expert advice from the existing contextual multi-arm bandits into one combined unit, thereby exploiting the different CMAB algorithm merits and providing personalized recommendations to the user's liking. The proposed HNB model decisions can be easily understandable by the user as the HNB agent ignores the non-performing bandit experts and considers the opinion of the majority of the bandit experts. The HNB model has been empirically compared with the existing state-of-the-art contextual bandit models over nine performance metrics, namely recall, specificity, precision, prevalence, F1 score, Matthews correlation coefficient (MCC), Fowlkes–Mallows index (FM), critical success index (CSI) and accuracy.

The rest of the chapter is organized as follows. Section 4.2 presents a detailed study of the developed HNB model to predict arm reward distribution. In Section 4.3 analyzes the performance of the developed methodology. It also compares the performance of the developed model with the traditional state-of-the-art contextual bandit algorithms. Finally, the conclusions are drawn in Section 4.4.

4.2 Development of Hybrid-Neuro Bandit (HNB) Model

In this section, a novel Hybrid HNB model is developed that optimizes decisions for each user based on the previous user’s likings and preferences in an online setting. This enables the developed model to run on the fly with minimal training, thereby making the model fast, optimal (fewer resources are required), and usable in various real-time applications where much training data is unavailable. At the same time, the user can also understand the decisions made by the HNB model simply and intuitively, thereby allowing the user to trust the model’s decisions.

4.2.1 HNB Model

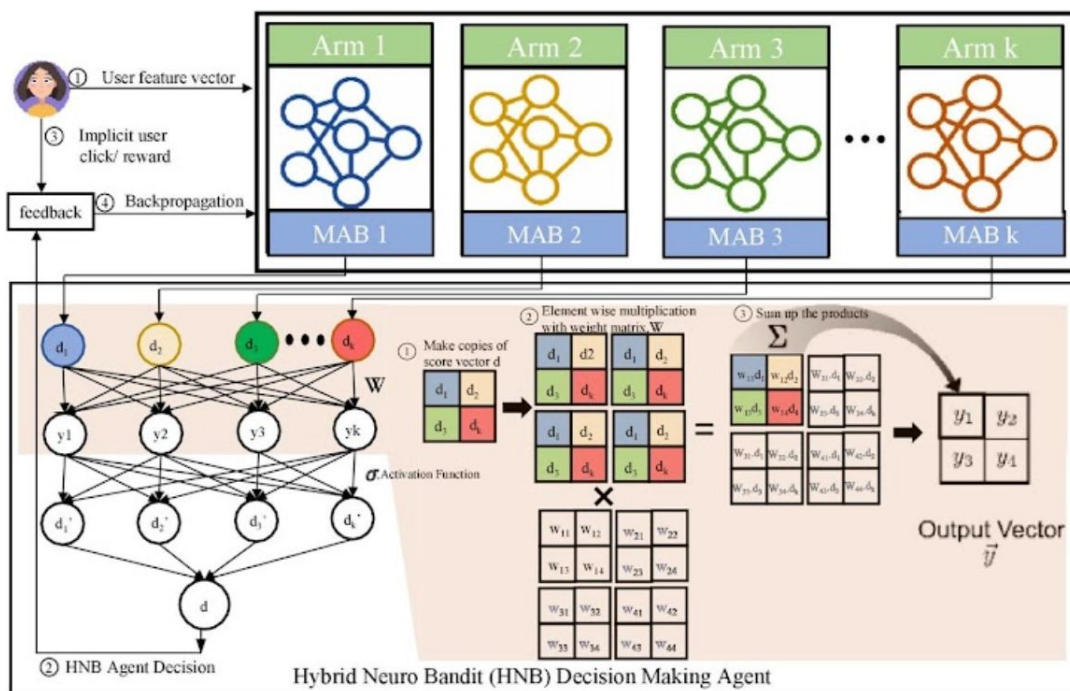


Figure 4.1: HNB model

As shown in Fig. 4.1, the developed model consists of k arms, where each arm corresponds to an existing contextual multi-arm bandit algorithm and acts as the bandit expert. Each bandit expert has merit and can be chosen based on their proven efficacy in the literature. The HNB model amalgamates these bandit algorithms into a cohesive unit, leveraging their respective merits and properties to overcome the challenges encountered by individual bandit algorithms. The amalgamation is done by fusing the decision made

by each bandit expert by assigning the weight to each bandit expert's opinion; in turn, this allows the HNB agent to ignore the opinion of any bandit expert in the final decision. This ensures that if any particular bandit expert consistently provides inaccurate predictions or underperforms in multiple trials, the HNB model disregards the input from that non-performing expert. Instead, it gives more weight to the opinions of other bandit experts that demonstrate better performance, thereby enhancing the overall accuracy and reliability of the model.

A crucial element for effectively achieving this is optimally and verifiably training the weights assigned to bandit experts. This is significant not only for explaining the decisions of the HNB model to the user but also for making the model interpretable. This transparency allows the HNB model to be accountable for its decisions, fostering user trust—an essential quality often absent in many existing state-of-the-art bandit models.

To understand the HNB model agent's decision, let us consider a training sample snippet as shown in Fig. 4.2. decision vectors d_1 , d_2 , and d_3 represent the decisions outputted by the existing bandit algorithm experts in the proposed HNB model (also called bandit experts). The ud vector represents the user's liking of the choice made and is represented as one if liked; otherwise, 0. Similarly, if the individual bandit expert decision is correct, i.e. the choice predicted by the bandit expert is the same as the user liking the corresponding decision vector, say d_1 is 1, otherwise 0.

At the beginning of the HNB model network training, let the initial weights and bias be zero, and the value of learning rate i.e. $a = 0.5$. 'z' will be the output of neuron i.e.

$$z = \sum_i d_i \cdot w_i + \text{bias} \quad (4.1)$$

and the output of the developed model will be decided by a step activation function δ (if the value is less than zero then the output will 0; be otherwise it will be 1).

Let us examine the weight updates during the first epoch of training. The first instance that the perceptron processes is given as below.

$$\vec{d} = (\vec{d}_1, \vec{d}_2, \vec{d}_3) = (0, 0, 0)^T \quad (4.2)$$

In this instance, the perceptron's net input is:

$$z = \mathbf{w}^T \vec{d} + b = 0 \times 0 + 0 \times 0 + 0 \times 0 + 0 = 0 \quad (4.3)$$

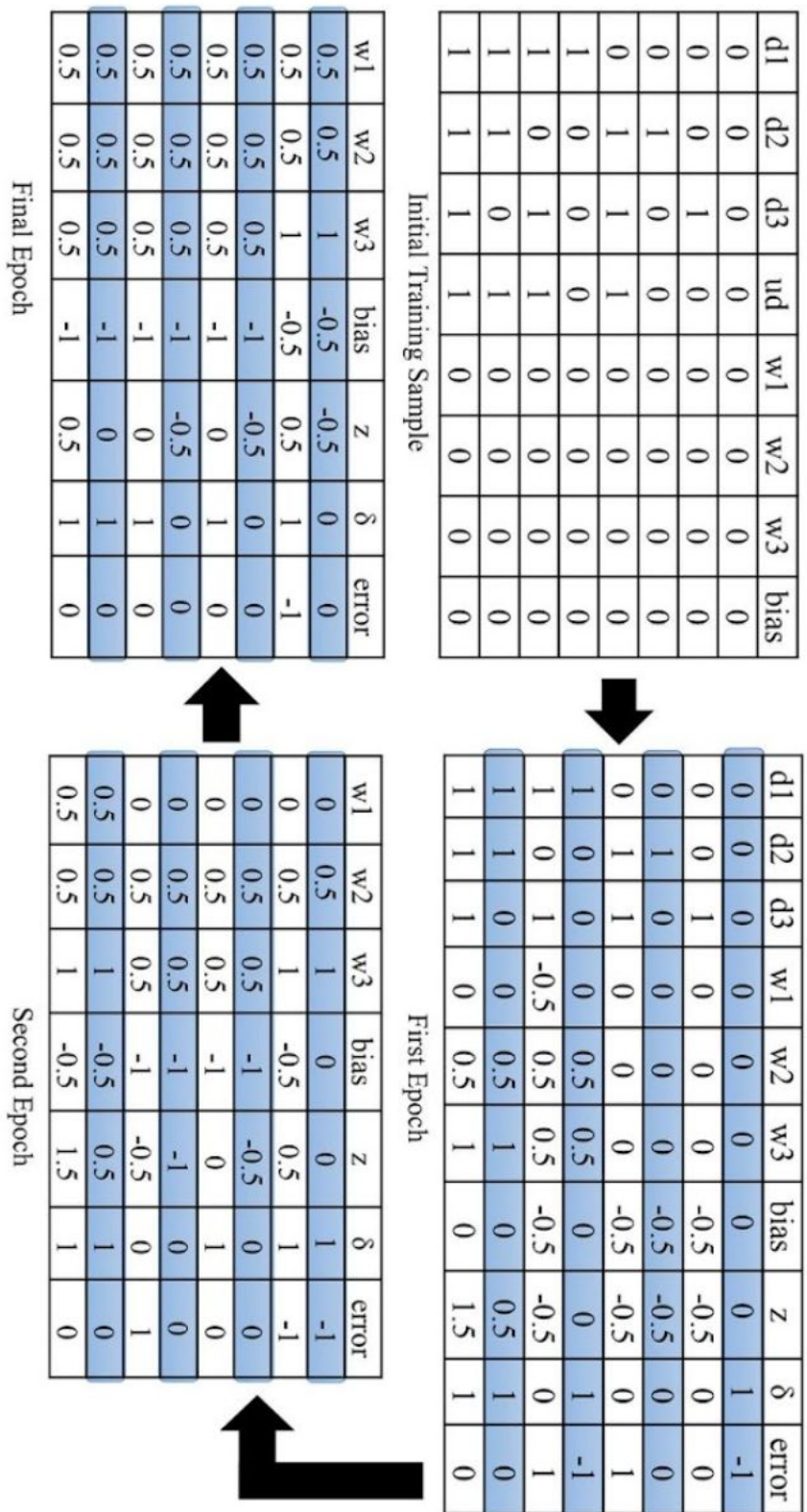


Figure 4.2: Instance of weight matrix update in HNB model

Thus, its output is $\vec{\sigma} = 1$, as the step function produces 1 when the input is ≥ 0 . However, the target label in this instance is $y = 0$, resulting in an error of $\vec{u} \cdot \vec{d} - \vec{\sigma} = -1$ for the perceptron.

Following the perceptron learning rule, each weight w_i is updated by adding as per Equation 4.4, to it.

$$\alpha(\vec{d} - \vec{\sigma}) \cdot x_i = -0.5 \cdot x_i \quad (4.4)$$

Since all the inputs in this instance are 0, except for the bias neuron ($x_0 = 1$), only the bias is modified to -0.5 instead of 0.

The same procedure is repeated for the other seven training examples, and weight updates are shown in the first epoch table in Fig. 4.2. It can be seen in the epoch table that during the first epoch, the perceptron makes four errors, and the value of the weight vector and bias (b) after the first epoch are as follows.

$$\mathbf{w} = \begin{pmatrix} 0 \\ 0.5 \\ 1 \end{pmatrix}^T \text{ and } b = 0 \quad (4.5)$$

During the second epoch of the training sample, similar operations are performed by the perceptron. This time, the perceptron only makes three errors. After the second epoch, the weight vector and the bias (b) are as per Equation 4.6.

$$\mathbf{w} = \begin{pmatrix} 0.5 \\ 0.5 \\ 1 \end{pmatrix}^T \text{ and } b = 0.5 \quad (4.6)$$

After updating the second example in this epoch, the perceptron converges to the weight vector that solves this classification problem as follows.

$$\mathbf{w} = \begin{pmatrix} 0.5 \\ 0.5 \\ 0.5 \end{pmatrix}^T \text{ and } b = -1 \quad (4.7)$$

Since all the weights are equal, the perceptron only fires when at least two of the inputs are 1, resulting in their weighted sum being greater or equal to 1, which is greater or equal to the absolute value of the bias (-1), thereby ensuring that the net input of the perceptron is non-negative.

4.2.2 HNB Ranking Algorithm

Algorithm 1 Hybrid Neuro Bandit (HNB) Algorithm

 Input: Given data-set of the training sample $\{x(t), t\}_{t=1}^N \in \{R^d \times R\}$

```

1: Initialize  $MAB_1, MAB_2, \dots, MAB_k$  for each arm      ▷  $MAB_i$  is any existing multi arm bandit algorithm
2: for epoch  $i = 1, 2, \dots$  do
3:   for  $t = 1, 2, \dots, N$  do
      // Calculate the decision vector,  $\vec{d}$  from the initialized  $MAB$  experts
4:     if  $MAB_i$  decision is correct then
5:        $d_i \leftarrow 1$ 
6:     else
7:        $d_i \leftarrow 0$ 
      // Input the decisions predicted by the MAB experts into Neural Network
8:     Initialize random weight matrix  $W$  for the  $MAB$  expert's decision
9:      $HNB\_decision[t] \leftarrow feedforward(\vec{d}, W)$       ▷  $z = sigmoid(d.W + bias)$ 
10:    get HNB agent action,  $\vec{a}(t) \leftarrow HNB\_decision[t]$ 
      // display action chosen by the HNB agent to the user chosen action,  $a(t)$ 
11:    if  $\vec{a}(t) = a(t)$  then error = 0
12:    else
13:      error = 1
14:    update( $W$ , error)
  
```

Figure 4.3: HNB ranking algorithm

The HNB model simulates contextual multi-arm bandit (CMAB) model settings where it inputs a finite n -dimension user context feature vector, $\mathbf{x}(t) = \{(x_1(t), x_2(t), \dots, x_n(t))\}$, and chooses an action, a_t , from an alternate number of choices/actions for each trial t . Each action is associated with a reward unknown to the model and is revealed after the action is chosen. The developed model observes a binary reward, $r_{at}(t)$, i.e., +1 if the user accepts the rendered action; otherwise, 0. In order to maximize the cumulative rewards over total trials, the HNB model uses a Hybrid Neuro Bandit Ranking Algorithm, as shown in Fig. 4.3. The HNB model consists of k HNB arms where each HNB arm mimics a contextual multi-arm bandit model acting as a bandit expert and inputs an n -dimension user context feature \mathbf{x}_t . For each trial, each bandit expert outputs a decision vector \mathbf{d}_i which represents the final action chosen by the bandit expert to be rendered to the end user. The decision vector $\vec{d} = \{d_1(t), d_2(t), \dots, d_k(t)\}$ is then assigned a random weight matrix and is inputted into the HNB neural network. The HNB neural agent then chooses the best decision from all the inputted decisions suggested by the bandit experts and displays the final action to the user. If the user accepts the rendered HNB decision, no error is computed. Otherwise, feedback is sent back, which allows the proposed model to explore new actions in the next iteration.

4.3 Results and Discussion

The performance of the developed method is evaluated based on the two publicly available datasets: the Mushroom dataset and Adult Income from the UCI Machine Learning Repository [113]. These datasets have been widely used in the literature as benchmark datasets to measure the performance of various contextual multi-arm bandit algorithms. Table 4.1 shows the number of contexts, the number of context features and the number of choices/ actions in each dataset. The brief description of the dataset is mentioned below.

1. **Mushroom dataset:** The mushroom dataset consists of 8124 instances of mushrooms as recorded data with 22 feature columns. The first column of the dataset is the classified mushroom type of that instance as a binary number (0/1), stating whether the mushroom belongs to the edible or poisonous category. The other features include cap-shape, cap-surface, cap-color, bruises, odor, gill-attachment, gill-spacing, gill-size, gill-color and stalk-shape. The aim is to correctly predict the category of mushroom, i.e., edible or poisonous.
2. **Adult income dataset:** The adult income dataset consists of 14 features such as age, education, fnlwgt, workclass, education-num, marital-status, occupation, relationship, race, sex, capital-gain, capital-loss, hours-per-week and native-country of an adult that determines the income of an adult. There is a total of 32561 instances of recorded data where the last column of the dataset is the classified adult income of that instance as a binary number (0/1).

Table 4.1: Description of datasets used by HNB model

Datasets	# Contexts	# Features	# Actions
Mushroom	8124	22	2
Adult Income	32561	13	2

The original datasets have missing values and consist of categorical and numerical features that need pre-processing for better prediction and outcome. Firstly, all the rows with missing values are dropped from the dataset, and standardization is performed to convert categorical features into numerical ones. One hot encoding and Ordinal Encoding are the two most popular standardization techniques used. Ordinal Encoding maps each unique feature to an integer value and is used as a known ordinal relationship between categorical features in the datasets. It transformed the dataset with a mean of zero and

a standard deviation of one. Then, the anomalies and outliers are removed from all the datasets. For that, a correlation matrix of all the dataset’s features is made, and the least relevant features are removed. For e.g. the feature *fnlwt* is least relevant in the Adult dataset as the correlation value is negative and, thus, is safely dropped from the dataset. Also, all the entries in the adult dataset having various anomalies, such as context feature horizontal and vertical distance, can’t be negative and are removed. There are approximately 200 outliers found in the dataset. Lastly, the features are scaled to a given range (default at (0,1)) with the help of a MinMax/ Robust scaler to prevent the optimization from getting stuck in local optima and make training faster.

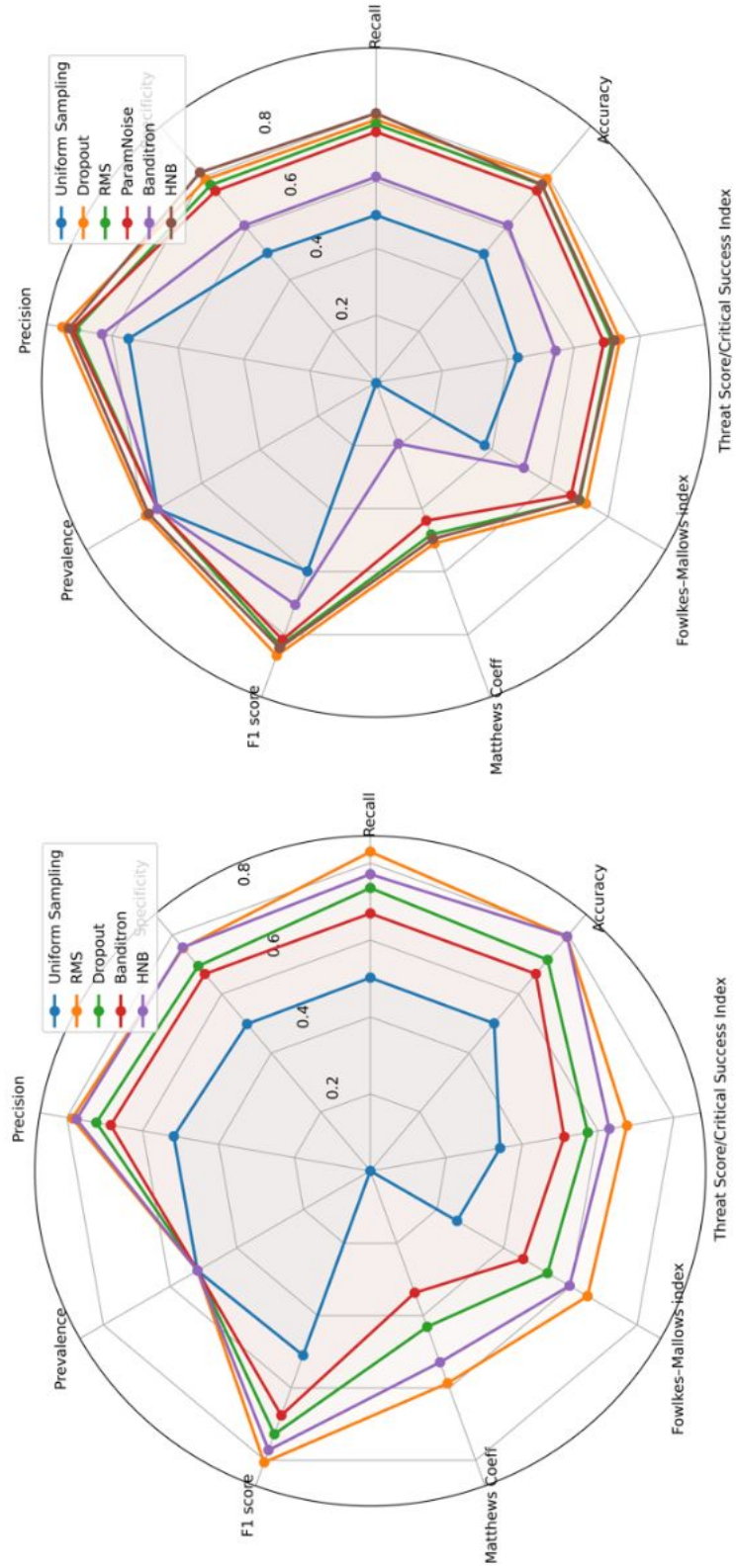
4.3.1 Experimental Results

In this section, the developed HNB model is compared with the existing CMAB algorithms; first is the uniform sampling model, which randomly makes the choice and ignores the context vector. Next are the banditron and dropout models, which use an epsilon greedy policy for exploration and multi-layer perceptron (MLP) variants to mimic each arm as a neural model to exploit the best choice. Lastly, the RMS model is chosen based on the highest score predicted, i.e., it acts greedily for the current context. The HNB model, alongside state-of-the-art contextual bandit models, is evaluated on ten performance metrics—accuracy, specificity, recall, precision, prevalence, F-score, Matthews Correlation Coefficient (MCC), Fowlkes–Mallows Index (FM), and Threat score. These metrics are crucial in informed decision-making, comprehensively assessing and optimizing targeted outcomes in diverse contexts and are explained below.

Table 4.2: Confusion matrix

		Actual Class	
		Positive	Negative
Predicted Class	Positive	True Positive (TP)	False Positive (FP)
	Negative	False Negative (FN)	True Negative (TN)

1. **Accuracy and regret:** Let $a^*(t)$ be the optimal arm which would yield the highest reward, $r_{a^*(t)}(t)$ and $r_{a(t)}(t)$ be the reward observed corresponding to the arm $a(t)$ chosen by the contextual bandit model for the user context feature vector x_t . Then, regret, as calculated in Equation 4.8, will be the cumulative sum of the difference between the maximum reward possible and the actual reward observed by



ii) Adult Dataset

i) Mushroom Dataset

Figure 4.4: Performance metric analysis of contextual multi-arm bandit algorithms on mushroom and adult datasets

the contextual bandit model over n trials, i.e.

$$\text{regret}(t) = r_{a^*(t)}(t) - r_{a(t)}(t) | x(t) \quad (4.8)$$

The overall goal is to minimize the regret. In applications such as an online advertisement model, the contextual bandit model renders a relevant advertisement, i.e. $a(t)$ to the user from the set of advertisements and if the user likes the chosen advertisement, the model gets a reward, $r_{a(t)}(t)$ as +1 otherwise 0. The maximum reward possible for each trial will be 1. In such applications, where the reward is either 0 or 1 based on whether the user liked the choice given by the model or not, the accuracy of the contextual bandit model can be defined as the ratio of the total number of choices made by the contextual bandit model liked by the user, i.e. correct predictions to the total number of the choices made or trials. Accuracy provides a straightforward measure of predictive success; however, it has a downside in that it can be misleading in scenarios with imbalanced class distributions. The accuracy can be calculated using Equation 4.9. The proposed HNB model has performed significantly better than the uniform sampling, dropout and banditron model on all the datasets. However, the HNB model has comparable accuracy with the RMS model.

$$\text{Accuracy} = \frac{\text{Correct Predictions}}{\text{Total Predictions}} \quad (4.9)$$

2. **Specificity:** Specificity is a crucial performance metric that provides insights into a model's ability to identify true negatives correctly. It is calculated using Equation 4.10. Fig. 4.4 shows that the HNB model has the highest specificity among the compared CMAB models on all the datasets. This signifies that the HNB model predicts fewer false positives, demonstrating its precision in identifying instances of the negative class.

$$\text{Specificity} = \frac{\text{Correctly Negatives Predicted}}{\text{Total Negative Data}} = \frac{TN}{TN + FP} \quad (4.10)$$

3. **Recall (Sensitivity):** Recall is the ratio of correct positives predicted over total positive data and can be successfully computed using Equation 4.11. The HNB model has the second highest recall value in the studied CMAB models, minimizing the risk of missing positive cases. Thus, the proposed HNB model captures and

recognizes a high number of true positive instances among other models.

$$\text{Recall} = \frac{\text{True Positives}}{\text{True Positives} + \text{False Negatives}} = \frac{TP}{TP + FN} \quad (4.11)$$

4. **Precision:** Precision is defined as the ratio of correct positives predicted over total positives predicted, as shown in Equation 4.13, reflecting the precision of the model in identifying positive instances. The proposed HNB model has better precision value than various CMAB models like dropout, banditron and uniform sampling and is useful in multiple applications where false positives carry significant consequences.

$$\text{Precision} = \frac{\text{Correctly Positives Predicted}}{\text{Total Positives Predicted}} = \frac{TP}{TP + FP} \quad (4.12)$$

5. **Prevalence:** Prevalence, within the realm of performance metrics, denotes the proportion of positive instances within a dataset or population in binary classification, as depicted in Equation 4.13. It is a critical factor influencing the interpretation of model performance. In scenarios with significant class imbalances, where one class prevails, prevalence impacts the reliability of metrics like accuracy. The developed HNB model scores high prevalence among its competing CMAB models, making it effective in real-world applications.

$$\text{Prevalence} = \frac{\text{Total Positive Data}}{\text{Total Predictions}} \quad (4.13)$$

6. **F-Score:** The F-score evaluates a model's performance by combining precision and recall and is valuable in scenarios where dataset balance is crucial, especially when one class is underrepresented. F-score is highly relevant in applications ranging from search engines to personalized recommendation systems, making it an essential performance metric. The F-score is calculated using Equation 4.14. From Fig. 4.4, it can be observed that uniform sampling has the minimum, and the RMS model has the highest F-score among all the models on all the datasets.

$$F \text{ score} = \frac{2 \cdot (\text{Precision} \cdot \text{Recall})}{\text{Precision} + \text{Recall}} \quad (4.14)$$

7. **Matthews Correlation Coefficient (MCC):** Unlike other metrics, such as accuracy or precision, which can be biased towards one class or another, MCC provides an

overall measure of classification performance that is representative of both positive and negative classes. As shown in Fig. 4.5, the output range of the MCC index is from -1 to 1, and the more it goes near 1 indicates better performance; the more it is around 0, it shows the random performance of the model, and the more value is around -1, it shows the performance is even worse than 0. MCC is usually used for imbalanced datasets, where the number of values for one classification is much more than the other. Because the MCC considers both true positives and negatives, it is unaffected by class imbalance and provides a more accurate performance measure for such datasets. From Fig. 4.4, the Uniform Sampling model has an MCC value of zero. In contrast, the RMS, HNB, Dropout and Banditron models have an MCC value near 1 in descending order on all the datasets.

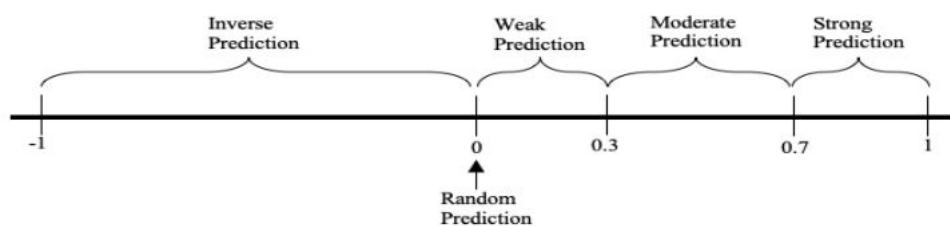


Figure 4.5: MCC index

8. **Fowlkes–Mallows Index (FM)**: It is a statistical measure designed to assess the quality of clustering or classification algorithms by quantifying the similarity between two sets of labelled data. FM index is computed with the help of equation (8), and the output range varies from 0 to 1, showing the similarity between the two sets. This makes FM particularly valuable in scenarios involving imbalanced datasets. Its use extends to applications where the identification of correctly clustered instances holds significance, such as image segmentation or biological data analysis. As an adaptable metric, the FM index contributes to the ongoing refinement of clustering algorithms, aiding in selecting and optimizing methods based on their ability to group similar instances accurately.

$$FM = \frac{TP}{\sqrt{(TP + FP) \cdot (TP + FP)}} \quad (4.15)$$

9. **Critical Success Index (CSI) or Threat Score (TS):** CSI is an invaluable tool in decision-making processes and provides a holistic view of success within the context of predefined critical factors. CSI can be successfully calculated using Equation 4.16. A higher CSI value signifies a more successful outcome, while a lower value may indicate areas that require attention and improvement.

$$CSI = \frac{\text{True Positives}}{\text{True Positives} + \text{False Negatives} + \text{False Positives}} \quad (4.16)$$

4.4 Conclusions

This chapter develops a novel Hybrid Neuro Bandit (HNB) that consists of multiple multi-arm bandit experts; each individually produces a decision based on the user feature vector and is fused in the HNB decision agent model. The vector difference between the agent's and the user's decisions is backpropagated to the HNB model to minimize regret and achieve high accuracy. Also, the underlying mechanism of the proposed HNB model is easy for the user to understand. It can be successfully trained in offline and online settings for real-time applications where the rewards and data distribution can be dynamic. Further, a comparative study of the contextual bandit's models on nine performance metrics, namely accuracy, specificity, recall, precision, prevalence, F-score, MCC (Matthews Correlation Coefficient), FM (Fowlkes–Mallows) index, and CSI (Critical Success Index), has been studied in detail on publicly available datasets. It is observed that none of the models has outperformed the other remaining models on different criteria in all the datasets. However, the HNB model has shown promising results on all the datasets.

Chapter 5

Contextual-POI-Bandit: A Contextual Multi-Arm based Framework for next-Points-of-Interest (POI) Recommendation

5.1 Introduction

With the ubiquity of mobiles, the check-in behaviour by Generation Z users has become a popular trend, as now they can easily share their experiences and reviews about various Points-of-Interest (POIs), e.g., cafés, restaurants, hotels, etc., along with their location, with friends on various Location-Based Social Networks (LBSNs). A check-in consists of a visited POI and other contextual information such as timestamps, comments, tags, and GPS coordinates. Millions of check-in data are collected in LBSN services (for example, Foursquare has over 10 billion check-ins from 55 million users) and provides an excellent opportunity to study the user's dynamic behaviour over time and to predict the next POI the user will most likely visit. This next POI recommendation is of great significance in various real-life applications; e.g., advertisers can use the next POI recommendation to effectively post ads on multiple pillars to maximize the number of views by the customers. Similarly, traffic authorities can collect vehicular mobility data through various surveillance cameras and positioning devices and effectively predict future vehicular movement on the road. By using this knowledge, traffic authorities can effectively manage traffic on the road as traffic lights can be designed dynamically to prevent congestion, and the public

may be informed in advance about the routes to avoid. However, in the realm of LBSNs, two major factors are pivotal in shaping how the next POI recommendation algorithms will perform: data sparsity in user-POI interactions and cold start problems for new users and POIs. There are millions of user check-ins available, but user interaction is limited to each POI. Further, these POIs are highly diverse in categories and locations with the infrequent exploration of these POIs by users. The result is a sparse matrix where the majority of entries are zero, indicating that users have checked into only a few POIs. This data sparsity poses a serious challenge as many traditional POI recommendation algorithms, such as traditional collaborative filtering [118] and matrix factorization methods [119], rely on sufficient interaction data to make accurate predictions. Without robust data on user preferences and behaviours across various POIs, capturing both temporal dynamics (such as changes in user preferences over time) and geographical dynamics (variations in popularity or accessibility of POIs) becomes inherently difficult. Cold start problem compounds these difficulties further. For example, when a new user joins the LBSN platform, there is insufficient historical data, i.e. previous check-ins, comments, reviews, etc, to predict the next POI as per the user's actual interests or preferences. Similarly, newly added POIs don't have any previous user visits and lack user interaction data to generate personalized recommendations. These challenges have been addressed extensively in the literature [120][121][122]; however, there is a need for innovative algorithmic approaches to effectively handle sparse interaction data and provide personalized recommendations even for new users and POIs.

5.1.1 Key Factors for Next POI Recommendation Model

It is pertinent to consider the following factors while designing and developing the next POI recommendation model.

1. The recommendation model must consider dynamic time intervals and geographical distance between neighbouring check-ins, also known as spatial-temporal intervals, to predict the user's next check-in accurately. Time and distance both play crucial roles in determining the users' check-in behaviour; for example, some office employees go to the gym after work, and some users will go to nearby restaurants from their offices for lunch; some tourists will sequentially check-in the popular places in a particular region. Thus, it is pertinent to analyze the user's historical behaviour of check-ins based on time and distance to recommend the next POI, as the user's former check-in trajectory may have a crucial impact on the next check-in.

Since there is a linear correlation between the user’s preferred check-in in the future and the check-ins made in the past, various Recurrent Neural Network (RNN)-based models [123] [124] [29] [125] have been proposed in the literature to exploit this property to explore the hidden intrinsic patterns in the user’s spatial-temporal check-in data. But unfortunately, they fail to fully capture correlation and dependencies over long periods. For examples,

- RNNs have a basic unit called a recurrent neuron, a self-loop structure that acts like a memory and remembers the information seen previously. This unit allows RNNs to process information effectively across multiple time steps and recognize sequential patterns in the user’s check-in data. Hence, RNNs are able to effectively determine short-term preferences or continuous activities in user POIs but fail miserably to exploit the relationships (long-term preferences) in a non-continuous sequence of user check-ins, be it in space or time.
- Studies [126] have extended RNNs capabilities with Gated Recurrent Unit (GRU) or LSTM to capture short-term and long-term preferences. The Spatial-Temporal Long-Short Term Memory (ST-LSTM) model [127] combines spatial-temporal influences into LSTM to mitigate data sparsity issues and present a hierarchical extension of the ST-LSTM, known as HST-LSTM [127], to model users’ historical visiting sequences in an encoder-decoder manner to enhance prediction performance. Spatio-Temporal Gated network (STGN) [128] enhances LSTM architecture by incorporating time gates and distance gates to model the spatio-temporal intervals between user check-ins to meet short-term and long-term user interests effectively for subsequent recommendations. Similarly, a Bi-LSTM-Attention model [129] was proposed to generate an attentional representation of the current user check-in sequence based on users’ long-term and short-term preferences. ST-RNN [130] also tries to overcome this problem by partitioning geo-spatial distances and time intervals into discrete bins but is unable to learn every continuous time interval and distance between neighbour check-ins, leading to poor recommendations. Time-LSTM [126], LSPL [131] and TMCA [132] extend LSTM to capture the dynamic preferences of the user for POI recommendation.

The majority of these models only take into account the variation between subsequent user check-ins as input to predict the next POI, whereas the next visited POI depends on both the distance from the current user's location and the intrinsic qualities of the POI. For instance, a user, after having dinner/lunch, may visit a cinema hall/ bar for entertainment rather than another restaurant (intrinsic characteristics of POI), but at the same time, the user may show less interest in a distant popular entertainment site (a geographical distance constraint). Also, the underlying real-world user check-in data is highly non-linear, and it is merely inaccurate to predict the next POI based on simple linear calculations. In short, the studied models fail to capture the high-order geographical influences while making recommendations. Moreover, most of these methods are designed for short-term preferences over POIs as they are perfectly modelled by the underlying RNN structure rather than long-term preferences, where short-term preferences mean the POIs recommended depend on the recently visited POIs by the user, and long-term preferences mean the recommended POIs depend on all historically visited POIs.

2. Most of the current research fails to address the problem of personalized recommendation, i.e., customized recommendations tailored as per the user's taste and liking. Personalization is crucial in designing the next POI recommendation model as different users have different preferences; some prefer short-term preferences, and some prefer long-term preferences; some prefer the combination of both short-term and long-term preferences. For example,

- Consider a user who likes to visit museums and historical sites on weekends, reflecting long-term preferences. On weekdays, after work, the same user likes to explore nearby cafes and trendy restaurants based on immediate mood and taste, which reflects the user's short-term preferences. Now, a model that fails to personalize will not consider the user's unique preferences and might fail to meet the user's expectations by suggesting the next place to visit.

Thus, it is pertinent to consider both long-term and short-term preferences while recommending where the next time user might visit, i.e. coffee shop recommendations during the week and historical spots for the weekends, thereby significantly enhancing the user experience and satisfaction.

3. Most existing models don't consider the user's context when recommending the next POI. A user's context consists of multiple components, such as the user's current activity, i.e. working, travelling or relaxing, and the user's demographic information, like age, gender, occupation, etc. Other relevant information like the user's current psychological and emotional state, social context (whether alone or with friends?; what are friend's likings? etc.), and environmental surroundings like local events and traffic situations also play crucial roles in deciding the next place a user might visit. For examples,

- Consider a user who likes to visit crowded places. Now, if the user wants to visit a new place but is under high stress due to work or an emotional breakup would most likely visit a calm place such as a park or meditation place for relaxation instead of a crowded place, and it is quite possible, that in the past user has mostly checked-in to social or crowded places. Further, most of the check-ins posted by the user are highly elusive or autonomous and don't represent the user's state at the time of posting.
- Consider another scenario where the user was undergoing depression or medical treatment at the time of check-in at various hospitals but has now fully recovered (i.e., the user no longer needs to visit the hospitals) and is now looking for entertainment places. Hence, a user might prefer different POI with different circumstances, and it is crucial to model the user's dynamic behaviour, i.e. what was once desired is no longer preferred.

Thus, the models must adapt to the user's new choices and tastes over time by incorporating user context effectively while suggesting new places to visit.

4. Most of the existing models in POI recommendation systems have predominantly employed an offline learning setting to analyze user preferences and suggest next-visit POIs. In an offline setting, models are trained using historical check-in data, and the learning process is static; once the model is trained, it relies on these static preferences to recommend future POIs. The primary advantage of the offline setting is that it allows the model to be fine-tuned on large historical data, thereby achieving high accuracy in predicting the next-visit check-in. However, the processing is computationally extensive, and the offline model fails to capture dynamic user

preferences and evolving activity patterns over time since it relies solely on historical user data. Consequently, there is a high probability that the next POI recommendation might not be relevant to the user’s current preferences, leading to user dissatisfaction over time.

5.1.2 Overview of Contextual-POI-Bandit

To address the above factor and challenges, an online interactive contextual multi-arm bandit (CMAB) based framework, Contextual-POI-Bandit, is an effective model for the next POI recommendation. Concretely, the framework consists of two modules:

- Knowledge graph (KG) module
- Contextual bandit (CB) module

In the KG module, the user’s Social (S), Geographical (G), Temporal (T), and Categorical (C) correlations among users and POIs are fused together into a unified user context feature vector matrix before making any suggestions. To incorporate social correlations, Contextual-POI-Bandit models the social check-in data by aggregating the check-in frequency or ratings of a user’s friends on a POI by power-law distribution and transforms it into a social relevance score for all the user’s POIs based on the historical check-in data of all users. Next, Contextual-POI-bandit models the geographical correlations between POIs by calculating a geographical relevance score using the kernel estimation method with an adaptive bandwidth. This method calculates the geographical relevance score by considering the spatial distribution of a user’s visited POIs and estimating the probability distribution over latitude and longitude coordinates. To model temporal information, Contextual-POI-bandit uses the Additive Markov chain [133] to effectively capture the sequential transition pattern between users and POIs. Lastly, to incorporate categorical correlations, Contextual-POI-Bandit, inspired by Zhang and Chow [134], uses power-law distribution to first evaluate a user’s biases towards specific POI categories based on the user’s past check-in data and then use these evaluated biases to calculate the weighted popularity of POIs within the same category and lastly, convert this weighted popularity into a categorical relevance score.

Finally, the Contextual-POI-bandit framework fuses these SGTC (Social, Geographical, Temporal, and Categorical) influences and their combined impact, such as the interplay between social and temporal factors or spatial and temporal factors, into a unified user vector, also known as user context. Contextual information significantly enhances

the performance of the next POI recommendation models. Various studies have integrated multiple contextual factors, such as geographical, temporal, categorical, and social information, into a unified model. For example, Baral and Li [135] developed a multi-aspect POI recommendation system using a graph-based model and the Top-Sensitive PageRank algorithm. Liu and Xiong [136] incorporated textual and contextual data using a Latent Dirichlet Allocation model for topic and location-aware recommendations. GeoSoCa [133] utilizes geographical, social, and categorical influences, employing kernel density estimation and power-law distributions for modelling these factors. Baral et al. [137] proposed a weighted matrix factorization model incorporating various aspects like geographical influence areas and temporal popularity. Other notable contributions include models by Yin et al. [138] leveraging item tags and co-occurrence patterns, Cheng et al. [139] focusing on users' localized movement constraints, and Zhang et al. [134] combining geographical, temporal, and social influences. Advanced methods include graph attention networks, probabilistic generative models, and adversarial models, all aiming to leverage contextual information to enhance POI recommendations. However, all these studies recommended the next POI based on the aggregated score of SGTC correlations and failed to consider the user's dynamic behaviour over time or provide personalized recommendations to cater to long-term preferences or short-term preferences effectively. Moreover, all of these models are trained in offline settings.

Contextual-POI-bandit overcomes these shortcomings. The user's SGTC contextual feature matrix computed in the KG module is inputted into the contextual bandit (CB) component. CB module calculates an exploration factor, β , for each user and studies the exploratory behaviour of the user before making final recommendations, thereby striking the right balance between exploration and exploitation. Exploration means recommending new or less-visited POIs which might be liked by the user, while exploitation means suggesting POIs that are known to have been visited by the user in the past. Furthermore, the CB module makes the next POI recommendation in an online setting. In an online setting, unlike offline models, the user has the option to provide feedback to the CB module about its liking or disliking of the POIs recommended. Thus, user activity, such as POI visits, is generated in real-time, thereby allowing the model to continuously update the understanding of user preferences based on recent activity data. For example, if a user starts exploring a new neighbourhood area or develops an interest in a new activity, the CB module quickly adapts to these new preferences and suggests relevant POIs. This allows the Contextual-POI-Bandit to adapt to changes swiftly by continuously integrating

new data and reflecting the user’s long-term and short-term interests and behaviours in real time.

The rest of the chapter is organised as follows. In the next section, the next POI recommendation as a Sequential Decision-Making (SDM) problem under uncertainty is mathematically formulated. Then, the popular state-of-the-art CMAB models are discussed and proposed Neuro-Fuzzy Bandit model is explained. After that, the Contextual-POI-bandit framework is explained in detail and how the CMAB models can seamlessly fit into the framework, offering personalised and efficient recommendations tailored to the needs of the user. Lastly, the empirical evaluation of the proposed framework is performed with the studied CMAB models on public benchmark datasets, namely Yelp, New York and Gowalla datasets, to validate and test the performance of models addressing the dynamic and context-sensitive nature of user preferences. The significance of each of the performance evaluation metrics like accuracy, precision, F1 score, MCC, etc. and their impact on deciding the best CMAB model to be integrated into the proposed framework to meet the growing demands for personalised user experiences in the next POI recommendation is also studied.

5.2 Development of Contextual-POI-Bandit Framework

In this section, Contextual-POI-Bandit framework is implemented for recommending the next Point-of-Interest the user might be willing to visit based on user’s social, categorical, geographical and temporal contexts. For recommending next POI, Contextual-POI-Bandit framework has to make a series of interdependent decisions over time in a stochastic or an uncertain environments. Fuzzy Neural Network (FNN), a CMAB algorithm, is implemented and integrated into the Contextual-POI-Bandit framework for personalised POI recommendation to the user.

5.2.1 Mathematical Modelling

The mathematical model is formulated for the next POI recommendation as a sequential decision-making (SDM) problem under uncertainty using a contextual multi-armed bandit. The notation used is as follows:

- $\mathcal{A} = \{1, 2, \dots, K\}$: Set of K POIs (arms).
- \mathcal{C} : Context space.

- $\mathbf{x}_t \in \mathcal{C}$: Context vector at time t .
- $A_t \in \mathcal{A}$: POI recommended at time t .
- $r_t(A_t)$: Reward received after recommending A_t at time t .
- θ : Parameter vector representing the underlying model to predict rewards.

The problem setup for each time step t ,

1. **Observe context**: Observe the context vector \mathbf{x}_t . The context (or state) includes the features of the user and the environment at the time of making the recommendation. This could include user preferences, location, time of day, weather, etc.
2. **Select action /arm**: Select a POI A_t based on the current policy $\pi(\mathbf{x}_t)$. Each POI is considered an arm in the bandit problem.
3. **Receive reward**: Receive the reward $r_t(A_t)$ after the user interacts with the recommended POI. The reward is the feedback received from the user after recommending a POI. This could be a binary reward (visited or not) or a continuous reward (rating given by the user).

The reward for selecting POI a given context \mathbf{x}_t can be modeled as:

$$r_t(a) = \mathbf{x}_t^\top \theta_a + \varepsilon_t \quad (5.1)$$

where θ_a is the parameter vector for POI a , and ε_t is the noise term (assumed to be i.i.d. with zero mean).

4. **Update policy**: Update the policy π based on the observed reward $r_t(A_t)$.

The policy π maps the context \mathbf{x}_t to an action A_t :

$$\pi : \mathcal{C} \rightarrow \mathcal{A} \quad (5.2)$$

In the case of Thomas Sampling (TS), the action is selected based on a probability distribution over the parameter θ :

$$A_t = \arg \max_a \left\{ \mathbf{x}_t^\top \hat{\theta}_a \right\} \quad (5.3)$$

where $\hat{\theta}_a$ is a sampled estimate from the posterior distribution of θ_a .

5. **Update model:** After observing the reward $r_t(A_t)$, the posterior distribution of θ_a is updated. This can be done using Bayesian updating or other suitable techniques like stochastic gradient descent.

6. **Objective :** The objective is to maximize the cumulative reward over a time horizon T :

$$\max_{\pi} \mathbb{E} \left[\sum_{t=1}^T r_t(A_t) \right] \quad (5.4)$$

where the expectation is taken over the randomness in the context, the action selection, and the rewards.

One of the most common CMAB algorithm is TS. It is a Bayesian approach to the multi-armed bandit problem. It maintains a posterior distribution over the parameters θ and selects actions based on samples from this distribution.

1. **Initialize:** Prior distribution for each θ_a .
2. **For each time step t :**
 - (a) Sample $\hat{\theta}_a$ from the posterior for each a .
 - (b) Select $A_t = \arg \max_a \mathbf{x}_t^\top \hat{\theta}_a$.
 - (c) Observe reward $r_t(A_t)$.
 - (d) Update the posterior distribution for θ_{A_t} .

Thus the next POI recommendation can be modelled as a sequential decision-making problem using a multi-armed contextual bandit. This can effectively incorporate context into the recommendation process and adaptively learn user preferences to improve recommendation quality over time.

5.2.2 Definitions

Definition 1 (POI Contextualization)

POI contextualization refers to the process of adapting and refining the characteristics and recommendations of a POI based on contextual factors such as time, location, and user behavior. The contextualized POI can be represented as $p_c = (p, c)$, where p is the POI and c is the contextual information influencing its recommendation.

Definition 2 (User Activity Pattern)

A user activity pattern A_u represents the typical behavior of a user u in terms of their visits to different POIs over time. It can be described as a sequence of visited POIs and their associated timestamps,

$$A_u = \{(p_1, t_1), (p_2, t_2), \dots, (p_n, t_n)\}, \quad (5.5)$$

where each pair (p_i, t_i) represents a POI visit and the time of the visit.

Definition 3 (Spatial-Temporal Relevance)

Spatial-temporal relevance $R(p, t)$ measures how relevant a POI p is to a user at a specific time t . This relevance is based on factors such as proximity to the user, time-specific events, and historical visit patterns. It is typically represented as a score that quantifies the likelihood of the user being interested in the POI at time t .

Definition 4 (Personalized POI Score)

A personalized POI score $S_{p,u}$ is a metric that evaluates the relevance of a POI p for a user u based on the user's preferences, historical data, and contextual factors. It is calculated using a recommendation algorithm and can be represented as $S_{p,u} \in [0, 1]$, where higher scores indicate greater relevance.

Definition 5 (Recommendation Algorithm)

A recommendation algorithm A is a computational method used to generate a list of POIs recommended to a user u . The algorithm takes into account various factors, such as user preferences, historical visit data, and contextual information, and produces a ranked list of POIs. It can be represented as

$$A(u, C) = \{p_1, p_2, \dots, p_m\}, \quad (5.6)$$

where C denotes the contextual information and $\{p_1, p_2, \dots, p_m\}$ is the list of recommended POIs.

Definition 6 (User Preference Profile)

A user preference profile P_u is a comprehensive representation of a user u 's interests

and preferences regarding POIs. It includes information such as favored POI categories, preferred locations, and historical interaction data. It can be denoted as

$$P_u = (c_1, c_2, \dots, c_k), \quad (5.7)$$

where each c_i represents a specific preference or interest of the user.

Definition 7 (Context-Aware Recommendation System)

A context-aware recommendation system R is a system that incorporates contextual information (e.g., location, time, weather) into its POI recommendation process to provide more relevant suggestions to users. The system can be described as $R(u, C)$, where u is the user, and C represents the contextual factors influencing the recommendations.

Definition 8 (POI Popularity Metric)

The POI popularity metric M_p quantifies the level of interest or popularity of a POI p based on factors such as user ratings, visit frequency, and social media mentions. It can be represented as $M_p \in [0, 1]$, where higher values indicate greater popularity.

Definition 9 (Dynamic POI Recommendation)

Dynamic POI recommendation refers to the process of updating and refining POI recommendations in real-time based on changing contextual information, user behavior, and environmental factors. The dynamic recommendation process can be denoted as $D(u, t, C)$, where t is the current time and C is the updated contextual information.

5.2.3 Proposed Methodology

As discussed earlier, the Contextual-POI-Bandit consists of two modules: KG module and CB module. In the KG module, a unified user context feature vector matrix, derived from the user's Social (S), Geographical (G), Temporal (T), and Categorical (C) correlations with both users and POIs, is generated and used as input in the CB module for the next POI recommendation.

KG Module

Social Information

Social information can be modeled using a power-law distribution based on social links. Let $P_{\text{soc}}(u, p)$ denote the social probability of user u visiting POI p :

$$P_{\text{soc}}(u, p) = 1 - (1 + x(u, p))^{1-\eta} \quad (5.8)$$

where η is estimated from the check-in matrix M and social link matrix A :

$$\eta = 1 + \frac{|U||P|}{\sum_{i \in U} \sum_{j \in P} \ln(1 + \sum_{k \in U} A_{i,k} \cdot M_{k,j})} \quad (5.9)$$

Here, $x(u, p)$ represents the aggregation of the check-in frequency of u 's friends on POI p :

$$x(u, p) = \sum_{v \in U} A_{u,v} M_{v,p} \quad (5.10)$$

Geographical Information

Geographical information can be modeled using kernel density estimation. Let $P_{\text{geo}}(u, p)$ denote the geographical probability of user u visiting POI p :

$$P_{\text{geo}}(u, p) = \frac{1}{N_u} \sum_{k=1}^{N_u} M_{u,p_k} \cdot K_h(p - p_k) \quad (5.11)$$

where N_u is the number of check-ins by user u , M is the user-POI check-in frequency matrix, and $K_h(p - p_k)$ is the kernel function with a bandwidth parameter h .

Temporal Information

The temporal information can be modeled using an Additive Markov Chain [134] to capture the sequential transition pattern between users and POIs. Let $P_{\text{temp}}(u, p)$ denote the temporal probability of user u visiting POI p :

$$P_{\text{temp}}(u, p) = \sum_{j=1}^m g(p_{j+1}, p_j, m+1-j) \quad (5.12)$$

where m is the number of POIs and $g(p_{j+1}, p_j, m+1-j)$ represents the sequential probability of visiting a new location based on the Additive Markov Chain:

$$g(p_{j+1}, p_j, m+1-j) = V(p_j) \cdot TP(p_j \rightarrow p_{j+1}) \Big/ \sum_{k=1}^m V(p_k) \quad (5.13)$$

Here, $V(p_j) = 2^{-\beta \cdot (m-j)}$ is the sequence decay weight with the decay rate parameter $\beta \geq 0$ and $TP(p_j \rightarrow p_{j+1})$ is the transition probability.

Categorical Information

Categorical information can be modeled using a power-law distribution for users' categorical check-in frequency. Let $P_{\text{cat}}(u, p)$ denote the categorical probability of user u visiting POI p :

$$P_{\text{cat}}(u, p) = 1 - (1 + h(u, p))^{1-\delta} \quad (5.14)$$

where $\delta > 1$ and $h(u, p)$ represents the categorical popularity of user u on POI p :

$$h(u, p) = \sum_{c \in \mathcal{C}} F(u, c) \cdot G(c, p) \quad (5.15)$$

Here, \mathcal{C} is the set of all POI categories, $F(u, c)$ is the frequency of user u visiting POIs in category c , and $G(c, p)$ is the check-in frequency of all users on POI p in category c .

The contextual information generated by the Equations [5.8](#), [5.11](#), [5.12](#) and [5.14](#) are combined together with the user features to form the final SGTC user context feature vector matrix which is fed into the FNN as described below.

CB Module

The CB module is designed to optimize POI recommendations by leveraging contextual information. The CB module consists of a multi-arm contextual bandit model with k arms where each arm corresponds to a possible POI to be recommended to the user from an alternate number of POIs possible. Each arm is represented by a FNN model, which is detailed below.

The FNN receives as input a user context feature vector generated by the KG module. This vector encapsulates diverse correlations, including social, geographical, temporal, and categorical relationships between users and POIs. This further helps the CB module to dynamically select the most relevant POIs to recommend, balancing the trade-off between exploration of new POIs and exploitation of known preferences to maximize user satisfaction over time. Also, to capture the exploratory behaviour of the user effectively, a new metric, exploration factor, has been defined. Exploration factor assess a user's tendency to explore new POIs versus revisiting previously visited ones.

Let Q_u represent the total number of unique POIs visited by user u , and C_u denote the total number of check-ins by that user. As shown in Equation [5.16](#), the user exploration

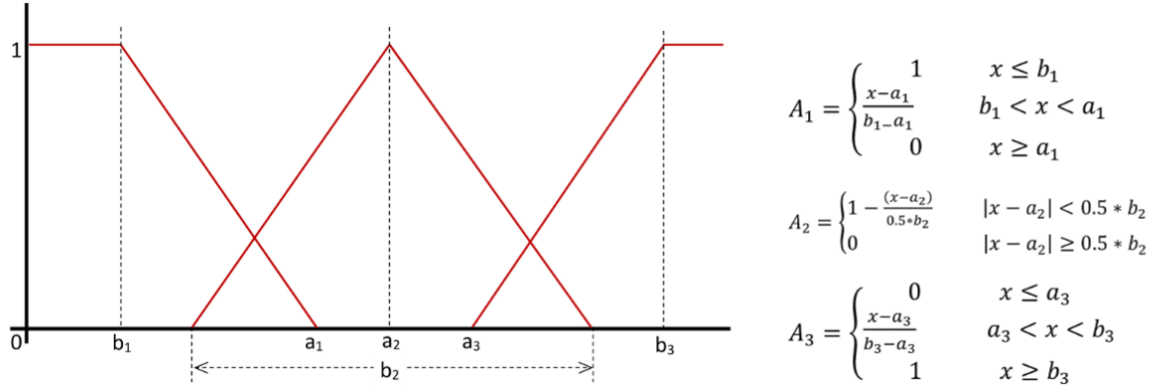


Figure 5.1: Trapezoidal membership function

factor is calculated by dividing the total number of unique POIs visited by the total number of check-ins. An exploration factor of one indicates that all of the user's check-ins are at new POIs, implying the user never revisits a location. Conversely, lower exploration factor values suggest that a user frequently revisits the same POIs. Users are categorized based on this exploration factor, and the performance of the models is analyzed accordingly.

$$\delta = Q_u / C_u \quad (5.16)$$

FNN

FNN model is a hybrid architecture that integrates the strengths of fuzzy system and artificial neural networks. A fuzzy system is a type of mathematical reasoning that handles truth degrees rather than binary numbers. It is a potent tool for simulating human perception and thought. Fuzzy systems are used to hold rules and estimate sampling functions based on language input. They prove to play a vital role in pattern recognition problems as they deal with high uncertainty. The FNN model uses fuzzy system in its first layer to effectively handle the uncertainty by performing non-linear mapping of the fuzzy user feature context vector comprising of social geographical, temporal and categorical contexts to crisp outputs. The FNN models uses Trapezoidal Membership Function (TMF) to map each input value to a degree of membership between 0 and 1. The TMF as shown in Fig. 5.1 is typically defined by four parameters: a , b , c , and d . These parameters determine the shape and position of the trapezoid on the x-axis.

- a : The point where the membership value starts increasing from 0.

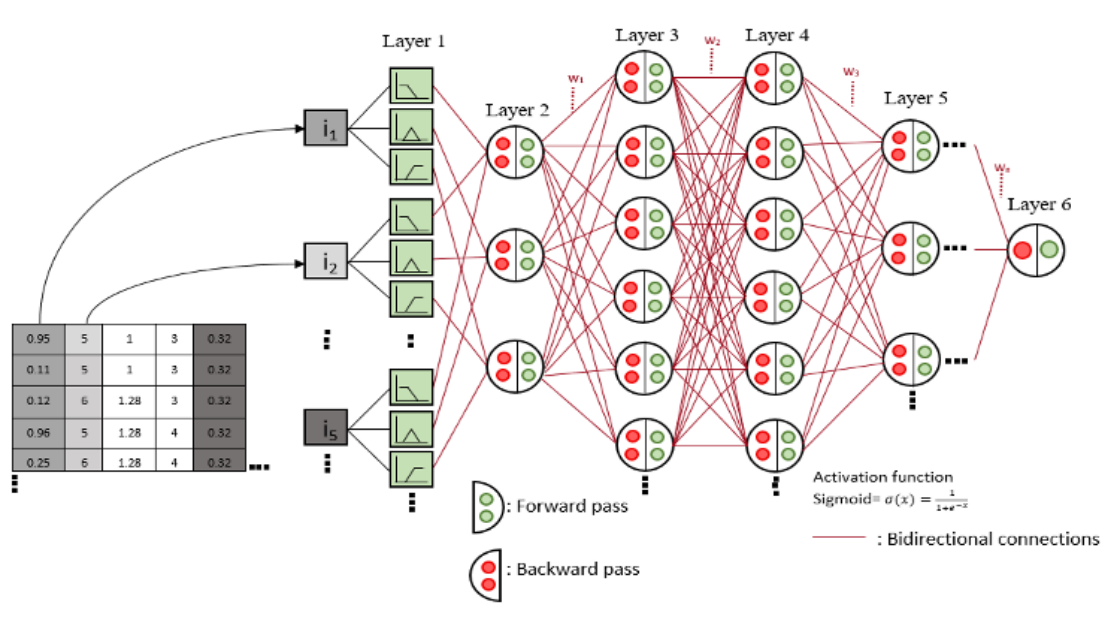


Figure 5.2: FNN model

- b : The point where the membership value reaches 1 (the left "shoulder" of the trapezoid).
- c : The point where the membership value starts decreasing from 1 (the right "shoulder" of the trapezoid).
- d : The point where the membership value returns to 0.

The mathematical representation of TMF can be seen in Fig 5.1 where,

- The **rising edge** from a to b represents the gradual increase in membership from 0 to 1.
- The **top plateau** from b to c represents full membership (value of 1).
- The **falling edge** from c to d represents the gradual decrease in membership from 1 to 0.
- Outside the range $[a, d]$, the membership value is 0.

The FNN model used in the process is shown in the given Figure 5.2. In FNN model, there are six layers and an input layer. First, there is an input layer where we retrieve data

in the form of $x(t) = \{i_1(t), i_2(t), i_3(t), i_4(t), \dots, i_n(t)\}$ from the dataset, where n is the total number of samples and t denotes the iteration number.

The fuzzification layer is the first layer of the model after the input layer. The membership function used in the fuzzification layer is the trapezoidal membership function. Graphical and mathematical representations of this membership function is shown in Fig. 5.2, where $a_1, b_1, a_2, b_2, a_3,$ and b_3 are randomly generated parameters, and x is the input provided to the fuzzification layer.

Layer 1 of Fig. 5.2 illustrates how each input is divided into three components following the fuzzification layer: $A_1, A_2,$ and A_3 . The production layer, the second layer in the model, takes the output of the first layer as input and outputs as x_1 , combining the fuzzy components from the membership function's output. $x_1^1 = A_1^1.A_1^2...A_1^n$, where n is number of inputs in one iteration, subscript shows the number of the layer and superscript shows the number of node in the present layer, as the membership function have three components, production layer will also have only three nodes combining the inputs. Similarly, we can calculate x_1^2 and x_1^3 . Subsequently, we will employ a membership function, namely the sigmoid function, on the output of the second layer and all subsequent layers. The sigmoid function is a membership function that helps to produce output in the range from 0 to 1, which makes it easy to handle outputs and then further process them. The sigmoid membership function is shown in Equation 5.17

$$x_1^o = \frac{1}{1 + e^{-x_1}} \quad (5.17)$$

Where x_1 denotes the output of the 1st layer which is feeded to sigmoid function and x_1^o is the layer's final output, which will be sent as input to the following layer. Further this output of the second layer is passed to the third layer which is a neural layer in FNN. w_1 is the weights for this neural layer and it will first be produced at random and will give output as x_2 , in generic form it will be given as:

$$x_2 = f(x_1^o, w_1) \quad (5.18)$$

Then, the output is passed through the sigmoid membership function in the second layer, and it can be presented as a similar notation, which is shown in Equation 5.19

$$x_2^o = \frac{1}{1 + e^{-x_2}} \quad (5.19)$$

Comparably, the function for the fourth layer is displayed in Equation 5.20. It accepts input as x_2^o . and handles weights as w_2 to produce an output of x_3 . This output is then transferred to the sigmoid membership function, and x_3^o is the layer's final output.

$$x_3 = f(x_2^o, w_2) \quad (5.20)$$

Equation 5.21 displays the function for the fifth layer. It handles weights as w_3 and accepts input as x_3^o to yield an output of x_4 . After that, this output is sent into the sigmoid membership function, and the layer's ultimate output is x_4^o .

$$x_4 = f(x_3^o, w_3) \quad (5.21)$$

Equation 5.22 illustrates the fifth layer's function. It takes in input as x_4^o and handles weights as w_4 to produce an output of x_5 . The final output of the layer is x_5^o . This output is then fed into the sigmoid membership function.

$$x_5 = f(x_4^o, w_4) \quad (5.22)$$

The final output of the FNN will be denoted by x_5^o as shown in Equation 5.23)

$$x_5^o = \frac{1}{1 + e^{-x_5}} \quad (5.23)$$

Since all of the initial parameters were created at random, it is only logical that there would be error in our output after this feed-forward process. This process of hyper-tuning the parameters to minimise the error is described in the next section.

Tuning of hyper-parameters: Let's denote the outcome of FNN with z , which is equal to x_5^o in Equation 5.24 and the output expected after the prediction is t . The Mean Square Error (MSE) for FNN is shown in Equation 5.24.

$$E = \frac{1}{2n} \sum_{i=1}^n (z - t)^2 \quad (5.24)$$

where 'n' is the total number of samples of input. The process of backtracking is depicted in a generalized form, and for example, the value of n is taken as 1 for the simplification of the derivations. The error function in Equation 5.24 will give the total error of the model, and it will be minimized using the process of gradient descent. Gradient Descent is an algorithm where, with each iteration, we try to update the parameters, which are weights in this case, to minimize the error.

Layer Six: According to the algorithm of gradient decent, in order to minimize the error value in output, the partial differentiation of error function is required w.r.t. parameters, i.e., weights of the respective layer. The deduction of the partial differentiation for the sixth layer is shown in Equation 5.25

$$\frac{\partial E}{\partial w_4} = \frac{\partial E}{\partial x_5^o} \cdot \frac{\partial x_5^o}{\partial x_5} \cdot \frac{\partial x_5}{\partial w_4} \quad (5.25)$$

where $\frac{\partial E}{\partial x_5^o}$ is the partial differentiation of Error function (E) w.r.t. x_5^o , which is the output of the last layer which is shown in Equation 5.26.

$$\frac{\partial E}{\partial x_5^o} = x_5^o - t \quad (5.26)$$

$\frac{\partial x_5^o}{\partial x_5}$ is the partial differentiation of the output of a layer after sigmoid membership function w.r.t. output of the layer without the membership function, it is shown in Equation 5.27 and also given a generalized notation i.e., σ_5 for it as this step will be repeated in every layer so it becomes easy in calculation to denote it with a generalized term.

$$\frac{\partial x_5^o}{\partial x_5} = \frac{e^{-x_5}}{(1 + e^{-x_5})^2} = \sigma_5 \quad (5.27)$$

$\frac{\partial x_5}{\partial w_4}$ is the partial differentiation of x_5 which is the output without membership function w.r.t. w_4 which are the weights between the present layer and previous layer which are needed to get update in order to achieve less error value and it shown in Equation 5.28.

$$\frac{\partial x_5}{\partial w_4} = f'(x_4^o, w_4) \cdot x_4 \quad (5.28)$$

In Equation 5.28, $f'(x_n^o, w_n)$ is the denotation for differentiated function which has final outputs of the previous layers as inputs and respective weights. Now combine the par-

tial differentiations from Equations [5.26](#), [5.27](#), [5.28](#), and put together in Equation [\(5.25\)](#).

$$\frac{\partial E}{\partial w_4} = (x_5^o - t) \cdot \sigma_5 \cdot f'(x_4^o, w_4) \cdot x_4^o \quad (5.29)$$

To simplify the terms, some general values can be clubbed together which will be repeating further in derivation. δ_5 is assumed notation for the ease of calculations as shown in Equation [5.30](#).

$$\Delta_5 = (x_5^o - t) \cdot \sigma_5 \cdot f'(x_4^o, w_4) \quad (5.30)$$

Now, putting the simplified assumed notation from Equation [5.30](#) to the expanded Equation [5.29](#).

$$\frac{\partial E}{\partial w_4} = \Delta_5 \cdot x_4^o \quad (5.31)$$

As after getting the partial differentiation of error function w.r.t. respective weight, the weight for the respective layer can be updated using Equation [5.32](#).

$$\text{updated_w_4} = w_4 - \alpha \frac{\partial E}{\partial w_4} \quad (5.32)$$

where α is the learning rate of the model, learning rate is the tuning parameter in the process of optimization which regulates the model's parameter changes in steps that lead to the lowest possible error value.

Layer Five: To update the weights of this layer i.e., w_3 , partial differentiation of E w.r.t. w_3 is needed, which is deduced in Equation [5.33](#).

$$\frac{\partial E}{\partial w_3} = \frac{\partial E}{\partial x_5^o} \cdot \frac{\partial x_5^o}{\partial x_5} \cdot \frac{\partial x_5}{\partial x_4^o} \cdot \frac{\partial x_4^o}{\partial x_4} \cdot \frac{\partial x_4}{\partial w_3} \quad (5.33)$$

Here values for $\frac{\partial E}{\partial x_5^o}$ and $\frac{\partial x_5^o}{\partial x_5}$ are available in previous equations i.e. Equations [5.26](#) and [5.27](#). $\frac{\partial x_5}{\partial x_4^o}$ is the partial differentiation of output of sixth layer w.r.t. to the final output of fifth layer in FNN, which is shown in Equation [5.34](#).

$$\frac{\partial x_5}{\partial x_4^o} = f'(x_4^o, w_4) \cdot w_4 \quad (5.34)$$

$\frac{\partial x_4^o}{\partial x_4}$ is the partial differentiation of the final output of the fifth layer after the membership function w.r.t. the output of the layer without membership function. It is very similar

to Equation 5.27 as it is the differentiation of sigmoid function only, so the generic notation continues for it which is shown in Equation 5.35.

$$\frac{\partial x_4^o}{\partial x_4} = \sigma_4 = \frac{e^{-x_4}}{(1 + e^{-x_4})^2} \quad (5.35)$$

$\frac{\partial x_4}{\partial w_3}$ is the partial differentiation of the output of the respective layer w.r.t. weights between the respective layer and the previous layer, it is shown in Equation 5.36.

$$\frac{\partial x_4}{\partial w_3} = f'(x_3^o, w_3) \cdot x_3^o \quad (5.36)$$

Putting deduced Equations 5.34, 5.35, 5.36 and previously calculated Equations 5.26, and 5.27 in Equations 5.33 for an expanded form which is shown in Equation 5.37.

$$\frac{\partial E}{\partial w_3} = (x_3^o - t) \cdot \sigma_5 \cdot f'(x_4^o, w_4) \cdot w_4 \cdot \sigma_4 \cdot f'(x_3^o, w_3) \cdot x_3^o \quad (5.37)$$

As there are some repetitive terms which can be replaced by generic assumed notations. So, using Equation 5.30 in Equation 5.37.

$$\frac{\partial E}{\partial w_3} = \Delta_5 \cdot w_4 \cdot \sigma_4 \cdot f'(x_3^o, w_3) \cdot x_3^o \quad (5.38)$$

To simplify the terms, some general values can be clubbed together which will be repeating further in derivation. δ_4 is assumed notation for the ease of calculations as shown in Equation 5.39.

$$\Delta_4 = \Delta_5 \cdot w_4 \cdot \sigma_4 \cdot f'(x_3^o, w_3) \quad (5.39)$$

After all the deductions and using all generalized notations the final equation for the partial differentiation of E w.r.t. w_3 is shown in Equation 5.40.

$$\frac{\partial E}{\partial w_3} = \Delta_4 \cdot x_3^o \quad (5.40)$$

And according to gradient decent algorithm the weights for this layer will be updated, which is shown in Equation 5.41.

$$\text{updated_w_3} = w_3 - \alpha \frac{\partial E}{\partial w_3} \quad (5.41)$$

Layer Four:

To update the respective parameters if this layer which is w_2 , partial differentiation of E

w.r.t. w_2 is needed and it is deduced in Equation [5.42](#).

$$\frac{\partial E}{\partial w_2} = \frac{\partial E}{\partial x_5^o} \cdot \frac{\partial x_5^o}{\partial x_5} \cdot \frac{\partial x_5}{\partial x_4^o} \cdot \frac{\partial x_4^o}{\partial x_4} \cdot \frac{\partial x_4}{\partial x_3^o} \cdot \frac{\partial x_3^o}{\partial x_3} \cdot \frac{\partial x_3}{\partial w_2} \quad (5.42)$$

Here we can directly take values from Equations [5.26](#), [5.27](#), [5.34](#), and [5.35](#). And rest of the values are creating a pattern and repeating themselves. So, the expanded form of Equation [5.42](#) is shown in Equation [5.43](#).

$$\frac{\partial E}{\partial w_2} = (x_5^o - t) \cdot \sigma_5 \cdot f'(x_4^o, w_4) \cdot w_4 \cdot \sigma_4 \cdot f'(x_3^o, w_3) \cdot w_3 \cdot \sigma_3 \cdot f'(x_2^o, w_2) \cdot x_2^o \quad (5.43)$$

Using Equation [5.30](#) in Equation [5.43](#) for simplification of the equation.

$$\frac{\partial E}{\partial w_2} = \Delta_5 \cdot w_4 \cdot \sigma_4 \cdot f'(x_3^o, w_3) \cdot w_3 \cdot \sigma_3 \cdot f'(x_2^o, w_2) \cdot x_2^o \quad (5.44)$$

Using Equation [5.39](#) in Equation [5.44](#).

$$\frac{\partial E}{\partial w_2} = \Delta_4 \cdot w_3 \cdot \sigma_3 \cdot f'(x_2^o, w_2) \cdot x_2^o \quad (5.45)$$

From above calculations Δ_3 can be computed in a similar pattern and is shown in Equation [5.46](#).

$$\Delta_3 = \Delta_4 \cdot w_3 \cdot \sigma_3 \cdot f'(x_2^o, w_2) \quad (5.46)$$

Using Equation [5.46](#) in Equation [5.45](#) to simplify and find the final equation of the partial differentiation of E with respect to w_2 .

$$\frac{\partial E}{\partial w_2} = \Delta_3 \cdot x_2^o \quad (5.47)$$

After getting the final equation in Equation [5.47](#), updated weight can be calculated as shown in Equation [5.48](#).

$$\text{updated_w_2} = w_2 - \alpha \frac{\partial E}{\partial w_2} \quad (5.48)$$

Layer Three: From previous findings and generic notations, the partial differentiation of E with respect to w_1 is deduced in Equation [5.50](#) from Equation [5.49](#) easily.

$$\frac{\partial E}{\partial w_1} = \frac{\partial E}{\partial x_3} \cdot \frac{\partial x_3}{\partial x_2^o} \cdot \frac{\partial x_2^o}{\partial x_2} \cdot \frac{\partial x_2}{\partial w_1} \quad (5.49)$$

$$\frac{\partial E}{\partial w_1} = \Delta_3 \cdot w_2 \cdot \sigma_2 \cdot f'(x_1^o, w_1) \cdot x_1^o \quad (5.50)$$

From Equation 5.50 and previous patterns, Δ_2 can be computed easily and is shown in Equation 5.51

$$\Delta_2 = \Delta_3 \cdot w_2 \cdot \sigma_2 \cdot f'(x_1^o, w_1) \quad (5.51)$$

Putting Equation 5.51 in Equation 5.50:

$$\frac{\partial E}{\partial w_1} = \Delta_2 \cdot x_1^o \quad (5.52)$$

After getting the final equation for the partial differentiation, weights can be updated easily as shown in Equation 5.53:

$$\text{updated_w_1} = w_1 - \alpha \frac{\partial E}{\partial w_1} \quad (5.53)$$

Layer Two: This layer is a production layer that combines all the components of the fuzzification layer. This particular layer does not have any variable weights, hence the update in the parameters is not possible for this layer.

Layer One: In the first layer during the forward pass, the trapezoidal function is used for the fuzzification, which is shown in Fig 2. Firstly, the partial differentiation of A_1 , A_2 , and A_3 will be taken on the basis of the respective variables a_1 , b_1 , a_2 , b_2 , a_3 , and b_3 , which is shown in equations from Equation 5.54 to Equation 5.59.

$$\frac{\partial A_1}{\partial a_1} = \frac{x - b_1}{(b_1 - a_1)^2} \quad (5.54)$$

where a_1 and b_1 are the hyperparameters for the first component of fuzzy logic, i.e., A_1 . Equation 5.54 shows the partial differentiation of component A_1 with respect to variable a_1 .

$$\frac{\partial A_1}{\partial b_1} = \frac{a_1 - x}{(b_1 - a_1)^2} \quad (5.55)$$

where the hyperparameters for the first fuzzy logic component A_1 are a_1 and b_1 . Additionally, Equation 5.55 illustrates the partial differentiation of component A_1 with respect to variable b_1 .

$$\frac{\partial A_2}{\partial a_2} = \frac{1}{0.5 \cdot b_2} \quad (5.56)$$

where a_2 and b_2 are the hyperparameters for the second fuzzy logic component A_2 . Furthermore, Equation 5.56 shows the partial differentiation of component A_2 with respect to variable a_2 .

$$\frac{\partial A_2}{\partial b_2} = \frac{x - a_2}{0.5 \cdot b_2^2} \quad (5.57)$$

where the hyperparameters for the second fuzzy logic component A_2 are a_2 and b_2 . Moreover, Equation 5.57 illustrates the partial differentiation of component A_2 with respect to variable b_2 .

$$\frac{\partial A_3}{\partial a_3} = \frac{x - b_3}{(b_3 - a_3)^2} \quad (5.58)$$

where a_3 and b_3 are the hyperparameters for the third fuzzy logic component A_3 . Additionally, Equation 5.58 shows how component A_3 is partially differentiated with respect to variable a_3 .

$$\frac{\partial A_3}{\partial b_3} = \frac{a_3 - x}{(b_3 - a_3)^2} \quad (5.59)$$

where the hyperparameters for the third fuzzy logic component A_3 are a_3 and b_3 . Furthermore, Equation 5.59 illustrates the partial differentiation of component A_3 with respect to variable b_3 .

Further to update all six variables, we need to differentiate E with respect to the respective variables, which is shown in equations from Equation 5.60 to Equation 5.68.

$$\frac{\partial E}{\partial a_1} = \frac{\partial E}{\partial x_2} \cdot \frac{\partial x_2}{\partial x_1^o} \cdot \frac{\partial x_1^o}{\partial x_1} \cdot \frac{\partial x_1}{\partial A_1} \cdot \frac{\partial A_1}{\partial a_1} \quad (5.60)$$

Equation 5.60 illustrates the partial differentiation of the error function with respect to the fuzzy parameter a_1 .

$$\frac{\partial E}{\partial a_1} = \Delta_2 \cdot w_1 \cdot \sigma_1 \cdot 1 \cdot \left(\frac{x - b_1}{(b_1 - a_1)^2} \right) \quad (5.61)$$

In Equation 5.61, the parameter Δ_2 is from Equation 5.51, w_1 is the parameter related to layers 2 and 3 of the FNN as seen in Fig. 5.2, σ_1 is the partial differentiation of the sigmoid function utilized in layer 3, a_1 and b_1 are the parameters related to fuzzy logic, and x is the input provided to the FNN model.

Similarly, remaining values will also be calculated.

$$\frac{\partial E}{\partial b_1} = \Delta_2 \cdot w_1 \cdot \sigma_1 \cdot 1 \cdot \left(\frac{a_1 - x}{(b_1 - a_1)^2} \right) \quad (5.62)$$

The partial differentiation of the error function with regard to the fuzzy parameter b_1 is shown in Equation 5.62.

$$\frac{\partial E}{\partial a_2} = \Delta_2 \cdot w_1 \cdot \sigma_1 \cdot 1 \cdot \frac{1}{0.5 \cdot b_2} \quad (5.63)$$

Equation 5.63 displays the partial differentiation of the error function with respect to the fuzzy parameter a_2 .

$$\frac{\partial E}{\partial b_2} = \Delta_2 \cdot w_1 \cdot \sigma_1 \cdot 1 \cdot \left(\frac{x - a_2}{0.5 \cdot b_2^2} \right) \quad (5.64)$$

The partial differentiation of the error function with regard to the fuzzy parameter b_2 is shown in Equation 5.64.

$$\frac{\partial E}{\partial a_3} = \Delta_2 \cdot w_1 \cdot \sigma_1 \cdot 1 \cdot \left(\frac{x - b_3}{(b_3 - a_3)^2} \right) \quad (5.65)$$

Equation 5.65 displays the partial differentiation of the error function with respect to the fuzzy parameter a_3 .

$$\frac{\partial E}{\partial b_3} = \Delta_2 \cdot w_1 \cdot \sigma_1 \cdot 1 \cdot \left(\frac{a_3 - x}{(b_3 - a_3)^2} \right) \quad (5.66)$$

The partial differentiation of the error function with regard to the fuzzy parameter b_3 is shown in Equation 5.66. These values can be updated easily with the help of general terms given in Equations 5.67 and 5.68.

$$\text{updated_}a_i = a_i - \alpha \frac{\partial E}{\partial a_i} \quad (5.67)$$

$$\text{updated_}b_i = b_i - \alpha \frac{\partial E}{\partial b_i} \quad (5.68)$$

From this complete hyper tuning process, some generalized equations can be deduced in the form of 'i-th' term and it is shown in equations from Equation 5.69 to Equation 5.73.

$$\frac{\partial E}{\partial w_i} = \Delta_{i+1} \cdot x_i^o \quad (5.69)$$

where $\frac{\partial E}{\partial w_i}$ is the partial differentiation of E with respect to weights associated with the respective layer, and x_i^o is the output of the previous layer associated with the weights for the i -th layer.

$$\Delta_i = \Delta_{i+1} \cdot w_i \cdot \sigma_i \cdot f'(x_{i-1}^o, w_{i-1}) \quad (5.70)$$

$$\sigma_i = \frac{e^{-x_i}}{(1 + e^{-x_i})^2} \quad (5.71)$$

where σ_i is the differentiation of the sigmoid function for the i -th layer.

$$w_i = w_i - \alpha \cdot \frac{\partial E}{\partial w_i} \quad (5.72)$$

Equation [5.72](#) is the generic formula to update the weight for the i -th layer.

$$\Delta_L = (x_L^o - t) \cdot \sigma_i \cdot f'(x_{i-1}^o, w_{i-1}) \quad (5.73)$$

Equation [5.73](#) is similar to Equation [5.70](#), where L in Equation [5.73](#) refers to the variables of the last layers.

5.3 Datasets and Evaluation Metrics

This section provides a detailed overview of the datasets utilized and the empirical metrics employed in the evaluation of the Contextual-POI-Bandit framework. Specifically, the characteristics and relevance of the selected datasets, including their origins, structure, and the nature of the data, are discussed. Additionally, the empirical metrics are outlined and used to assess the performance and effectiveness of the proposed framework. Through a comprehensive analysis of these datasets and metrics, a thorough evaluation of the Contextual-POI-Bandit framework's capabilities in real-world scenarios is conducted.

5.3.1 Datasets

For this study, four real-world check-in datasets, sourced from Yelp, Gowalla, New York and Tokyo are used. The Yelp dataset utilized for next-POI recommendation comprises a comprehensive collection of user reviews, check-ins, and business information from the Yelp platform. This dataset includes detailed user profiles, which provide insights into individual preferences and behaviors, as well as a wide array of businesses, categorized

by type, location, and other attributes. Each entry contains temporal data on user check-ins, allowing for the analysis of user activity patterns over time. Additionally, the dataset features textual reviews that offer qualitative assessments of various POIs, enriching the contextual understanding of user preferences. This rich dataset enables the development and evaluation of recommendation algorithms aimed at predicting the next POI a user is likely to visit based on historical interactions and contextual factors.

The Gowalla dataset, utilized for next-POI recommendation, comprises location-based check-in data from the Gowalla social networking application. This dataset includes detailed records of users' visits to various Points of Interest (POIs) along with temporal and spatial metadata. Each entry in the dataset provides information about the user, the specific POI visited, and the timestamp of the check-in. The dataset encompasses a wide range of POIs across multiple categories, offering a rich source of information for analyzing user behavior and preferences. It serves as a valuable resource for developing and evaluating recommendation algorithms aimed at predicting users' next likely POI based on their historical check-in patterns and contextual factors.

The New York and Tokyo datasets are valuable resources for analyzing user behavior in Location-Based Social Networks (LBSNs) [140]. These datasets, collected from Foursquare, provide a detailed record of user check-ins over a period of approximately 10 months, from April 12, 2012, to February 16, 2013. The New York dataset comprises 227,428 check-ins, while the Tokyo dataset contains a more extensive collection of 573,703 check-ins. Each entry in these datasets includes critical information such as the user's anonymized ID, the venue ID, the venue category ID and name, along with the geographic coordinates (latitude and longitude) of the check-in location. Additionally, the datasets provide the UTC time of the check-in and the timezone offset in minutes, which indicates the difference between the local check-in time and Coordinated Universal Time (UTC). The dataset was originally compiled to analyze the spatial-temporal regularity of user activity within LBSNs. The data is structured in a Tab-Separated Values (TSV) format, encompassing eight essential columns: anonymized user ID, Foursquare venue ID, venue category ID, venue category name, latitude, longitude, timezone offset in minutes, and the time in UTC. This rich dataset serves as a valuable resource for studying urban mobility patterns, user behavior, and the dynamics of social interactions within New York and Tokyo city.

Table 5.1: Description of datasets for next POI recommendation. $|U|$: number of users, $|S|$: number of social links, $|C|$: number of categories, $|\text{checkins}|$: number of check-ins, $|\text{POIs}|$: number of POIs.

Datasets	$ U $	$ \text{POIs} $	$ \text{checkins} $	$ C $	$ S $	$\frac{ \text{checkins} }{ U }$	$\frac{ \text{checkins} }{ \text{POIs} }$	% Sparsity
Yelp	7,135	16,621	301,753	595	46,778	159.42	68.43	99.94%
Gowalla	5,628	31,803	620,683	-	46,001	110.28	19.51	99.78%

5.3.2 Evaluation Metrics

To assess the efficacy of the proposed methods, three well-established evaluation metrics for location-based recommendations were utilized: Precision at K ($P@K$), Recall at K ($R@K$), and Normalized Discounted Cumulative Gain at K ($n\text{DCG}@K$). Here, K takes values from the set $\{10, 20\}$. These metrics were applied to the top- K POIs suggested for a user u .

Precision at K ($P@K$) is calculated using the formula:

$$P@K = \frac{TP_u}{TP_u + FP_u}$$

where TP_u represents the number of relevant POIs correctly recommended to the user, and FP_u denotes the number of irrelevant POIs that were incorrectly recommended.

Recall at K ($R@K$) is determined as:

$$R@K = \frac{TP_u}{TP_u + FN_u}$$

Here, TP_u again represents the relevant POIs correctly recommended, while FN_u signifies the number of relevant POIs that were not included in the top- K recommendations.

These metrics provide a comprehensive view of how well the recommendations align with the user’s actual preferences by balancing the trade-off between precision and recall in the context of the top- K POIs returned.

For example, in order to calculate precision and recall based on the below dataset (as depicted in Table 5.2) for a user u and $K=2$, following steps are to be followed.

Step-by-Step Calculation for a user, u

1. True Positives (TP_u): POIs that were recommended and were actually visited by the user.

Table 5.2: Data for recommended POIs and actual POIs visited by user

Instance	Recommended POI	Actual POI visited by user
t=0	{p1, p2}	p3
t=1	{p2, p3}	p4
t=2	{p2, p4}	p5
t=3	{p3, p5}	p3
t=4	{p4, p5}	p5
Total	10	5

From the table:

- $t = 3$: {p3, p5}, and the user visited $p3$ (so $TP_u = p3$).

- $t = 4$: {p4, p5}, and the user visited $p5$ (so $TP_u = p5$).

So, total $TP_u = 2$ ($p3$ and $p5$)

2. False Positives (FP_u): POIs that were recommended but were not visited by the user.

From the table:

- $t = 0$: {p1, p2}, but the user visited $p3$ (so $FP_u = p1, p2$).

- $t = 1$: {p2, p3}, but the user visited $p4$ (so $FP_u = p2, p3$).

- $t = 2$: {p2, p4}, but the user visited $p5$ (so $FP_u = p2, p4$).

- $t = 3$: {p3, p5}, the user visited $p3$ but not $p5$ (so $FP_u = p5$).

- $t = 4$: {p4, p5}, the user visited $p5$ but not $p4$ (so $FP_u = p4$).

So, total $FP_u = 8$ ($p1, p2, p2, p3, p2, p4, p5, p4$).

3. False Negatives (FN_u): POIs that were not recommended but were actually visited by the user.

From the table:

- $t = 0$: User visited $p3$, but it was not recommended ($FN_u = p3$).

- $t = 1$: User visited $p4$, but it was not recommended ($FN_u = p4$).

- $t = 2$: User visited $p5$, but it was not recommended ($FN_u = p5$).

So, total $FN_u = 3$ ($p3, p4, p5$).

4. True Negatives (TN_u): POIs that were neither recommended nor visited by the user.

From the table: True negatives would include all POIs that weren't in the recommendations or the actual visits. These would require knowledge of the full set of POIs in the system as shown in Table [5.3](#).

$$\text{Precision at K=2 (P@2)} = \frac{TP_u}{TP_u + FP_u} = \frac{2}{2+8} = \frac{2}{10} = 0.2$$

So, $\mathbf{P@2} = 0.2$ or **20%**. This means that 20% of the recommended POIs were actually relevant.

$$\text{Recall at K=2 (R@2)} = \frac{TP_u}{TP_u + FN_u} = \frac{2}{2+3} = \frac{2}{5} = 0.4$$

So, $\mathbf{R@2} = 0.4$ or **40%**. This means that 40% of the relevant POIs were successfully recommended.

Table 5.3: Confusion matrix

Instance	TP	FP	FN	TN
t=0	0	{p1, p2}	{p3}	{p4, p5}
t=1	0	{p2, p3}	{p4}	{p1, p5}
t=2	0	{p2, p4}	{p5}	{p1, p3}
t=3	{p3}	{p5}	0	{p1, p2, p4}
t=4	{p5}	{p4}	0	{p1, p2, p3}
Total	2	8	3	12

The summarized results for the above case is tabulated in Table 5.3 as confusion matrix. This shows that while the system was able to identify some relevant POIs, many of its recommendations were not actually useful to the user.

5.4 Results and Discussion

The performance of the Contextual-POI-Bandit model is evaluated on three real-world datasets: New York, Gowalla, and Tokyo, as shown in Table 5.4. The evaluation metrics used to assess the model’s effectiveness include Precision at 10 (P@10), Precision at 20 (P@20), Recall at 10 (R@10), and Recall at 20 (R@20). These metrics provide insight into how well the model recommends relevant POIs to users within the top 10 and top 20 suggestions.

It can be observed from the Table 5.4,

1. Across all datasets, precision tends to decrease as the number of recommendations increases from 10 to 20. This trend aligns with the typical trade-off between precision and recall, where increasing the number of recommendations enhances the likelihood

Table 5.4: Empirical evaluation of Contextual-POI-Bandit for New York, Tokyo, and Gowalla datasets

Dataset	P@10	P@20	R@10	R@20
New York	0.0121	0.1736	0.0164	0.3012
Tokyo	0.0294	0.0312	0.3013	0.6991
Gowalla	0.0096	0.00473	0.098	0.102

of covering more relevant POIs but also introduces more irrelevant POIs, thereby reducing precision.

2. Recall consistently increases as the number of recommendations grows, which is expected since a broader range of POIs is considered, improving the chance of identifying all relevant POIs.
3. For the New York dataset, the Contextual-POI-Bandit achieved a Recall of 0.0164 at 10 recommendations, which increases substantially to 0.3012 when considering the top 20 recommendations, indicating a higher coverage of relevant POIs as more suggestions are made. The precision at 10 recommendations is relatively low at 0.0121, indicating that out of the top 10 recommended POIs, only a small percentage were relevant to the users. However, precision improves significantly at 20 recommendations, with a value of 0.1736.
4. For Tokyo dataset, the precision at 10 is 0.0294, higher than the New York dataset, suggesting better relevance in the top 10 recommendations for Tokyo. Also, recall at 10 is much higher at 0.3013 which further increases to 0.6991 at 20 recommendations. Overall, the Contextual-POI-Bandit performed best in all evaluation metrics on the Tokyo dataset compared to the other datasets, New York and Gowalla.
5. For the Gowalla dataset, the precision at 10 recommendations is 0.0096, the lowest among the three datasets analyzed. This precision further declines to 0.00473 when the number of recommendations is increased to 20, suggesting a significant decrease in relevance as the volume of recommendations grows. In contrast, the Contextual-POI-Bandit model achieves recall values of 0.098 at 10 recommendations and 0.102 at 20 recommendations.

The Contextual-POI-Bandit is a novel contextual bandit model that leverages reinforcement learning to dynamically adapt its recommendations by integrating users' social

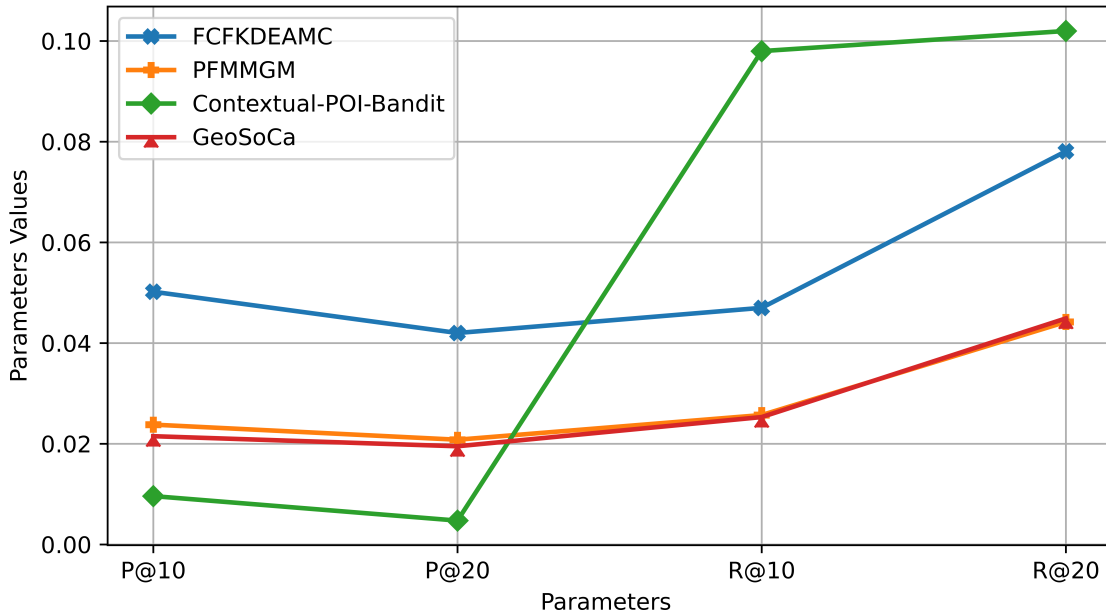


Figure 5.3: Performance analysis of Contextual-POI-Bandit vs other models for Gowalla dataset

preferences, location, category, and time into a unified user context feature vector. Three baseline algorithms, GeoSoCa [134], PFMMGM [141], and FCFKDEAMC [133], are considered for comparison on Gowalla dataset, as these models also incorporate spatio-temporal, geographical, and categorical (STGC) contextual information similar to the Contextual-POI-Bandit for next-POI recommendation. However, it is important to note that all three baseline models are supervised machine learning approaches that provide static recommendations. In general, supervised learning models tend to produce better results compared to reinforcement learning models that train the model on the fly. In Fig. 5.3, a comparative analysis is presented for four different models used in Point of Interest (POI) recommendation systems: FCFKDEAMC, PFMMGM, Contextual-POI-Bandit, and GeoSoCa. This x-axis represents the key performance metrics, namely Precision at 10 (P@10), Precision at 20 (P@20), Recall at 10 (R@10), and Recall at 20 (R@20). For Fig. 5.3, it can be observed:

1. All four models demonstrate a lower P@20 value compared to P@10, and a higher R@20 value compared to R@10. This pattern arises because, as K (number of recommendations for each instance) increases, precision generally decreases while recall increases. Recommending a larger number of POIs improves the chances of identify-

ing locations that users may be interested in visiting. However, this also includes POIs that users are less likely to visit, thereby lowering the precision.

2. The FCFKDEAMC model exhibits the highest precision at both P@10 and P@20, indicating its superior ability to make relevant recommendations within smaller lists.
3. PFMGM, while consistent, falls behind in precision compared to FCFKDEAMC but remains ahead of Contextual-POI-Bandit and GeoSoCa.
4. The Contextual-POI-Bandit model has the lowest precision, suggesting that it is less effective in making highly relevant recommendations, particularly in smaller lists.
5. GeoSoCa shows better precision than Contextual-POI-Bandit but still lags behind FCFKDEAMC and PFMGM.
6. The Contextual-POI-Bandit model outperforms all others in terms of recall, especially as the recommendation list grows, indicating its effectiveness in covering a wide range of relevant POIs.
7. FCFKDEAMC, PFMGM and GeoSoCa models show a steady improvement in recall, although they do not reach the levels achieved by the Contextual-POI-Bandit model.

Next, the performance of all variants of the baseline algorithms, using all possible permutations of the STGC contextual information, is evaluated on two benchmark datasets: the Yelp dataset and the Gowalla dataset. This approach allows for a comprehensive analysis of the contribution of each contextual component. It is important to note that the Gowalla dataset lacks categorical information, restricting our experiments to models based solely on social, temporal, and geographical (STC) context.

Figures 5.4, 5.5, 5.6 provide a comparative analysis of different algorithms across several evaluation metrics using the Yelp dataset. Similarly, Figs. 5.7, 5.8, 5.9 offer a comparative analysis of these algorithms on the Gowalla dataset. Each figure includes a legend that identifies the different variants of a similar algorithm being compared. For instance, in Fig. 5.4, GeoSoCa.G represents the GeoSoCa algorithm with geographical context considered, while GeoSoCa.GS incorporates both geographical and social contexts, and GeoSoCa.GC includes geographical and categorical contexts, among others. The x-axis across all graphs represents various evaluation parameters—P@10, P@20, R@10, R@20, nDCG@10, and nDCG@20. These evaluation metrics are standard in information retrieval and recommendation systems, where precision, recall, and normalized

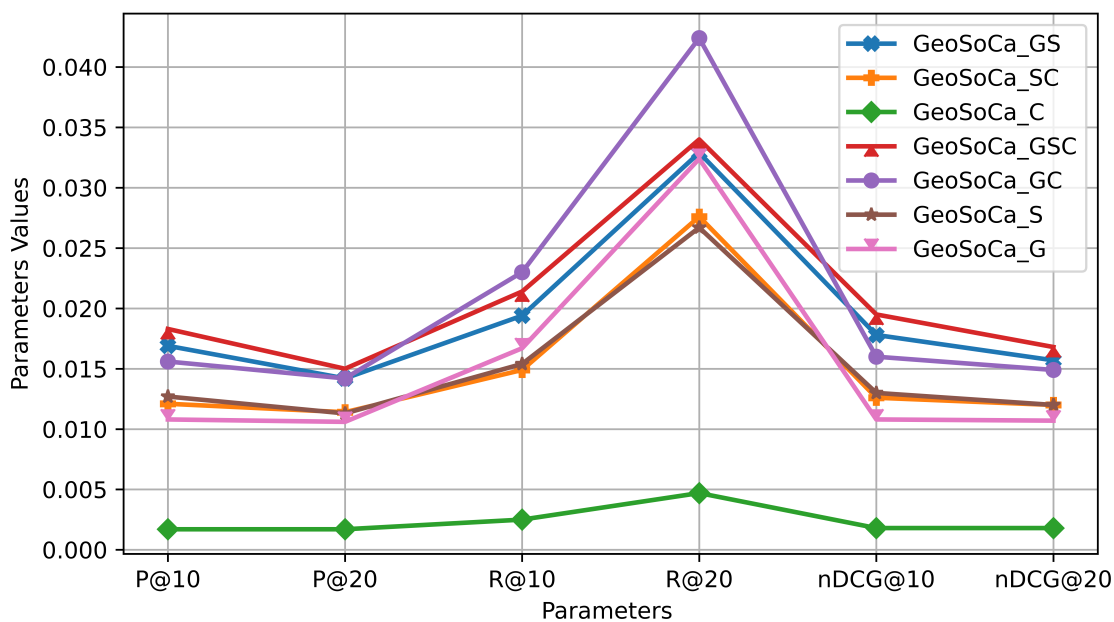


Figure 5.4: GeoSoCa algorithm performance for different permutations of SGTC context on Yelp Dataset

discounted cumulative gain (nDCG) are crucial for assessing the quality of an algorithm’s recommendations. The y-axis indicates the values corresponding to these metrics, enabling a detailed performance evaluation of each algorithm across different metrics.

For Fig. 5.4, the following observations can be made:

- The precision metrics, P@10 and P@20, are relatively close for all models, with minor variations. GeoSoCa_GSC and GeoSoCa_GC show slightly better performance than other models, indicating marginally higher accuracy in the top 10 and top 20 recommendations.
- The recall metrics, R@10 and R@20, show more significant differences. GeoSoCa_GC outperforms the other models significantly at R@10, suggesting a superior ability to capture relevant recommendations within a smaller list. However, this advantage diminishes slightly at R@20, where GeoSoCa_GS performs more competitively.
- Regarding nDCG, which evaluates the ranking of results, GeoSoCa_GC and GeoSoCa_G exhibit the best performance. This indicates that these models not only identify relevant items but also rank them appropriately within the recommendation list.

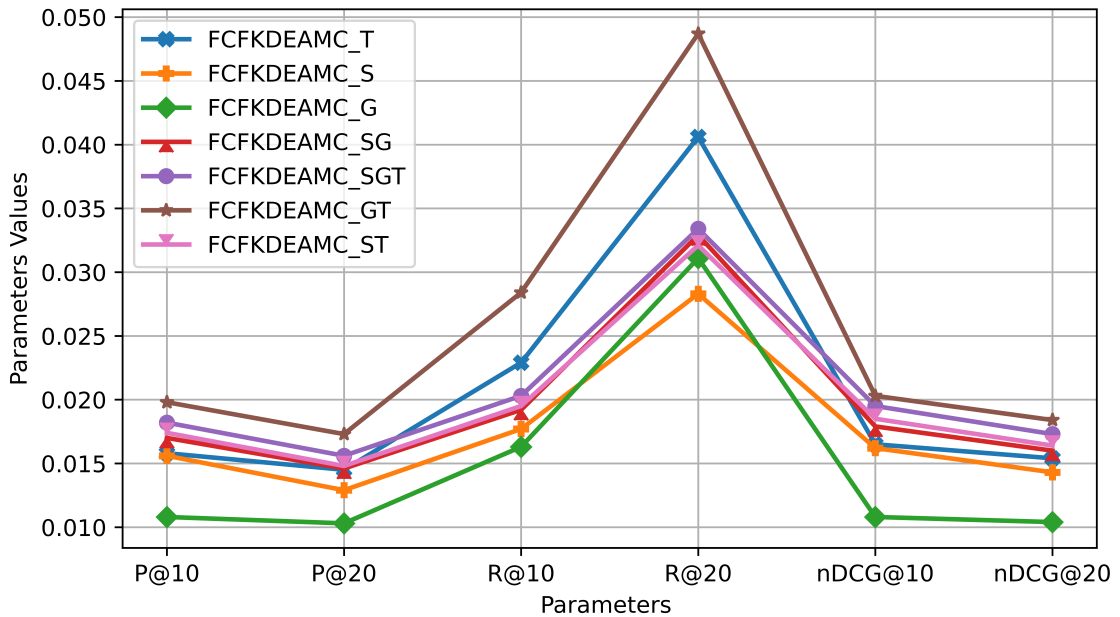


Figure 5.5: FCFKDEAMC algorithm performance for different permutations of SGTC context on Yelp dataset

- Overall, the GeoSoCa_GC model appears to have an edge over the other models, particularly in recall and nDCG, making it potentially the most effective among the GeoSoCa variants for scenarios where both relevance and ranking are critical.

For Fig. 5.5, the following observations can be made:

- All models show similar performance in terms of precision, with FCFKDEAMC_SG and FCFKDEAMC_T slightly outperforming the others. This suggests that these two models are more effective in predicting the most relevant items at the top of the list.
- Recall metrics show a wider variation, with FCFKDEAMC_SG clearly leading at R@10, indicating its effectiveness in capturing relevant items within the top 10. This makes it particularly suitable for scenarios where maximizing recall in a smaller recommendation set is essential.
- The nDCG values indicate that FCFKDEAMC_SG excels in ranking as well, consistently outperforming other models. The decline in nDCG@20 for models like FCFKDEAMC_GT suggests that while these models can identify relevant items, they may struggle with ranking them effectively in longer lists.

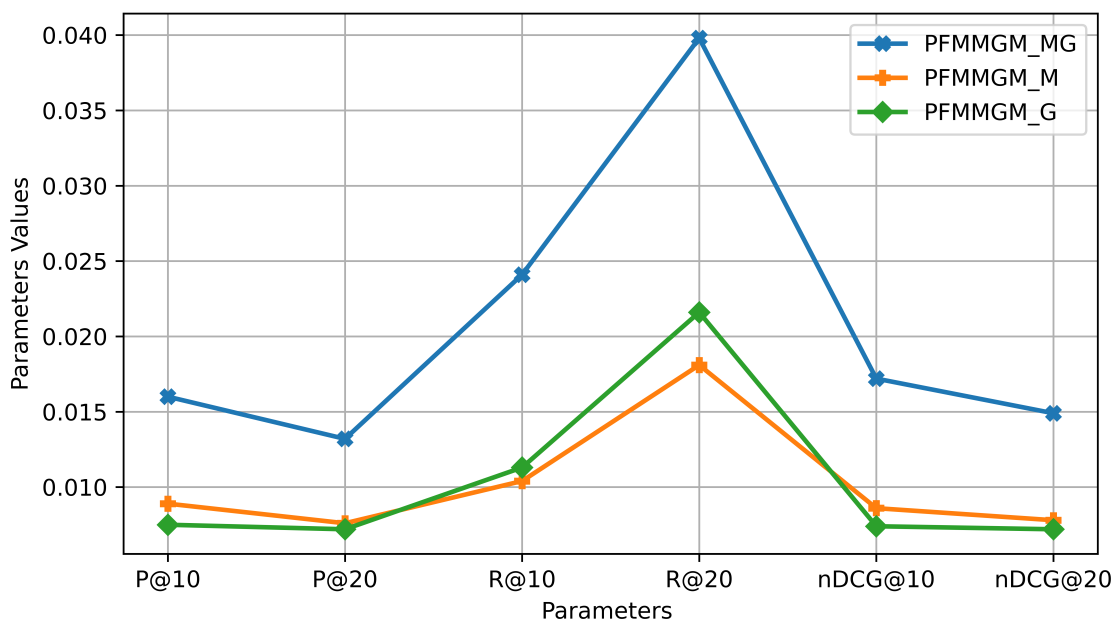


Figure 5.6: PFMMGM algorithm performance for different permutations of SGTC context on Yelp dataset

- Overall, the FCFKDEAMC_SG model emerges as the best performer among the FCFKDEAMC variants, particularly excelling in recall and nDCG metrics.

For Fig. 5.6, the following observations can be made:

- All three models perform similarly in terms of precision at both P@10 and P@20.
- PFMMGM_MG shows a notable improvement in R@10, indicating its effectiveness in identifying relevant items within the top 10 recommendations. This model also performs well at R@20.
- PFMMGM_MG outperforms the other models for nDCG@10 and nDCG@20.
- Overall, the PFMMGM_MG model outperforms its counterparts, particularly in recall and nDCG, indicating its superior ability to rank relevant items effectively.

For Fig. 5.7, the following observations can be made:

- GeoSoCa_GS consistently shows superior performance across both P@10 and P@20, suggesting that it is the most effective in accurately predicting the top items. GeoSoCa_S follows closely, while GeoSoCa_G lags slightly behind.

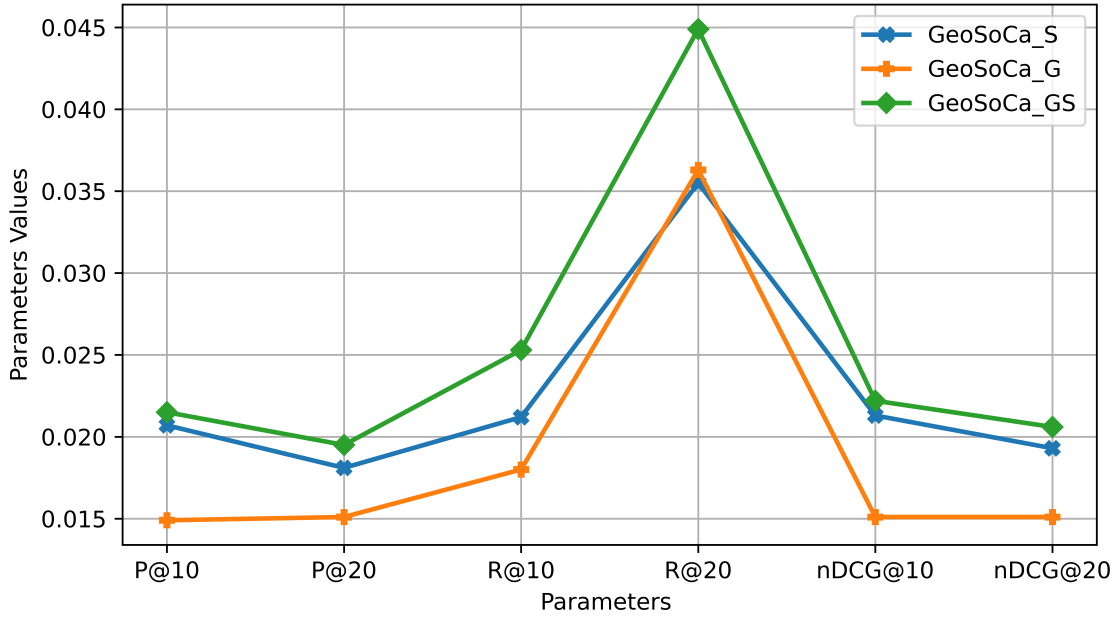


Figure 5.7: GeoSoCa algorithm performance for different permutations of SGT context on Gowalla dataset

- The recall metrics, R@10 and R@20, provide insights into the models' ability to capture all relevant items within the top 10 and top 20 recommendations. GeoSoCa_GS again outperforms other models, especially at R@10, indicating its effectiveness in ensuring that relevant items are included in the top 10 recommendations.
- GeoSoCa_GS continues to lead in nDCG@10, underscoring its ability to rank relevant items effectively. The performance remains strong at nDCG@20, although the margin between GeoSoCa_GS and other models decreases slightly.
- The results suggest that incorporating both geographical and social contextual information in a balanced manner, as likely implemented in GeoSoCa_GS, leads to better performance in location-based recommendation tasks.

Figure 5.8 evaluates various configurations of the FCFKDEAMC model, including FCFKDEAMC_SG, FCFKDEAMC_S, FCFKDEAMC_T, FCFKDEAMC_SGT, FCFKDEAMC_ST, FCFKDEAMC_GT, and FCFKDEAMC_G based on different combinations of social, geographical, and temporal factors. The following observations can be made:

- FCFKDEAMC_SG emerges as the top performer, particularly at P@10. FCFKDEAMC_T and FCFKDEAMC_S also perform well, but FCFKDEAMC_GT and FCFKDEAMC_G

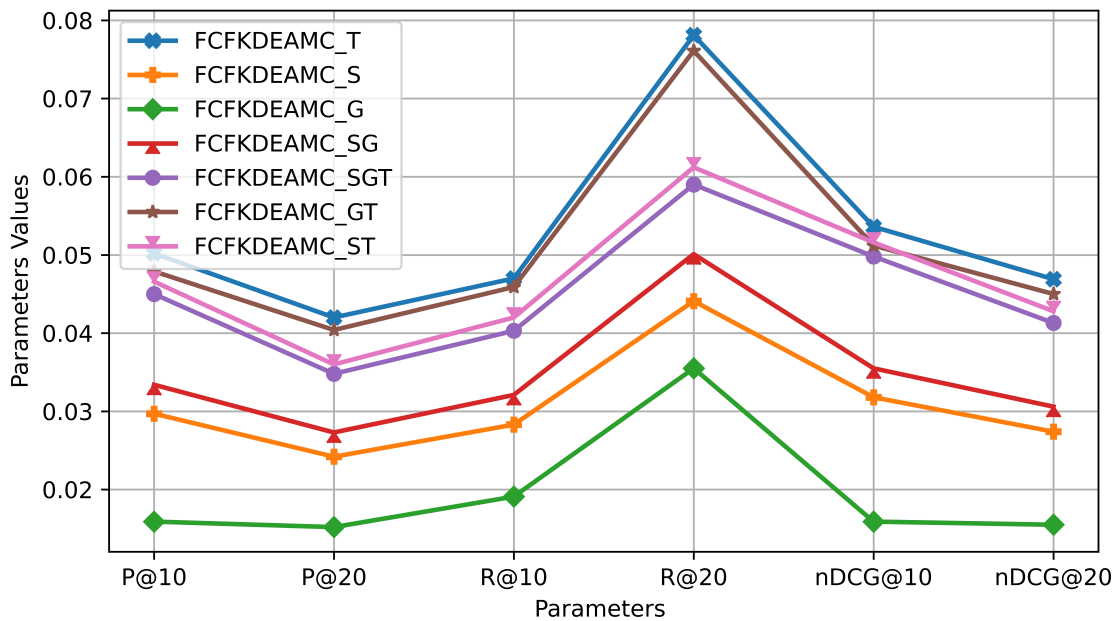


Figure 5.8: FCFKDEAMC algorithm performance for different permutations of SGT context on Gowalla dataset

struggle, indicating that these models may have difficulty with precision when geographical factors are heavily weighted.

- FCFKDEAMC_SG leads again, especially at R@10, and is particularly effective in capturing relevant items in the top 10 recommendations, making it a strong candidate for applications where recall is critical.
- The FCFKDEAMC_SG model stands out as the most effective among the FCFKDEAMC variants, particularly in terms of precision, recall, and nDCG.

For Fig. 5.9, the following observations can be made:

- The PFMMGM_MG model clearly outperforms the other two models, particularly at P@10. The performance of PFMMGM_G and PFMMGM_M is relatively similar, with PFMMGM_G slightly ahead at P@20.
- The recall metrics show a significant advantage for PFMMGM_MG at both R@10 and R@20.
- The PFMMGM_MG model emerges as the best performer among the PFMMGM variants, particularly in precision, recall, and nDCG.

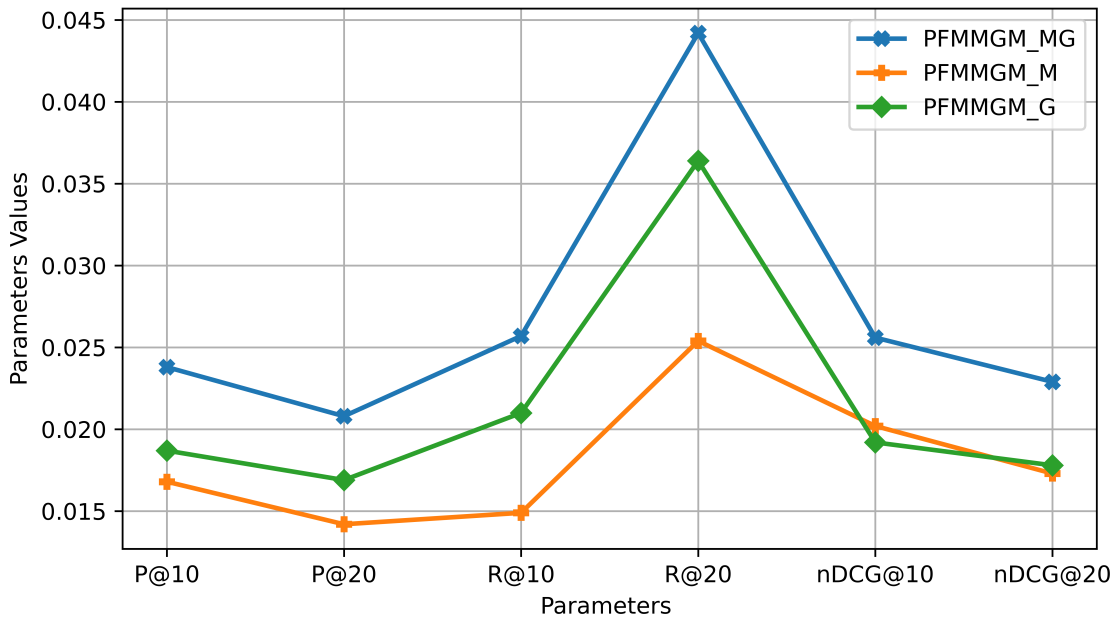


Figure 5.9: PFMMGM algorithm performance for different permutations of SGT context on Gowalla dataset

5.5 Conclusions

This chapter develops a novel Contextual-POI-Bandit framework that models POI recommendation as sequential decision making problem under uncertainty and provides personalized next Point of Interest (POI) recommendation by fusing together social, geographical, temporal and categorical contextual information. By analysing various contextual information models, it is evident that temporal and geographical data significantly enhance model performance, while categorical information alone is insufficient. Although combining all available contextual data might seem advantageous, it does not always yield better outcomes and can sometimes hinder performance. Notably, the effectiveness of models is heavily influenced by the ratio of users to check-ins within datasets. The contextual multi-arm models having Fuzzy Neural Network (FNN) as its base generally outperform matrix factorization models, likely due to their ability to capture complex, non-linear relationships between users and POIs. Furthermore, recommendations tend to be more accurate when users remain within smaller geographical areas, while the accuracy diminishes as users explore more widely and as the time intervals between consecutive check-ins increase. Overall, geographical and temporal contexts emerge as the most valuable in enhancing the performance of location-based recommendation systems. To

this end, following key conclusions of the presented work are as follows.

1. An innovative online framework named Contextual-POI-Bandit is developed, which leverages contextual multi-armed bandits (CMAB), to mathematically formulate the problem of next POI recommendation. The framework uses CMAB to model the recommendation problem as a series of decisions where each "arm" of the bandit represents a possible POI to recommend. The framework fuses the social, geographic, temporal, and categorical (SGTC) correlations between users and POIs into a combined user context feature vector matrix to optimize the selection of the next POI.
2. The proposed Contextual-POI-Bandit for the next POI recommendation balances both the exploratory and the exploitative behaviour of a user while generating personalized recommendations. The exploratory behaviour means the user wants to seek out new experiences and try unfamiliar POIs out of curiosity and a desire to discover new preferences. The exploitative behavior, on the other hand, focuses on selecting well-known POIs that the user has previously enjoyed, aiming to maximize satisfaction based on past experiences. Contextual-POI-Bandit achieves this by uniquely incorporating user feedback for the recommended POIs and training the underlying model based on the feedback received, enabling the recommendations to adjust dynamically to the user's behaviour and needs over time.
3. The proposed framework is designed with remarkable flexibility, allowing it to integrate both the existing best Contextual Multi-Armed Bandits (CMAB) models and any future, improved CMAB models. This adaptability is a significant strength, ensuring that the framework can leverage state-of-the-art advancements in CMAB techniques as they emerge. By accommodating the best current models, our framework ensures that recommendations are generated with high accuracy. This means that users receive suggestions that are most relevant to their preferences and behaviours, maximizing their satisfaction and engagement. The precision of these recommendations is critical in providing a seamless and personalized user experience, making the system more effective in meeting individual user needs. Moreover, the ability to incorporate future CMAB models means that the framework is not static but rather evolves alongside advancements in the field. As new models are developed that offer improved algorithms and methodologies, they can be seamlessly integrated into the framework. This ensures that the recommendations continue to

improve over time, maintaining a cutting-edge approach to personalization. Thus, our proposed framework stands out for its ability to integrate both the best existing and future CMAB models. This flexibility not only guarantees high accuracy in recommendations but also enables continuous enhancement of personalization.

4. The proposed CMAB algorithm, Neuro-Fuzzy Bandit, to seamlessly integrate with the proposed framework for next POI recommendation. The developed model leverages the capabilities of neural networks as well as the interpretability of fuzzy logic to offer robust solutions for complex sequential decision-making problems under uncertainty. Neural networks can process vast amounts of data to identify patterns and trends and model complex relationships between inputs and outputs effectively. Fuzzy logic, on the other hand, excels in dealing with uncertainty and is useful in real-world scenarios where information is often imprecise or incomplete. The integration of fuzzy units into the Neuro-Fuzzy Bandit model enables the model to craft fuzzy rules to represent expert knowledge and handle ambiguous data effectively, enhancing the model's interpretability.
5. The state-of-the-art and popular CMAB models are integrated into the proposed framework, and comprehensive empirical evaluation is performed to assess their performance on various metrics such as accuracy, precision, recall, etc. Extensive optimization efforts to fine-tune these models are undertaken, aiming to maximize their performance in the specific context of the next POI recommendations. Further, to evaluate the effectiveness of these integrated CMAB models, empirical tests are conducted using several benchmark datasets, namely Yelp, Gowalla, and New York. These datasets provide a diverse and comprehensive basis for assessing the models' performance.

Chapter 6

Conclusion and Future Scope

6.1 Introduction

In the current digital world, providing relevant content and personalized services has become crucial to cater individual user preferences. Several applications, such as online advertising and recommendation systems, rely on understanding user behavior and preferences to deliver optimal content. Although significant advances have been made in recommendation algorithms, challenges in personalization persist. This is due to the inherent complexity of balancing exploration and exploitation in recommendation systems. Contextual bandit algorithms are particularly relevant in this context as they bridge this gap by enabling systems to learn and adapt recommendations based on real-time feedback.

The Role of Contextual Bandits in Personalization

The Contextual bandit algorithm addresses the exploration-exploitation trade-off by dynamically adjusting recommendations based on user interactions. Unlike conventional recommendation systems that rely on static data, contextual bandits continuously learn and adapt with each interaction, using the available context. For example, when displaying an advertisement on a website, the contextual bandit algorithm must decide whether to show an ad previously clicked by the user (exploitation) or to explore a new ad based on the current context, such as the time of day or the user's browsing history.

Although contextual bandits hold promise for enhancing personalization, they often operate as black boxes, leading to transparency and accountability issues. Understanding the decision-making process of these algorithms is crucial, especially in applications

where the consequences of incorrect decisions can be significant. To address these concerns, explainable AI (XAI) approaches are being integrated into contextual bandit models.

FuzzyBandit: An explainable AI (XAI) Based Model

The FuzzyBandit model represents a novel approach at the intersection of contextual bandits and XAI. In this model, each arm is an Adaptive Neuro-Fuzzy Inference System (ANFIS) designed to handle the inherent imprecision of arm selection in contextual bandit settings. The model adjusts its parameters based on feedback to ensure that the most relevant and diverse features are used to maximize rewards.

What sets the FuzzyBandit model apart is its ability to provide understandable explanations for its decisions. By optimizing its decision-making process based on previous observations, the model improves its performance over time while offering interpretable insights into its decision-making. This transparency is crucial for building trust in the recommendations provided by the model, especially in domains where accountability is essential.

Empirical testing of the FuzzyBandit model has demonstrated its competitiveness against several popular models across multiple benchmark datasets. The model's efficiency was assessed using parameters such as recall, specificity, precision, F1 score, and accuracy. Additionally, a mathematical framework has been developed to calculate a trust score for the FuzzyBandit model, enhancing its reliability and aiding in error detection.

Hybrid Neuro Bandit Model for Online Recommendations

A practical application of contextual bandit algorithms is the development of a Hybrid Neuro Bandit (HNB) model, designed for online recommendations. For example, consider a system like YouTube that recommends ads to users while they watch video content. The system aims to display the most relevant ads to increase user engagement, typically measured by click-through rates. The challenge lies in balancing exploration (showing new ads) and exploitation (showing ads previously clicked by users).

The HNB model addresses this challenge by incorporating advice from a pool of experts, each representing a contextual multi-armed bandit model. These experts make recommendations based on various contextual factors, and their contributions are combined

within the HNB model. The model assigns weights to each expert, which are updated using a neural network trained with user feedback. This approach enhances personalization by leveraging multiple CMAB models, providing more relevant and timely recommendations.

The performance of the HNB model has been rigorously tested against other state-of-the-art CMAB models using nine different metrics, including specificity, precision, and Matthews Correlation Coefficient (MCC). The HNB model has proven to be superior in delivering real-time, personalized content and improving the overall user experience.

Contextual-Bandit-POI: Personalized Next Point-of-Interest Recommendations

Contextual bandits have significant applications in location-based services, where users share their experiences at various Points-of-Interest (POIs) such as cafes, restaurants, and hotels. The check-in data collected, including timestamps, comments, and GPS coordinates, provides valuable contextual information for predicting future user behavior. However, predicting the next POI for a user is challenging due to issues such as data sparsity and the cold start problem, where limited historical data makes accurate recommendations difficult.

To improve next POI recommendations, it is essential to consider spatial-temporal intervals and user preferences. Traditional models, such as those based on Recurrent Neural Networks (RNNs), often struggle to capture diverse user behaviors over long periods. Enhanced models using Long Short-Term Memory (LSTM) or Gated Recurrent Units (GRUs) address some of these limitations but still fall short in personalizing recommendations and adapting to dynamic user preferences.

The Contextual-POI-Bandit framework was developed to address these challenges by integrating various CMAB algorithms into the recommendation process. This framework balances exploration and exploitation by suggesting both new and familiar POIs based on the user's past behavior. The framework is flexible, designed to incorporate current and future CMAB models.

It includes a novel Fuzzy Neural Network (FNN) algorithm that combines neural networks' data processing capabilities with fuzzy logic's interpretability to handle uncertainty effectively. The framework has been empirically validated against benchmark datasets, showing significant improvements in recommendation accuracy and relevance.

6.2 Future Scope

Significant advancements have been made in developing models like FuzzyBandit, Hybrid Neuro Bandit, and Contextual-POI-Bandit for personalized recommendations. These models address the exploration-exploitation dilemma and incorporate new explainable AI (XAI) techniques, enhancing performance and transparency.

The FuzzyBandit model stands out for its clarity and interpretability, providing understandable explanations for decisions and promoting trust in the system. Its performance across various benchmark datasets and the introduction of a trust score framework further validate its effectiveness for personalized recommendations.

The Hybrid Neuro Bandit (HNB) model excels in dynamic weight updating and adaptation to changing user preferences, offering a more engaging and satisfying user experience. The Contextual-POI-Bandit framework effectively predicts the next POI by considering both short- and long-term user preferences and spatial-temporal intervals, overcoming challenges like data sparsity and the cold start problem.

Looking forward, there are several avenues for further research. One direction is to integrate more XAI into contextual bandit models, enhancing their transparency and accountability. Additionally, these online learning settings could be applied to other domains beyond POI recommendations, leading to further improvements in personalization and adaptability.

References

- [1] H. Robbins, “Some aspects of the sequential design of experiments,” Bulletin of the American Mathematical Society, vol. 58, no. 5, pp. 527 – 535, 1952.
- [2] M. Tokic, “Adaptive ϵ -greedy exploration in reinforcement learning based on value differences,” in Annual conference on artificial intelligence. Springer, 2010, pp. 203–210.
- [3] J. Langford and T. Zhang, “The epoch-greedy algorithm for multi-armed bandits with side information,” Advances in neural information processing systems, vol. 20, 2007.
- [4] O. Chapelle and L. Li, “An empirical evaluation of thompson sampling,” Advances in neural information processing systems, vol. 24, 2011.
- [5] S. Agrawal and N. Goyal, “Analysis of thompson sampling for the multi-armed bandit problem,” in Conference on learning theory. JMLR Workshop and Conference Proceedings, 2012, pp. 39–1.
- [6] L. Li, W. Chu, J. Langford, and R. E. Schapire, “A contextual-bandit approach to personalized news article recommendation,” in Proceedings of the 19th international conference on World wide web, 2010, pp. 661–670.
- [7] W. Chu, L. Li, L. Reyzin, and R. Schapire, “Contextual bandits with linear payoff functions,” in Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics. JMLR Workshop and Conference Proceedings, 2011, pp. 208–214.
- [8] P. Auer, N. Cesa-Bianchi, Y. Freund, and R. E. Schapire, “Gambling in a rigged casino: The adversarial multi-armed bandit problem,” in Proceedings of IEEE 36th annual foundations of computer science. IEEE, 1995, pp. 322–331.

References

- [9] T. L. Lai and H. Robbins, “Asymptotically efficient adaptive allocation rules,” Advances in applied mathematics, vol. 6, no. 1, pp. 4–22, 1985.
- [10] P. Auer, N. Cesa-Bianchi, and P. Fischer, “Finite-time analysis of the multiarmed bandit problem,” Machine learning, vol. 47, pp. 235–256, 2002.
- [11] P. Auer, N. Cesa-Bianchi, Y. Freund, and R. E. Schapire, “The nonstochastic multiarmed bandit problem,” SIAM journal on computing, vol. 32, no. 1, pp. 48–77, 2002.
- [12] P. Auer and N. Cesa-Bianchi, “On-line learning with malicious noise and the closure algorithm,” Annals of mathematics and artificial intelligence, vol. 23, pp. 83–99, 1998.
- [13] W. R. Thompson, “On the likelihood that one unknown probability exceeds another in view of the evidence of two samples,” Biometrika, vol. 25, no. 3-4, pp. 285–294, 1933.
- [14] D. Chen, X. Cao, L. Wang, F. Wen, and J. Sun, “Bayesian face revisited: A joint formulation,” in Computer Vision—ECCV 2012: 12th European Conference on Computer Vision, Florence, Italy, October 7-13, 2012, Proceedings, Part III 12. Springer, 2012, pp. 566–579.
- [15] J. Vermorel and M. Mohri, “Multi-armed bandit algorithms and empirical evaluation,” in European conference on machine learning. Springer, 2005, pp. 437–448.
- [16] M. Dudik, D. Hsu, S. Kale, N. Karampatziakis, J. Langford, L. Reyzin, and T. Zhang, “Efficient optimal learning for contextual bandits,” arXiv preprint arXiv:1106.2369, 2011.
- [17] S. M. Kakade, S. Shalev-Shwartz, and A. Tewari, “Efficient bandit algorithms for online multiclass prediction,” in Proceedings of the 25th international conference on Machine learning, 2008, pp. 440–447.
- [18] F. Rosenblatt, “The perceptron: a probabilistic model for information storage and organization in the brain.” Psychological review, vol. 65, no. 6, p. 386, 1958.
- [19] K. Crammer and C. Gentile, “Multiclass classification with bandit feedback using adaptive regularization,” Machine learning, vol. 90, no. 3, pp. 347–383, 2013.

- [20] C. Riquelme, G. Tucker, and J. Snoek, “Deep bayesian bandits showdown: An empirical comparison of bayesian deep networks for thompson sampling,” arXiv preprint arXiv:1802.09127, 2018.
- [21] R. Allesiardo, R. Féraud, and D. Bouneffouf, “A neural networks committee for the contextual bandit problem,” in Neural Information Processing: 21st International Conference, ICONIP 2014, Kuching, Malaysia, November 3-6, 2014. Proceedings, Part I 21. Springer, 2014, pp. 374–381.
- [22] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, “Dropout: a simple way to prevent neural networks from overfitting,” The journal of machine learning research, vol. 15, no. 1, pp. 1929–1958, 2014.
- [23] Y. Gao, T. Sheng, Y. Xiang, Y. Xiong, H. Wang, and J. Zhang, “Chat-rec: Towards interactive and explainable llms-augmented recommender system,” arXiv preprint arXiv:2303.14524, 2023.
- [24] J. Chen, L. Ma, X. Li, N. Thakurdesai, J. Xu, J. H. Cho, K. Nag, E. Korpeoglu, S. Kumar, and K. Achan, “Knowledge graph completion models are few-shot learners: An empirical study of relation labeling in e-commerce with llms,” arXiv preprint arXiv:2305.09858, 2023.
- [25] X. Chen, W. Fan, J. Chen, H. Liu, Z. Liu, Z. Zhang, and Q. Li, “Fairly adaptive negative sampling for recommendations,” in Proceedings of the ACM Web Conference 2023, 2023, pp. 3723–3733.
- [26] W. Fan, X. Zhao, X. Chen, J. Su, J. Gao, L. Wang, Q. Liu, Y. Wang, H. Xu, L. Chen et al., “A comprehensive survey on trustworthy recommender systems,” arXiv preprint arXiv:2209.10117, 2022.
- [27] S. Zhang, L. Yao, A. Sun, and Y. Tay, “Deep learning based recommender system: A survey and new perspectives,” ACM computing surveys (CSUR), vol. 52, no. 1, pp. 1–38, 2019.
- [28] C. Liu, W. Fan, Y. Liu, J. Li, H. Li, H. Liu, J. Tang, and Q. Li, “Generative diffusion models on graphs: Methods and applications,” arXiv preprint arXiv:2302.02591, 2023.

References

- [29] B. Hidasi, A. Karatzoglou, L. Baltrunas, and D. Tikk, “Session-based recommendations with recurrent neural networks,” [arXiv preprint arXiv:1511.06939](#), 2015.
- [30] W. Fan, Y. Ma, D. Yin, J. Wang, J. Tang, and Q. Li, “Deep social collaborative filtering,” in [Proceedings of the 13th ACM conference on recommender systems](#), 2019, pp. 305–313.
- [31] W. Fan, Y. Ma, Q. Li, Y. He, E. Zhao, J. Tang, and D. Yin, “Graph neural networks for social recommendation,” in [The world wide web conference](#), 2019, pp. 417–426.
- [32] Z. Qiu, X. Wu, J. Gao, and W. Fan, “U-bert: Pre-training user representations for improved recommendation,” in [Proceedings of the AAAI Conference on Artificial Intelligence](#), vol. 35, no. 5, 2021, pp. 4320–4327.
- [33] Z. Chen, H. Mao, H. Li, W. Jin, H. Wen, X. Wei, S. Wang, D. Yin, W. Fan, H. Liu [et al.](#), “Exploring the potential of large language models (llms) in learning on graphs,” [ACM SIGKDD Explorations Newsletter](#), vol. 25, no. 2, pp. 42–61, 2024.
- [34] W. X. Zhao, K. Zhou, J. Li, T. Tang, X. Wang, Y. Hou, Y. Min, B. Zhang, J. Zhang, Z. Dong [et al.](#), “A survey of large language models,” [arXiv preprint arXiv:2303.18223](#), 2023.
- [35] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, “Attention is all you need,” [Advances in neural information processing systems](#), vol. 30, 2017.
- [36] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, “Bert: Pre-training of deep bidirectional transformers for language understanding,” [arXiv preprint arXiv:1810.04805](#), 2018.
- [37] C. Raffel, N. Shazeer, A. Roberts, K. Lee, S. Narang, M. Matena, Y. Zhou, W. Li, and P. J. Liu, “Exploring the limits of transfer learning with a unified text-to-text transformer,” [Journal of machine learning research](#), vol. 21, no. 140, pp. 1–67, 2020.

- [38] W.-C. Kang, J. Ni, N. Mehta, M. Sathiamoorthy, L. Hong, E. Chi, and D. Z. Cheng, “Do llms understand user preferences? evaluating llms on user rating prediction,” [arXiv preprint arXiv:2305.06474](#), 2023.
- [39] A. Zhiyuli, Y. Chen, X. Zhang, and X. Liang, “Bookgpt: A general framework for book recommendation empowered by large language model,” [arXiv preprint arXiv:2305.15673](#), 2023.
- [40] K. Bao, J. Zhang, Y. Zhang, W. Wang, F. Feng, and X. He, “Tallrec: An effective and efficient tuning framework to align large language model with recommendation,” in [Proceedings of the 17th ACM Conference on Recommender Systems](#), 2023, pp. 1007–1014.
- [41] Z. Cui, J. Ma, C. Zhou, J. Zhou, and H. Yang, “M6-rec: Generative pre-trained language models are open-ended recommender systems,” [arXiv preprint arXiv:2205.08084](#), 2022.
- [42] Z. Chen, “Palr: Personalization aware llms for recommendation,” [arXiv preprint arXiv:2305.07622](#), 2023.
- [43] S. Geng, S. Liu, Z. Fu, Y. Ge, and Y. Zhang, “Recommendation as language processing (rlp): A unified pretrain, personalized prompt & predict paradigm (p5),” in [Proceedings of the 16th ACM Conference on Recommender Systems](#), 2022, pp. 299–315.
- [44] F. Petroni, T. Rocktäschel, P. Lewis, A. Bakhtin, Y. Wu, A. H. Miller, and S. Riedel, “Language models as knowledge bases?” [arXiv preprint arXiv:1909.01066](#), 2019.
- [45] A. Roberts, C. Raffel, and N. Shazeer, “How much knowledge can you pack into the parameters of a language model?” [arXiv preprint arXiv:2002.08910](#), 2020.
- [46] F. Petroni, P. Lewis, A. Piktus, T. Rocktäschel, Y. Wu, A. H. Miller, and S. Riedel, “How context affects language models’ factual predictions,” [arXiv preprint arXiv:2005.04611](#), 2020.
- [47] Z. Jiang, F. F. Xu, J. Araki, and G. Neubig, “How can we know what language models know?” [Transactions of the Association for Computational Linguistics](#), vol. 8, pp. 423–438, 2020.

References

- [48] C. Wang, X. Liu, and D. Song, “Language models are open knowledge graphs,” arXiv preprint arXiv:2010.11967, 2020.
- [49] N. Poerner, U. Waltinger, and H. Schütze, “E-bert: Efficient-yet-effective entity embeddings for bert,” arXiv preprint arXiv:1911.03681, 2019.
- [50] B. Heinzerling and K. Inui, “Language models as knowledge bases: On entity representations, storage capacity, and paraphrased queries,” arXiv preprint arXiv:2008.09036, 2020.
- [51] C. Wang, P. Liu, and Y. Zhang, “Can generative pre-trained language models serve as knowledge bases for closed-book qa?” arXiv preprint arXiv:2106.01561, 2021.
- [52] K. Guu, K. Lee, Z. Tung, P. Pasupat, and M. Chang, “Retrieval augmented language model pre-training,” in International conference on machine learning. PMLR, 2020, pp. 3929–3938.
- [53] A. Bordes, N. Usunier, A. Garcia-Duran, J. Weston, and O. Yakhnenko, “Translating embeddings for modeling multi-relational data,” Advances in neural information processing systems, vol. 26, 2013.
- [54] Y. Zhu, X. Wang, J. Chen, S. Qiao, Y. Ou, Y. Yao, S. Deng, H. Chen, and N. Zhang, “Llms for knowledge graph construction and reasoning: Recent capabilities and future opportunities,” arXiv preprint arXiv:2305.13168, 2023.
- [55] Z. Zhang, X. Liu, Y. Zhang, Q. Su, X. Sun, and B. He, “Pretrain-kge: learning knowledge representation from pretrained language models,” in Findings of the Association for Computational Linguistics: EMNLP 2020, 2020, pp. 259–266.
- [56] A. Kumar, A. Pandey, R. Gadia, and M. Mishra, “Building knowledge graph using pre-trained language model for learning entity-aware relationships,” in 2020 IEEE International Conference on Computing, Power and Communication Technologies (GUCON). IEEE, 2020, pp. 310–315.
- [57] S. Razniewski, A. Yates, N. Kassner, and G. Weikum, “Language models as or for knowledge bases,” arXiv preprint arXiv:2110.04888, 2021.
- [58] S. Liu, C. Gao, Y. Chen, D. Jin, and Y. Li, “Learnable embedding sizes for recommender systems,” arXiv preprint arXiv:2101.07577, 2021.

- [59] H. Liu, X. Zhao, C. Wang, X. Liu, and J. Tang, “Automated embedding size search in deep recommender systems,” in Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval, 2020, pp. 2307–2316.
- [60] W. Deng, J. Pan, T. Zhou, D. Kong, A. Flores, and G. Lin, “Deelight: Deep lightweight feature interactions for accelerating ctr predictions in ad serving,” in Proceedings of the 14th ACM international conference on Web search and data mining, 2021, pp. 922–930.
- [61] A. A. Ginart, M. Naumov, D. Mudigere, J. Yang, and J. Zou, “Mixed dimension embeddings with application to memory-efficient recommendation systems,” in 2021 IEEE International Symposium on Information Theory (ISIT). IEEE, 2021, pp. 2786–2791.
- [62] H. Wang, C. Focke, R. Sylvester, N. Mishra, and W. Wang, “Fine-tune bert for docred with two-step process,” arXiv preprint arXiv:1909.11898, 2019.
- [63] H. Yan, T. Gui, J. Dai, Q. Guo, Z. Zhang, and X. Qiu, “A unified generative framework for various ner subtasks,” arXiv preprint arXiv:2106.01223, 2021.
- [64] B. Li, W. Yin, and M. Chen, “Ultra-fine entity typing with indirect supervision from natural language inference,” Transactions of the Association for Computational Linguistics, vol. 10, pp. 607–622, 2022.
- [65] Y. Kirstain, O. Ram, and O. Levy, “Coreference resolution without span representations,” arXiv preprint arXiv:2101.00434, 2021.
- [66] A. Cattan, A. Eirew, G. Stanovsky, M. Joshi, and I. Dagan, “Cross-document coreference resolution over predicted mentions,” arXiv preprint arXiv:2106.01210, 2021.
- [67] S. Lyu and H. Chen, “Relation classification with entity type restriction,” arXiv preprint arXiv:2105.08393, 2021.
- [68] J. Han, N. Collier, W. Buntine, and E. Shareghi, “Pive: Prompting with iterative verification improving graph-based generative capability of llms,” arXiv preprint arXiv:2305.12392, 2023.

References

- [69] M. Trajanoska, R. Stojanov, and D. Trajanov, “Enhancing knowledge graph construction using large language models,” [arXiv preprint arXiv:2305.04676](#), 2023.
- [70] J. Wei, Y. Tay, R. Bommasani, C. Raffel, B. Zoph, S. Borgeaud, D. Yogatama, M. Bosma, D. Zhou, D. Metzler *et al.*, “Emergent abilities of large language models,” [arXiv preprint arXiv:2206.07682](#), 2022.
- [71] J. Wei, X. Wang, D. Schuurmans, M. Bosma, F. Xia, E. Chi, Q. V. Le, D. Zhou *et al.*, “Chain-of-thought prompting elicits reasoning in large language models,” [Advances in neural information processing systems](#), vol. 35, pp. 24 824–24 837, 2022.
- [72] L. Wang and E.-P. Lim, “Zero-shot next-item recommendation using large pre-trained language models,” [arXiv preprint arXiv:2304.03153](#), 2023.
- [73] T.-H. Wen, D. Vandyke, N. Mrksic, M. Gasic, L. M. Rojas-Barahona, P.-H. Su, S. Ultes, and S. Young, “A network-based end-to-end trainable task-oriented dialogue system,” [arXiv preprint arXiv:1604.04562](#), 2016.
- [74] Y. Zhang, S. Sun, M. Galley, Y.-C. Chen, C. Brockett, X. Gao, J. Gao, J. Liu, and B. Dolan, “Dialogpt: Large-scale generative pre-training for conversational response generation,” [arXiv preprint arXiv:1911.00536](#), 2019.
- [75] K. Yao, G. Zweig, M.-Y. Hwang, Y. Shi, and D. Yu, “Recurrent neural networks for language understanding.” in [Interspeech](#), 2013, pp. 2524–2528.
- [76] G. Mesnil, X. He, L. Deng, and Y. Bengio, “Investigation of recurrent-neural-network architectures and learning methods for spoken language understanding.” in [Interspeech](#), 2013, pp. 3771–3775.
- [77] N. Mrkšić, D. O. Séaghdha, T.-H. Wen, B. Thomson, and S. Young, “Neural belief tracker: Data-driven dialogue state tracking,” [arXiv preprint arXiv:1606.03777](#), 2016.
- [78] H. Cuayáhuitl, S. Keizer, and O. Lemon, “Strategic dialogue management via deep reinforcement learning,” [arXiv preprint arXiv:1511.08099](#), 2015.

- [79] H. Zhou, M. Huang, and X. Zhu, “Context-aware natural language generation for spoken dialogue systems,” in Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers, 2016, pp. 2032–2041.
- [80] O. Dušek and F. Jurčiček, “Sequence-to-sequence generation for spoken dialogue via deep syntax trees and strings,” arXiv preprint arXiv:1606.05491, 2016.
- [81] X. Zhang, Y. Zou, H. Zhang, J. Zhou, S. Diao, J. Chen, Z. Ding, Z. He, X. He, Y. Xiao et al., “Automatic product copywriting for e-commerce,” in Proceedings of the AAAI Conference on Artificial Intelligence, vol. 36, no. 11, 2022, pp. 12 423–12 431.
- [82] Z. Lei, C. Zhang, X. Xu, W. Wu, Z.-Y. Niu, H. Wu, H. Wang, Y. Yang, and S. Li, “Plato-ad: A unified advertisement text generation framework with multi-task prompt learning,” in Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing: Industry Track, 2022, pp. 512–520.
- [83] S. Thomaidou, I. Lourentzou, P. Katsivelis-Perakis, and M. Vazirgiannis, “Automated snippet generation for online advertising,” in Proceedings of the 22nd ACM international conference on Information & Knowledge Management, 2013, pp. 1841–1844.
- [84] K. Bartz, C. Barr, and A. Aijaz, “Natural language generation for sponsored-search advertisements,” in Proceedings of the 9th ACM Conference on Electronic Commerce, 2008, pp. 1–9.
- [85] A. Fujita, K. Ikushima, S. Sato, R. Kamite, K. Ishiyama, and O. Tamachi, “Automatic generation of listing ads by reusing promotional texts,” in Proceedings of the 12th international conference on electronic commerce: Roadmap for the future of electronic business, 2010, pp. 179–188.
- [86] J. W. Hughes, K.-h. Chang, and R. Zhang, “Generating better search engine text advertisements with deep reinforcement learning,” in Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, 2019, pp. 2269–2277.

References

- [87] X. Wang, X. Gu, J. Cao, Z. Zhao, Y. Yan, B. Middha, and X. Xie, “Reinforcing pretrained models for generating attractive text advertisements,” in Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining, 2021, pp. 3697–3707.
- [88] C. Chen, X. Wang, X. Yi, F. Wu, X. Xie, and R. Yan, “Personalized chit-chat generation for recommendation using external chat corpora,” in Proceedings of the 28th ACM SIGKDD conference on knowledge discovery and data mining, 2022, pp. 2721–2731.
- [89] Y. S. Kanungo, S. Negi, and A. Rajan, “Ad headline generation using self-critical masked language model,” in Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies: Industry Papers, 2021, pp. 263–271.
- [90] P. Wei, X. Yang, S. Liu, L. Wang, and B. Zheng, “Creator: Ctr-driven advertising text generation with controlled pre-training and contrastive fine-tuning,” arXiv preprint arXiv:2205.08943, 2022.
- [91] Y. S. Kanungo, G. Das, and S. Negi, “Cobart: Controlled, optimized, bidirectional and auto-regressive transformer for ad headline generation,” in Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, 2022, pp. 3127–3136.
- [92] Q. Chen, J. Lin, Y. Zhang, H. Yang, J. Zhou, and J. Tang, “Towards knowledge-based personalized product description generation in e-commerce,” in Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, 2019, pp. 3040–3050.
- [93] W. Chu, L. Li, L. Reyzin, and R. Schapire, “Contextual bandits with linear payoff functions,” in Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics. JMLR Workshop and Conference Proceedings, 2011, pp. 208–214.
- [94] K. Crammer and C. Gentile, “Multiclass classification with bandit feedback using adaptive regularization,” Machine learning, vol. 90, no. 3, pp. 347–383, 2013.
- [95] J. Vermorel and M. Mohri, “Multi-armed bandit algorithms and empirical evaluation,” in European conference on machine learning. Springer, 2005, pp. 437–448.

- [96] T. L. Lai, P. W. Lavori, and K. W. Tsang, “Adaptive design of confirmatory trials: Advances and challenges,” Contemporary clinical trials, vol. 45, pp. 93–102, 2015.
- [97] M. Dudik, D. Hsu, S. Kale, N. Karampatziakis, J. Langford, L. Reyzin, and T. Zhang, “Efficient optimal learning for contextual bandits,” arXiv preprint arXiv:1106.2369, 2011.
- [98] Q. Zhou, X. Zhang, J. Xu, and B. Liang, “Large-scale bandit approaches for recommender systems,” in Neural Information Processing: 24th International Conference, ICONIP 2017, Guangzhou, China, November 14-18, 2017, Proceedings, Part I 24. Springer, 2017, pp. 811–821.
- [99] D. Bouneffouf, I. Rish, and G. A. Cecchi, “Bandit models of human behavior: Reward processing in mental disorders,” in Artificial General Intelligence: 10th International Conference, AGI 2017, Melbourne, VIC, Australia, August 15-18, 2017, Proceedings 10. Springer, 2017, pp. 237–248.
- [100] S. M. Kakade, S. Shalev-Shwartz, and A. Tewari, “Efficient bandit algorithms for online multiclass prediction,” in Proceedings of the 25th international conference on Machine learning, 2008, pp. 440–447.
- [101] D. Chen, X. Cao, L. Wang, F. Wen, and J. Sun, “Bayesian face revisited: A joint formulation,” in Computer Vision–ECCV 2012: 12th European Conference on Computer Vision, Florence, Italy, October 7-13, 2012, Proceedings, Part III 12. Springer, 2012, pp. 566–579.
- [102] T. L. Lai and H. Robbins, “Asymptotically efficient adaptive allocation rules,” Advances in applied mathematics, vol. 6, no. 1, pp. 4–22, 1985.
- [103] J. Langford and T. Zhang, “The epoch-greedy algorithm for multi-armed bandits with side information,” Advances in neural information processing systems, vol. 20, 2007.
- [104] M. Sugeno and G. Kang, “Structure identification of fuzzy model,” Fuzzy sets and systems, vol. 28, no. 1, pp. 15–33, 1988.
- [105] L. Li, W. Chu, J. Langford, and R. E. Schapire, “A contextual-bandit approach to personalized news article recommendation,” in Proceedings of the 19th international conference on World wide web, 2010, pp. 661–670.

References

- [106] M. Tokic, “Adaptive ϵ -greedy exploration in reinforcement learning based on value differences,” in Annual conference on artificial intelligence. Springer, 2010, pp. 203–210.
- [107] T. Takagi and M. Sugeno, “Fuzzy identification of systems and its applications to modeling and control,” IEEE transactions on systems, man, and cybernetics, no. 1, pp. 116–132, 1985.
- [108] J.-S. Jang, “Anfis: adaptive-network-based fuzzy inference system,” IEEE transactions on systems, man, and cybernetics, vol. 23, no. 3, pp. 665–685, 1993.
- [109] L. Li, W. Chu, J. Langford, and R. E. Schapire, “A contextual-bandit approach to personalized news article recommendation,” in Proceedings of the 19th international conference on World wide web, 2010, pp. 661–670.
- [110] M. Sugeno and G. Kang, “Structure identification of fuzzy model,” Fuzzy sets and systems, vol. 28, no. 1, pp. 15–33, 1988.
- [111] J.-S. Jang, “Anfis: adaptive-network-based fuzzy inference system,” IEEE transactions on systems, man, and cybernetics, vol. 23, no. 3, pp. 665–685, 1993.
- [112] T. Takagi and M. Sugeno, “Fuzzy identification of systems and its applications to modeling and control,” IEEE transactions on systems, man, and cybernetics, no. 1, pp. 116–132, 1985.
- [113] D. Dua, C. Graff et al., “Uci machine learning repository, 2017,” URL <http://archive.ics.uci.edu/ml>, vol. 7, no. 1, p. 62, 2017.
- [114] T. Kumari, R. Sharma, and P. Bedi, “A contextual-bandit approach for multifaceted reciprocal recommendations in online dating,” Journal of Intelligent Information Systems, vol. 59, no. 3, pp. 705–731, 2022.
- [115] B. Efron, The jackknife, the bootstrap and other resampling plans. SIAM, 1982.
- [116] J. Hernandez-Lobato, Y. Li, M. Rowland, T. Bui, D. Hernández-Lobato, and R. Turner, “Black-box alpha divergence minimization,” in International conference on machine learning. PMLR, 2016, pp. 1511–1520.

- [117] M. Plappert, R. Houthoofd, P. Dhariwal, S. Sidor, R. Y. Chen, X. Chen, T. Asfour, P. Abbeel, and M. Andrychowicz, “Parameter space noise for exploration,” arXiv preprint arXiv:1706.01905, 2017.
- [118] Z. Cai, G. Yuan, S. Qiao, S. Qu, Y. Zhang, and R. Bing, “Fg-cf: Friends-aware graph collaborative filtering for poi recommendation,” Neurocomputing, vol. 488, pp. 107–119, 2022.
- [119] Y. Koren, R. Bell, and C. Volinsky, “Matrix factorization techniques for recommender systems,” Computer, vol. 42, no. 8, pp. 30–37, 2009.
- [120] S. Feng, X. Li, Y. Zeng, G. Cong, Y. M. Chee, and Q. Yuan, “Personalized ranking metric embedding for next new poi recommendation,” in Proceedings of the 24th International Conference on Artificial Intelligence, ser. IJCAI’15. AAAI Press, 2015, p. 2069–2075.
- [121] J. Bobadilla, F. Ortega, A. Hernando, and J. Bernal, “A collaborative filtering approach to mitigate the new user cold start problem,” Knowledge-based systems, vol. 26, pp. 225–238, 2012.
- [122] Y. Liu, C. Liu, B. Liu, M. Qu, and H. Xiong, “Unified point-of-interest recommendation with temporal interval assessment,” in Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining, 2016, pp. 1015–1024.
- [123] D. Yang, B. Fankhauser, P. Rosso, and P. Cudre-Mauroux, “Location prediction over sparse user mobility traces using rnns,” in Proceedings of the twenty-ninth international joint conference on artificial intelligence, 2020, pp. 2184–2190.
- [124] T. Liu, J. Liao, Z. Wu, Y. Wang, and J. Wang, “A geographical-temporal awareness hierarchical attention network for next point-of-interest recommendation,” in Proceedings of the 2019 international conference on multimedia retrieval, 2019, pp. 7–15.
- [125] M. Chen, W.-Z. Li, L. Qian, S.-L. Lu, and D.-X. Chen, “Next poi recommendation based on location interest mining with recurrent neural networks,” Journal of Computer Science and Technology, vol. 35, pp. 603–616, 2020.

References

- [126] Y. Zhu, H. Li, Y. Liao, B. Wang, Z. Guan, H. Liu, and D. Cai, “What to do next: Modeling user behaviors by time-lstm.” in IJCAI, vol. 17, 2017, pp. 3602–3608.
- [127] J. Li, W. Zeng, D. Liu, Z. Yang, and W. Cong, “Spatial-temporal long short-term memory networks for airport flight delay prediction,” in Proceedings of the 19th World Transport Convention, 2019.
- [128] P. Zhao, A. Luo, Y. Liu, J. Xu, Z. Li, F. Zhuang, V. S. Sheng, and X. Zhou, “Where to go next: A spatio-temporal gated network for next poi recommendation,” IEEE Transactions on Knowledge and Data Engineering, vol. 34, no. 5, pp. 2512–2524, 2020.
- [129] Y. Liu and A.-b. Wu, “Poi recommendation method using deep learning in location-based social networks,” Wireless Communications and Mobile Computing, vol. 2021, no. 1, p. 9120864, 2021.
- [130] Q. Liu, S. Wu, L. Wang, and T. Tan, “Predicting the next location: A recurrent model with spatial and temporal contexts,” in Proceedings of the AAAI conference on artificial intelligence, vol. 30, no. 1, 2016.
- [131] Y. Wu, K. Li, G. Zhao, and X. Qian, “Long-and short-term preference learning for next poi recommendation,” in Proceedings of the 28th ACM international conference on information and knowledge management, 2019, pp. 2301–2304.
- [132] R. Li, Y. Shen, and Y. Zhu, “Next point-of-interest recommendation with temporal and multi-level context attention,” in 2018 IEEE International Conference on Data Mining (ICDM). IEEE, 2018, pp. 1110–1115.
- [133] J.-D. Zhang, C.-Y. Chow, and Y. Li, “Lore: Exploiting sequential influence for location recommendations,” in Proceedings of the 22nd ACM SIGSPATIAL international conference on advances in geographic information systems, 2014, pp. 103–112.
- [134] J.-D. Zhang and C.-Y. Chow, “Geosoca: Exploiting geographical, social and categorical correlations for point-of-interest recommendations,” in Proceedings of the 38th international ACM SIGIR conference on research and development in information retrieval, 2015, pp. 443–452.

- [135] R. Baral and T. Li, “Exploiting the roles of aspects in personalized poi recommender systems,” Data Mining and Knowledge Discovery, vol. 32, pp. 320–343, 2018.
- [136] B. Liu and H. Xiong, “Point-of-interest recommendation in location based social networks with topic and location awareness,” in Proceedings of the 2013 SIAM international conference on data mining. SIAM, 2013, pp. 396–404.
- [137] R. Baral, D. Wang, T. Li, and S.-C. Chen, “Geotecs: exploiting geographical, temporal, categorical and social aspects for personalized poi recommendation,” in 2016 IEEE 17th International Conference on Information Reuse and Integration (IRI). IEEE, 2016, pp. 94–101.
- [138] H. Yin, Y. Sun, B. Cui, Z. Hu, and L. Chen, “Lcars: a location-content-aware recommender system,” in Proceedings of the 19th ACM SIGKDD international conference on Knowledge discovery and data mining, 2013, pp. 221–229.
- [139] C. Cheng, H. Yang, I. King, and M. R. Lyu, “A unified point-of-interest recommendation framework in location-based social networks,” ACM Transactions on Intelligent Systems and Technology (TIST), vol. 8, no. 1, pp. 1–21, 2016.
- [140] D. Yang, D. Zhang, V. W. Zheng, and Z. Yu, “Modeling user activity preference by leveraging user spatial temporal characteristics in lbsns,” IEEE Transactions on Systems, Man, and Cybernetics: Systems, vol. 45, no. 1, pp. 129–142, 2014.
- [141] C. Cheng, H. Yang, I. King, and M. Lyu, “Fused matrix factorization with geographical and social influence in location-based social networks,” in Proceedings of the AAAI conference on artificial intelligence, vol. 26, no. 1, 2012, pp. 17–23.

PAPER NAME

nipun_thesis.pdf

AUTHOR

Nipun Bansal

WORD COUNT

37666 Words

CHARACTER COUNT

206647 Characters

PAGE COUNT

137 Pages

FILE SIZE

8.8MB

SUBMISSION DATE

Aug 27, 2024 1:03 AM GMT+5:30

REPORT DATE

Aug 27, 2024 1:05 AM GMT+5:30

● 4% Overall Similarity

The combined total of all matches, including overlapping sources, for each database.

- 2% Internet database
- 2% Publications database
- Crossref database
- Crossref Posted Content database
- 2% Submitted Works database

● Excluded from Similarity Report

- Bibliographic material
- Quoted material
- Cited material
- Small Matches (Less than 10 words)
- Manually excluded text blocks