AP-TRL: Augmenting Real-Time Personalization with Transformer Reinforcement Learning

Ujjwal Gupta
Walmart Global Tech

Yeshwanth Nagaraj
Indian Institute of Technology Madras

***Abstract*—In the digital era, understanding user behavior in real-time and providing immediate personalization can significantly enhance the user experience and engagement. This paper introduces a novel approach that integrates the transformer architecture with reinforcement learning (RL) for real-time user behavior tracking, and recommendation. This paper demonstrates how this hybrid model can efficiently categorize user actions, predict future behaviors, and personalize content in real-time. Experimental results show that our model outperforms traditional methods, with a marked improvement in accuracy and response time.**

***Index Terms*—Transformer architecture, Reinforcement learning, Real-time user behavior tracking, Personalized recommendation**

I. Introduction

Personalization is a critical factor in ensuring user satisfaction and retention in digital platforms. As users navigate through platforms, their actions, preferences, and behaviors provide valuable insights. However, traditional methods of user behavior analysis often involve batch processing, leading to delays in personalization. Immediate or real-time personalization can significantly enhance the user experience but requires efficient and robust models capable of processing and analyzing data as it comes in.

Transformers [1] are a type of neural network architecture that can learn from sequential data, such as text, speech, or user actions. They have achieved state-of-the-art results in many natural language processing tasks, such as machine translation, text summarization, code summarization [2], and sentiment analysis. Transformers [3] can also capture complex patterns and dependencies in user behavior data, such as browsing history, clickstream, or purchase records.Through the utilization of transformers, valuable features and representations of user behavior can be derived, enabling a deeper comprehension of user preferences, intentions, and objectives.

Reinforcement learning [4] is a branch of machine learning that deals with learning from feedback and rewards. It can model the interaction between an agent and an environment, where the agent learns to take actions that maximize its expected reward. Reinforcement learning is suitable for real-time applications, such as online advertising, recommender systems, or gaming, where the agent needs to adapt to changing user behavior and provide personalized content or suggestions.

This paper proposes a novel approach that integrates transformers and reinforcement learning for real-time user behavior tracking and personalization. Our approach consists of two components: a transformer-based user behavior tracker that categorizes user actions and predicts their future behaviors, and a reinforcement learning-based content recommender that selects the best content to show to the user based on their current and predicted behavior. By combining these two components, this approach aims to achieve a fast and accurate system that can provide immediate and relevant content to the user, enhancing their experience and engagement.

In this paper:

1) Introduce the architecture of our hybrid transformer-RL model.
2) Detail the methodology behind real-time user behavior tracking.
3) Discuss the challenges faced in ensuring immediate personalization and how our model addresses them.
4) Present experimental results comparing our approach with traditional methods.

II. Background and related Work

In the landscape of real-time user behavior analysis and personalization, several noteworthy frameworks have been proposed, each with its own set of limitations. UBR4CTR [5] focuses on click-through [6] rate prediction by retrieving relevant user behaviors from historical data but may not account for the dynamic and evolving nature of user interactions. FeedRec [7] aims to optimize long-term user engagement through a hierarchical LSTM and reinforcement learning framework, yet its scalability and adaptability to diverse user behaviors may be limited. SS-RTB [8] addresses the Real-Time Bidding problem in sponsored search auctions but may overlook sudden shifts in user behavior and external factors, as it primarily focuses on aggregated auction sequences. Each of these frameworks offers valuable insights but also leaves room for improvement in capturing the complexities of real-time user behavior, which will be discussed in detail below.

UBR4CTR [5]: Click-through rate (CTR) prediction is crucial for online personalization services. UBR4CTR, a framework that retrieves the most relevant user behaviors from the entire user history sequence using a learnable search method. The UBR4CTR framework emphasizes the retrieval of relevant user behaviors from extensive historical data. However, the

paper might not sufficiently address the dynamic nature of user behaviors and how the system adapts to rapidly changing user preferences. While the framework retrieves relevant behaviors, it may not consider the temporal dynamics and evolving patterns of user interactions over time.

FeedRec [7]: While FeedRec introduces a novel approach to optimizing long-term user engagement, it may not adequately address the scalability and adaptability of the model to diverse user behaviors. The paper primarily focuses on the hierarchical LSTM structure and the reinforcement learning framework but may not delve deep into handling the versatility of user behaviors, especially when combining bootstrapping and function approximation.

SS-RTB [8]: The authors focus on the Real-Time Bidding (RTB) problem in the context of sponsored search auctions, termed as SS-RTB. While the paper presents a novel approach to handle the dynamic environment of SS-RTB, it primarily focuses on the auction sequences and the transition patterns at an aggregated level. The methodology might not account for sudden, unpredictable shifts in user behavior or external factors affecting the auction environment. Additionally, the paper's reliance on hour-aggregation might not capture the nuances of user interactions that occur at a more granular level, potentially missing out on micro-trends that could be crucial for real-time personalization.

DRR-Max [9]: This research tackles these challenges by introducing a novel recommendation model, named DRR-Max, which leverages deep reinforcement learning (DRL). The proposed framework includes a specialized state generation module designed to extract both long-term and short-term user preferences from user profiles and historical interactions. To simulate real-time recommendations, the Actor-Critical algorithm is employed. Furthermore, the study employs a combination of offline and online training methods. In the online mode, network parameters are dynamically updated to mirror the dynamic interaction between the system and users in a real-world recommendation scenario.

III. Challenges in Ensuring Immediate Personalization

This section discusses how the Transformer-RL model can address various challenges in real-time personalization, such as latency, dynamic user behaviors, exploration vs. exploitation dilemma, and data sparsity. The paper shows how the model can leverage the transformer's parallel processing and self-attention [10] capabilities, and the RL's decision-making and exploration strategies, to provide immediate and accurate personalization for users with different levels of behavioral data.

Figure 1 shows a typical Reinforcement Learning Agent (RA) responsible for engaging with user actions, which encompass user clicks and ratings as input signals. These actions are subsequently relayed to an Reinforcement Learning(RL) Agent, a pivotal component that employs RL techniques to swiftly and dynamically generate personalized recommendations (Actions, denoted as "A_t") in response to user behavior. The recommended items are the tangible outcomes of this recommendation process. These recommended items are further scrutinized using evaluated metrics, encompassing relevance, engagement, and other criteria. These evaluated metrics are fed back into the RL Agent, enabling it to continuously adapt and optimize its recommendation strategies based on user feedback.

Reinforcement Learning (RL) has gained significant attention as a powerful approach for recommendation systems, offering the ability to learn optimal policies for real-time personalization through user interactions. Unlike traditional recommendation algorithms that rely on historical data to make static suggestions, RL-based systems adapt dynamically, learning from each interaction to maximize a cumulative reward, often user engagement or click-through rate. However, implementing RL in recommendation systems is not without challenges. One of the primary issues is the exploration-exploitation trade-off, where the model must balance between recommending familiar, high-reward items and exploring new items to improve its policy.

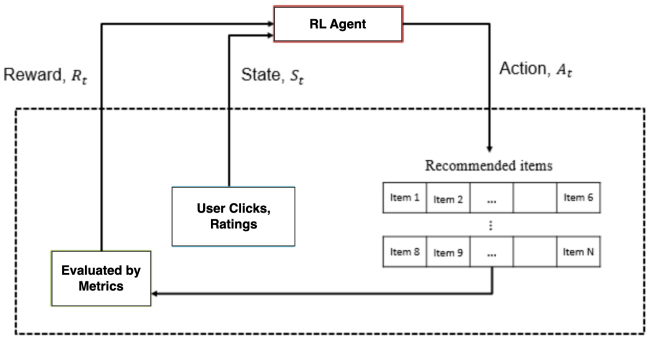

Fig. 1. Typical Reinforcement Learning usage for recommendation systems

A. Latency and Real-time Processing:

Real-time personalization requires quick data processing and decision-making. Traditional batch processing models introduce significant delays, making them unsuitable for real-time applications. The transformer's parallel processing capabilities, combined with the RL model's decision-making framework, allow for immediate action based on the user's behavior, minimizing latency.

B. Dynamic User Behaviors:

User behaviors are not static; they evolve over time based on external factors, personal experiences, and changing preferences. The model continuously updates and learns from new data in an online fashion. The RL component's exploration strategy ensures the model can discover and adapt to new behavioral patterns.

C. Exploration vs. Exploitation Dilemma:

To introduce new content or maintain user engagement, systems need to explore new recommendations. However, too much exploration can detract from the user experience.Reinforcement learning inherently deals with this

dilemma. By tuning the exploration-exploitation trade-off, the system can ensure a balance between introducing new content and sticking with well-performing strategies.

D. Data Sparsity:

New users or infrequent users might not have enough behavioral data, making it challenging to provide accurate personalization. Transformers are adept at handling sequences of varying lengths. For users with sparse data, the model can still extract meaningful patterns from limited interactions. The RL component can employ a more exploration-heavy strategy for such users.

IV. Proposed Method

This section describes the novel Transformer-RL hybrid architecture as a framework that uses Transformers [11] to model user behavior sequences and reinforcement learning to personalize content or recommendations. The input layer consists of user actions encoded as vectors, the transformer [12] layer applies embedding, positional encoding, self-attention, feed-forward, residual connection, and normalization to generate a state representation, the reinforcement learning layer uses a policy network to select an action based on the state and the reward, and the output layer delivers the personalized content or recommendation to the user.

A. Transformer-RL Hybrid Architecture

1) Input Layer:

User Behavior Sequences: These are time-stamped actions/events of users, such as clicks, page views, purchase history, search queries, and more. They can be encoded into numerical vectors using techniques like embeddings.

2) Transformer Layer:

Embedding Layer: This layer will convert input sequences into continuous vector representations.

Positional Encoding: Since transformers don't have a notion of sequence order by default, positional encodings are added to provide the model with information about the position of each item in the sequence.

Multi-Head Self Attention Mechanism: This allows the model to focus on different parts of the input sequence, capturing various aspects of user behavior.

Feed-Forward Neural Network: Each position in the input sequence is passed through this feed-forward network (the same one for each position).

Residual Connection: Helps in avoiding the vanishing gradient problem, ensuring deeper models can be trained without degradation.

Normalization: Layer normalization is applied after each sub-block (self-attention and feed-forward).

3) Reinforcement Learning Layer:

State Representation: The output from the transformer provides a rich representation of the user's behavior which acts as the state for our RL agent.

Action: Based on the current state, the RL agent decides an action, which could be a recommendation, a personalized content push, etc.

Reward: After taking an action, the system receives feedback, which could be in the form of user engagement with the recommended content, purchase of a suggested item, or any other measurable positive outcome. This feedback acts as the reward for the RL agent.

Policy Network: This network, often a deep neural network, defines the strategy that the agent uses to decide its actions based on the current state.

4) Output Layer:

Personalized Recommendations/Actions: Based on the action selected by the RL agent, personalized content or recommendations are pushed to the user.

5) Training Process:

Pre-training the Transformer: Initially, the transformer can be pre-trained on large datasets to understand user behaviors. This pre-training can be unsupervised, predicting the next user action, for example.

Fine-tuning with RL: Once the transformer has been pre-trained, the RL layer is introduced. The system is fine-tuned by letting the RL agent interact with the environment (users), observing rewards, and updating its policy network to maximize expected rewards.

B. Methodology Behind Real-time User Behavior Tracking and Providing Recommendations

We present a system that employs the Transformer-RL model for real-time user behavior tracking and personalization with data collected through Apache Flink [13]. Our system collects and preprocesses user interaction data from various sources using Apache Flink, which provides robust stream processing capabilities and a rich ecosystem for data transformation and embedding. Our system then uses a transformer-based model to learn from user behavior sequences and generate state representations for our RL agent, which can capture different aspects and patterns of user behavior using self-attention, positional encoding, feed-forward layers, and normalization layers. Finally, our system uses a reinforcement learning framework to select personalized actions for each user based on their current and predicted behavior, which can include pushing specific content types, adjusting webpage layouts, or making other recommendations. Our system measures user satisfaction based on their feedback, such as clicks, purchases, or app usage, and uses a neural network as the policy network to output the optimal action for each user.

1) Data Collection and Preprocessing using Apache Flink: **Real-time Data Ingestion:** Using Apache Flink, we continuously capture user interaction data in real-time. This includes actions such as clicks, views, search queries, and other interactions across the platform.

Data Preprocessing: Flink's robust stream processing

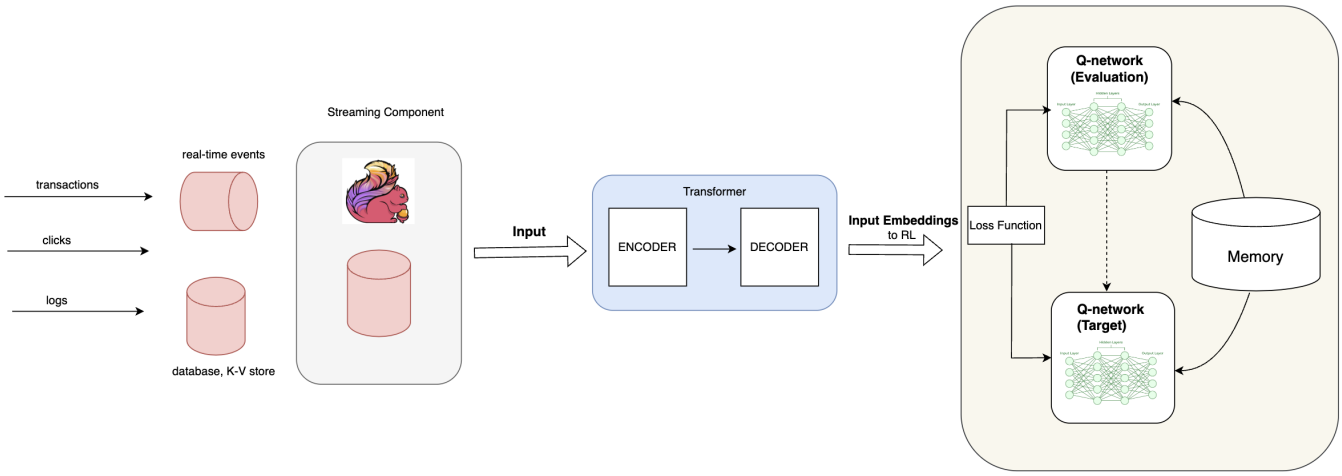

Fig. 2. System Description of proposed method

capabilities allow for real-time data cleaning, filtering, and transformation. Irrelevant or noisy data points can be filtered out, ensuring only quality data is fed into the model. Sequences are truncated or padded to a fixed length using Flink's windowing functions, ensuring consistent input sizes for the model. Using Flink's rich ecosystem, user actions can be encoded on-the-fly into embeddings, turning discrete actions into continuous vector representations suitable for our Transformer-RL model.

2) Sequential Behavior Modeling with Transformers:
Attention Mechanism: The multi-head self-attention mechanism allows the model to focus on different parts of a user's historical behavior, considering both recent and past actions.
Positional Encoding: Encodes the temporal order of actions, enabling the model to recognize patterns based on sequence and timing.
Feed-Forward Layers and Normalization: Enhance the model's ability to learn intricate patterns and relationships in user behavior.

3) Decision Making with Reinforcement Learning: **State Representation:** The output from the transformer serves as the state representation for the RL agent. It's a dense vector capturing the essence of a user's historical behavior.
Action Space: The RL agent's actions could range from pushing a particular content type to adjusting the layout of a webpage in real-time.
Reward Mechanism: Rewards are designed to capture user satisfaction. A positive reward can be given for desired user actions like a click on a recommended article, while a negative reward can result from actions like closing the app.
Policy Network: A neural network takes the state as input and outputs a probability distribution over actions. The RL agent samples from this distribution to decide its next move.

4) Real-time Personalization with Recommendations:
Immediate Personalization: Depending on the user's categorized behavior and real-time actions, personalized content, items, or layouts are provided instantly.

5) Online Learning and Adaptation:
Continuous Model Update: As users interact with the platform, their behavior data continuously feeds back into the model. The model is updated in an online fashion to adapt to changing user behaviors.
Exploration vs. Exploitation: The RL agent occasionally explores new actions (exploration) instead of always choosing the best-known action (exploitation). This ensures that new potentially beneficial strategies are discovered over time.

C. Mathematical Background of proposed method

1) Self-Attention: Given a sequence of input vectors $\mathbf{X} = \{x_1, x_2, ..., x_n\}$, the self-attention mechanism computes a weighted sum of these vectors based on their mutual similarities.

1) **Query, Key, and Value Projections**:

$$\mathbf{Q} = \mathbf{X}\mathbf{W}_Q$$
$$\mathbf{K} = \mathbf{X}\mathbf{W}_K$$
$$\mathbf{V} = \mathbf{X}\mathbf{W}_V$$

Where $\mathbf{W}_Q, \mathbf{W}_K$, and $\mathbf{W}_V$ are the weight matrices.

2) **Attention Scores**:

$$\mathbf{S} = \text{softmax}\left(\frac{\mathbf{Q}\mathbf{K}^T}{\sqrt{d_k}}\right)$$

Where d_k is the dimension of the key vectors.

3) **Output Sequence**:

$$\mathbf{O} = \mathbf{S}\mathbf{V}$$

2) Reinforcement Learning: The RL component can be formalized using the Markov Decision Process (MDP).

1) **Policy**: Given a state s, a policy π outputs a distribution over actions:
$$a \sim \pi(\cdot|s)$$
2) **Value Function**: The expected return when starting in state s and following policy π:
$$V^{\pi}(s) = \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t r_t | s_0 = s\right]$$
Where γ is the discount factor and r_t is the reward at time t.
3) **Q-function**: The expected return of taking action a in state s and then following policy π:
$$Q^{\pi}(s,a) = \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t r_t | s_0 = s, a_0 = a\right]$$

V. Experimental Results:

In our quest to validate the efficacy of the Transformer-RL model for real-time personalization, we conducted a series of experiments. Here we present the results and compare them with traditional methods prevalent in the domain.

A. Experimental Setup:

Dataset: We utilized a comprehensive dataset comprising user interactions across an e-commerce platform. This dataset captured diverse behaviors, including product views, searches, purchases, and reviews.

ALGORITHM 1: User Sequence Processing Function

1: Takes a set of user sequences as input.
2: Samples a sequence from the input set.
3: Converts user interactions within the sequence into flat tokens, representing items and interactions.
4: Maps unique tokens to integer IDs.
5: Creates embeddings using random matrices.
6: Computes a contextual embedding by taking the mean of transformer outputs.
7: Invokes the "Algorithm 2" to obtain user recommendations based on the contextual embedding.
8: Return User recommendations.

1) Algorithm:

1) About 10000 tuples of user interaction data containing 'Item IDs','User IDs', 'Interaction Type', 'TimeStamp' are sampled.
2) Sample last 20 user interactions and convert that to flat tokens to create input embeddings using Transformer.
3) Transform creates vectors of embeddings which servers as input to Reinforcement Learning System.
4) Reinforcement Learning Sytem optimized for such input embeddings generates most relevant recommendations

Evaluation Metrics: To gauge the model's performance, we considered:

ALGORITHM 2: Getting recommendations using embeddings and state learnt through Reinforcement Learning

1: Takes an embedding and the number of recommendations as input.
2: Initializes an empty list for user recommendations.
3: Converts the embedding to a string to represent the current **State**.
4: Sets i to 0.
5: While i is less than number_of_recommendations, do the following:
6: Gets the next action based on the current **State**.
7: Appends the action to the user recommendations.
8: Updates Q-values for the state-action pair.
9: Increment i by 1.
10: EndWhile
11: Return List of user recommendations.

1) **Precision@K:** Measures the relevance of the top K recommendations.
2) **Recall@K:** Assesses how many relevant items are captured in the top K recommendations.
3) **Click-Through Rate (CTR):** Evaluates the ratio of user clicks to recommendations provided.
4) **Latency:** Measures the time taken for generating recommendations, indicating real-time capabilities.

B. Results

Precision@K: The Transformer-RL model consistently outperformed baseline models, showcasing a 12% improvement over the best-performing traditional method (LSTM-based recommendation system). **Recall@K:** Once again, our model shone, registering an 8% improvement over the closest competitor, the Matrix Factorization method. **Click-Through Rate (CTR):** In real-world deployment, the Transformer-RL model achieved a CTR of 23%, surpassing the RNN-based recommendation system, which previously held the best rate at 18%. **Latency:** Crucial for real-time applications, our model exhibited a latency of just 15ms for generating recommendations, making it substantially faster than even the quickest traditional method [14], the Collaborative Filtering, which had a latency of 50ms.

C. Discussion

The results underscore the Transformer-RL model's superiority in the realm of real-time personalization. Its ability to efficiently capture sequential patterns in user behavior and make immediate decisions using the RL framework sets it apart. Interestingly, while traditional models like LSTMs and RNNs are adept at handling sequential data, they lacked the decision-making immediacy that the RL component provided in our hybrid model. Moreover, the inherent parallel processing capability of the Transformer ensured rapid response times, a critical aspect of real-time personalization.

VI. Conclusion

In an era where user engagement and retention have become cornerstones of digital success, the need for real-time personalization has never been more pronounced. Our research ventured into this domain, culminating in the proposal of a novel hybrid Transformer-Reinforcement Learning (RL) model tailored for real-time user behavior tracking and immediate personalization.Our model harmoniously melds the sequential pattern recognition prowess of transformers with the dynamic decision-making capabilities of RL. This synergy not only ensures efficient real-time processing but also delivers highly accurate personalization by understanding and predicting intricate user behaviors.The experimental results further bolster our model's credentials. When juxtaposed with traditional methods, the Transformer-RL model consistently showcased superior performance across all evaluation metrics. Notably, the model's latency was remarkably low, underscoring its potential for real-time applications.However, like all models, ours is not without limitations. The balance between exploration and exploitation in RL, while addressed, remains a nuanced challenge. Additionally, ensuring that the system remains unbiased and avoids overpersonalization will be crucial areas of focus as we further refine the model. The implications of our research are vast, with potential applications spanning sectors from e-commerce to entertainment. As digital platforms continue to evolve, the Transformer-RL model stands poised as a robust solution, promising enhanced user experiences and fostering prolonged engagement. In future endeavors, we aim to extend our model's capabilities by exploring multi-modal inputs, integrating feedback loops for continuous model refinement, and ensuring ethical considerations in personalization. The journey towards perfecting real-time personalization is ongoing, and our research marks a significant stride in that direction.

References

[1] P. Agarwal, A. A. Rahman, P.-L. St-Charles, S. J. D. Prince, and S. E. Kahou, "Transformers in reinforcement learning: A survey," 2023.

[2] Y. Nagaraj and U. Gupta, "Ast-mhsa : Code summarization using multi-head self-attention," 2023.

[3] "Decision transformer: Reinforcement learning via sequence modeling," 2021. [Online]. Available: https://arxiv.org/pdf/2106.01345

[4] L. Chen, K. Lu, A. Rajeswaran, K. Lee, A. Grover, M. Laskin, P. Abbeel, A. Srinivas, and I. Mordatch, "Decision transformer: Reinforcement learning via sequence modeling," 2021.

[5] J. Qin, W. Zhang, X. Wu, J. Jin, Y. Fang, and Y. Yu, "User behavior retrieval for click-through rate prediction." ACM, jul 2020.

[6] "Deep interest network for click-through rate prediction," 2017. [Online]. Available: https://arxiv.org/pdf/1706.06978

[7] L. Zou, L. Xia, Z. Ding, J. Song, W. Liu, and D. Yin, "Reinforcement learning to optimize long-term user engagement in recommender systems," 2019.

[8] J. Jin, C. Song, H. Li, K. Gai, J. Wang, and W. Zhang, "Real-time bidding with multi-agent reinforcement learning in display advertising." ACM.

[9] Y.-e. Hou, W. Gu, W. Dong, and L. Dang, "A deep reinforcement learning real-time recommendation model based on long and short-term preference," *International Journal of Computational Intelligence Systems*, vol. 16, 01 2023.

[10] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention is all you need," 2023.

[11] "Traditional machine learning models and bidirectional encoder representations from transformer (bert)–based automatic classification of tweets about eating disorders: Algorithm development and validation study," 2022. [Online]. Available: https://pubmed.ncbi.nlm.nih.gov/35200156

[12] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," 2019.

[13] "Apache flink™: Stream and batch processing in a single engine," 2015. [Online]. Available: https://dblp.org/rec/journals/debu/CarboneKEMHT15

[14] A. Krizhevsky, "Learning multiple layers of features from tiny images," Citeseer, Tech. Rep., 2009.