Tuijin Jishu/Journal of Propulsion Technology
ISSN: 1001-4055
Vol. 44 No. 5 (2023)

Conservative Learning Method for Diabetic Issues Using Reinforcement Learning & Brain Strom Optimization

S.anusuya[1], Dr. K. Anandapadmanabhan[2]

RESEARCH SCHOLAR[1], ASSOCIATE PROFESSOR[2]
anuciaa@gmail.com[1], kapn0305@gmail.com[2]
SRI VASAVI COLLEGE, SELF FINANCE WING, ERODE.

Abstract: Promising outcomes are shown when attention-based sequential recommendation techniques effectively capture the dynamic interests of users in previous interactions. Recent work has started incorporating reinforcement learning (RL) into these models in addition to improving user representations. We offer a recommender system (RS) that combines direct user feedback into the compensating format and takes into account significant features to deliver a more personalized experience, enabling development, by building up consecutive suggestions for RL problems with compensating signals. Recent RS reinforcement learning systems integrate Supervised Negative Q Learning (SNQN) and Supervised Superior Actor Critic (SA2C), yet these issues still exist. For instance, because there aren't many positive compensation signals, Q-value estimates are typically biased toward negative values.Additionally, the precise timestamp of the sequence has a significant influence on the Q value. We suggest using contrast-based objectives using Brain Strom optimization in conjunction with extensions to solve datasets with larger ranges in order to resolve the aforementioned problems. Furthermore, we are aware that using negative sampling could lead to possible instability problems. As a result, we present an enhanced strategy to address these issues with greater effectiveness.

Keywords: *Reinforcement Learning (RL), recommender systems (RS), Supervised Negative Q-Learning (SNQN), Supervised Advantage Actor-Critic (SA2C), Brain Strom Optimization.*

1. Introduction

Recommendation systems (RS) are now a crucial tool in e-commerce, social networking, and journalism for introducing products and offering customers personalized content. User interactions with projects usually happen in a step-by-step fashion. These interactions' order and timing are crucial [1]. Recurrent neural network (RNN) based models outperform first-generation sequential recommendation systems by a large margin. But in some cases, such as when dealing with language or time series data, these left-to-right processing models are not the greatest at capturing the nuances of user activity sequences that do not follow a strict order.In order to increase suggestion accuracy, recent models like as Transformer take into account bidirectional dependencies and better grasp the user's preferred operation order. It was discovered that large-scale language models that are physically comparable to RS transformers, such GPT-4, work well in zero-shot item recommendation processes [2].

Proposed for reinforcement learning in RS, self-guided reinforcement learning [3] demonstrated encouraging outcomes on offline evaluation criteria. Self-supervised Actor-Critic (SAC) and Self-supervised Q-learning (SQN) are the two learning frameworks that have been presented. Using reinforcement learning components as a kind of normalization to improve recommendation models for specified compensations is the fundamental idea behind self-supervised reinforcement learning. E-commerce domains, for instance, offer recommendations that increase sales rather than decrease them. Press [4]. Despite their high performance, SQN and SAC still have certain drawbacks. SAC weighs the participant (supervised portion) using the output Q-value 2 as a critic. The Q-value is heavily reliant on the sequence's unique timestamp, which adds more bias to the learning process [5].

Map Speech Q-Learning (SNQN) is the name we give it. Imitation learning in a scenario of scarcity compensation provides an additional rationale for negative sampling in reinforcement learning [6]. In this configuration, the RL component of SNQN functions as a strong ranking model that may be utilized to produce recommendations in addition to being a standardization tool.The "benefits" of positive activities in comparison to other behaviors can be computed using the sampled negative behaviors and Q-value estimates [7]. Here, the supervised output layer's weights are added using Supervised Advantage Actor-Critic (SA2C) using this advantage in place of the initial Q-value. When estimating Q-values, the benefit value can be considered a normalized Q-value that helps minimize sequence timestamp bias. This contributes in the following ways:

- We investigate the use of sequential reinforcement schemes and contrastive learning objectives, and we present empirical results on a range of challenging real-world datasets demonstrating their efficacy.
- In line with earlier research, we identified the fundamental problems brought about by the use of negative work sampling and created Brain to lessen these instabilities when RL training happened. We advise more cautious objectives and Strom optimization.

In order to identify instabilities that could impair model performance in online deployments, our analysis emphasizes the necessity of keeping an eye on the training process of reinforcement learning-based models. When using reinforcement learning, we recommend documenting training progress and tabulating outcomes.

2. Literature Survey

Itinerant Suggestions The goal of sequential suggestions is to ascertain users' preferences from historical behavior. Markov chain models and latent representation techniques have received the majority of attention in the past. Convolutional neural networks, recurrent neural networks, and graph neural networks have gained popularity and become strong foundational models for recommendation systems with the advent of deep learning. Transformer models have been coupled with RL in sequence recommendation tasks because of their performance in sequence modeling challenges across multiple domains. SASRec modifies the converter based on the next forecast made by the recommendation system.By applying weights to various items in a user's past, the transformer architecture employed in this assignment is able to determine which elements are most pertinent to the user's current state of affairs. [8] Make better, more personalized recommendations by utilizing BERT. Bidirectional encoder representations of sensors are incorporated into BERT4Rec [9], which consider that sequential suggestions could not strictly follow the ordering requirements of the language model.

Comparative Education for Suggestion By bringing comparable occurrences closer to the representation space and dissimilar instances farther away, contrastive learning seeks to learn data representations. Although contrastive learning has been extensively researched and demonstrated outstanding performance in computer vision [11] and natural language processing [12], it has not been thoroughly investigated in recommender systems. Contrastive learning objectives are included into the SASRec framework by CL4SRec [13]. While recommendation datasets are used for evaluation, reward-based datasets are not taken into account, and reinforcement learning is not integrated into our methodology.The graph contrastive learning paradigm [14] trains embeddings self-guidedly, thereby reducing the randomness of message loss. The model contrasted with GNN-based recommendation models and multiple matrix factorization is shown in the image.
Additionally, research on slate-based recommendations was done in [15]. This work is regarded as a slate. If you enable this setting, your workspace will increase significantly. Lastly, from the standpoint of long-term optimization, the Bandit algorithm is likewise compensation-centric. But the Bandit algorithm believes that actions have no effect on the state, whereas recommendations have an impact on the user's behavior. RL is a wise option for RS operation as a result. Imitation learning, which teaches policies via expert demonstrations, is another similar field.

3. Methods

Recommendation in RL Problem

Considering the recommendation problem from the standpoint of reinforcement learning offers distinct insights for user preference modeling and optimization of suggestion strategies. According to this concept, a recommender system gains knowledge about how to communicate with people and objects by carrying out a task (making recommendations), watching for rewards that follow (getting feedback from users), and gradually enhancing its policy. It, based on user profile, context, and interaction history, which products or content to suggest in response to incoming user queries. The recommendation issue is stated as a Markov decision process, with a state space S and an action space A, represented by the tuple ▨S, A, P, R.Objects that are recommended are represented by task aÎA, and interactive objects, or status sÎS, indicate the user's interest. The distribution of possible transitions that capture st+1 to P(.|st,at) is denoted by P. Lastly, the immediate compensation received by carrying out action a on state s is represented by reward r(s,a). Map learning and reinforcement learning can share knowledge thanks to this pattern of base model sharing. The first stage's TD (time difference) error is used to define the loss of the reinforcement learning component.

$$L_Q = E[r(s_t, a_t) + \gamma \max_{at+1} Q(s_{t+1}, a_{t+1} - Q(s_t, a_t))^2]$$

For each subsequent state-behavior combination, the discounted projected Q value is subtracted from the actual observed compensated total to determine the TD error. In this approach, learning losses from both teaching and reinforcement occur simultaneously throughout training.

Supervised Negative Q-learning (SNQN)

The cross-entropy across the classification distribution can be used to define the supervised training loss given an input user-item interaction sequence $x1{:}t$ and an existing recommendation model (·).

$$L_s = -\sum_{i=1}^{n} y_i \log(p_i), where\ p_i = \frac{e^{y_i}}{\sum_{i'=1}^{n} e^{y_{i'}}} (4)$$

As soon as the user interacts with the j-th item at the following timestamp, the display function Y is declared as Y= 1. If not, Y = 0. It is clear that the positive logit climbs to greater levels as a result of the cross-entropy loss. Conversely, cross-entropy loss may result in a negative learning signal by lowering the output value of objects that have not yet been touched by the user. This is especially helpful in RS situations, where the primary objective is to rank items according on how probable it is that users would interact with them in the parent place. We may utilize the latent state st for RL training directly as (·) has already encoded the input sequence. We create an additional output layer that converts the shared base model (·) states into Q-values.

$$Q(s_t, a_t) = \delta(s_t h_t^T + b) = \delta(G(x_{i:t} h_t^T + b) \quad (5)$$

where ht and b are the trainable parameters of the Q-learning output layer, and δ is the activation function. Negative reward signals are frequently absent when learning from logged implicit feedback data (Hu et al. 2008, Rendle et al. 2009). As a result, suggestions based just on observed (positive) behaviors cannot be produced using such output Q values. In Figure 4.4, the supervised negative Q-learning architecture is shown.

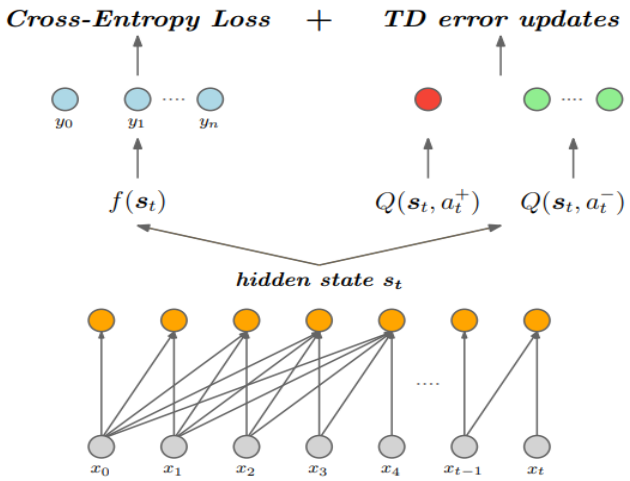

Figure 4.4 Learning Framework Architecture of SNQN

To this end, define the one-step time difference (TD) Q-loss of SNQN as:

$L_q = L_p$ (positive TD error)+ L_n (negative TD error) (6)

Consequently, at timestamp t, the positive action is represented by $a + t$ and the negative action by $a - t$. For negative activities, a constant reward value, jn, is assigned. Next, using the replay buffer created by the logged implicit feedback data, train both the supervised loss and the RL loss simultaneously:

$$L_{snqn} = L_s + L_q \quad (7)$$

For improved learning stability, double Q-learning is applied (Hasselt, 2010), where two copies of the model parameters are alternatively trained. Thus, recommendations can be produced by the supervised head as well as the RL head. Research indicates that teaching both heads simultaneously using a common base model leads to superior performance compared to individual learning.

Overestimation is another issue with Q-learning techniques, however SQN can help with it. This approach has been effectively used recently in conjunction with function approximation in the form of SQNs. The technique, known as SNQN, is a modification of the conventional SQN algorithm to the idea of supervised Q-Learning. For Q-learning algorithms, the generic weight update rule can be expressed as

$$\theta_{t+1} = \theta_t + \alpha\big(Y_t - Q(S_t, A_t; \theta_t)\big)\nabla_{\theta_t} Q(S_t, A_t; \theta_t)$$

The target used in the weight update at step t is denoted by Y_t^Q in this instance. Every algorithm that is given has a distinct version of this. The objective Y_t^Q for the conventional Q-learning method with a single network is represented by the following expression:

$$Y_t^Q = R_{t+1} + \gamma \max_a Q(S_{t+1}, a; \theta_t)$$

The target for SQN can be written as

$$Y_t^{SQN} = R_{t+1} + \gamma \max_a Q(S_{t+1}, a; \theta_t^-)$$

where θ_t^- denotes the target network's parameters, which are copies of the primary network's parameters that are changed every N^-steps. The aforementioned goal is represented in conventional Supervised Q-learning by

$$Y_t^{SupervisedQ} = R_{t+1} + \gamma Q(S_{t+1}, \operatorname{argmax}_a Q(S_{t+1}, a; \theta_t); \theta_t')$$

Having two distinct sets of network parameters represented by δ_t and δ_t^'. Both networks alternately take on different roles during training; that is, one is utilized as the target network and the other to update its weights at random, resulting in a training that is balanced for both. The objective suggested for the SNQN strategy, which combines the two predecessors that were previously presented—Y_t^SupervisedQ and Y_t^SNQN—is the last one that can be utilized. The following can be written for it:

$$Y_t^{SNQN} = R_{t+1} + \gamma Q(S_{t+1}, \underset{a}{\operatorname{argmax}}\, Q(S_{t+1}, a; \theta_t); \theta_t^-)$$

The technique combines the two approaches to embrace the advantages of Supervised Q-Learning while maintaining the idea of periodically copying parameters from the main network. While keeping the idea of replicating the parameters of the main network per N^-steps, it also adopts the important idea from Supervised Q-Learning, which is to separate the actual value retrieval for S_(t+1) and the selected argmax┬a from the maximization step.

Supervised Advantage Actor-Critic (SA2C)

RL and supervised learning are combined with the shared base model by SNQN, which also integrates negative sampling within the RL training process. The sampled actions, which can be viewed as normalized Q-values, are used by SA2C to determine the benefit of activities in the beginning. On the basis of this advantage estimate critique, the performer is then reweighted. In RL research, actor-critic (AC) methods have been successfully used. The actor, the supervised element of the suggested SNQN technique, aims to mimic the behavior of the logged user. It is simple to respond to the critic's objection by using the output Q-values from the RL head because they measure the overall benefits the system receives for the state-action pair. These Q-values are influenced by the exact date of the series. Since Q-values are based on the total gains of all the following actions in this series, a negative action at the beginning of a long series, for example, may also have a high Q-value. This benefit can assist us in reducing the bias that the sequence timestamp introduced. Figure 4.5 shows the SA2C architecture.

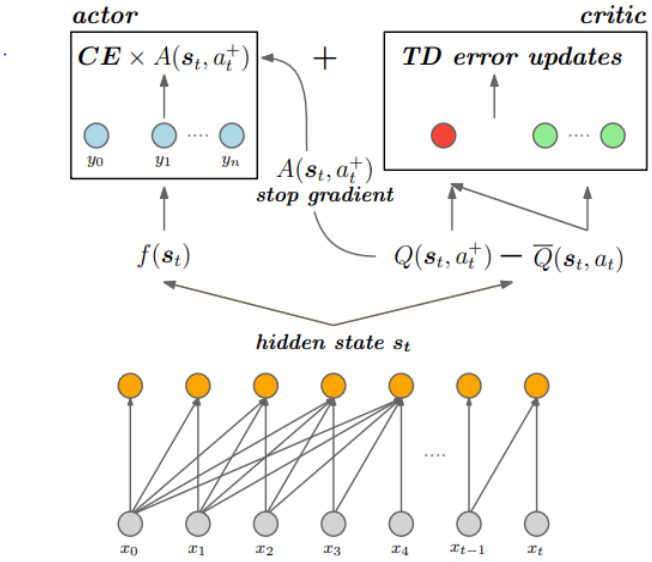

Figure 4.5 Learning framework architecture of SA2C.

On the other hand, computing the average Q-values over the entire action space would incur extra computing costs, particularly if the candidate item set is big. Because of this, negative samples are included in the suggested SNQN techniques. Therefore, a simple method is to approximate the average of the sampled acts (good and negative instances included). This motivation allows us to define the average Q-values as follows:

$$\bar{Q}(s_t, a) = \frac{\sum_{a' \in a_t^+ \cap N_t} Q(s_t, a')}{|N_t| + 1} \quad (8)$$

An observed (positive) action's benefit is expressed as follows:

$$A(s_t, a_t^+) = Q(s_t, a_t^+) - \bar{Q}(s_t, a) \quad (9)$$

Next, reweight the actor (i.e., the supervised head) by utilizing this advantage. In the supervised training process, give more weight to good actions if they outperform the average, and vice versa. When determining the average and advantage, halt the gradient flow and adjust the Q values to improve stability. Next, train the reviewer and the actor together. The SA2C training loss is expressed as follows:

$$L_{sa2c} = L_a + L_q \text{ , where } L_a = L_s . A(s_t, a_t^+) \quad (10)$$

Q-value learning can be unstable during the training process, especially in the early stages (Parisotto et al., 2019). With the exception of reweighting Ls and computing advantage, the training process of SA2C is identical to that of Algorithm 4.2 and involves double Q-learning. Benefit values can be thought of as normalized Q-values that help reduce the bias that results from overestimating the negative effects on Q-value estimates. The off-policy learning outlined in the recommendation phase is then implemented by combining this with a propensity score.

Let M = {S, A, T, R, γ} represent a Markov Decision Process (MDP), where S stands for the state space, A for the action space, T for the transition dynamics, R for the reward, and γ for the discount factor. The trained model's output represents the state space of the MDP in the specified active learning context, and the action consists of subsampling informative cases with generated labels. If the state space is above the positive threshold, designating the region of interest as a genuine nodule, or below the negative threshold, designating the sample as normal, then it is deemed informative. The sampling strategy that yields the most expected reward is the best course of action, or π ∗.

While there shouldn't be any limitations on the trained model's architecture that would prevent us from using a SA2C technique, U-Net [2] segmentation network—which offers cutting-edge performance in the field of medical imaging—was employed. A reward function approximation based on expert demonstration was created using maximum margin IRL [16] since the U-Net's final performance is unmanageable during the training rounds. This was utilized to update the A2C network along with validation accuracy.

We will formalize the agent's interactions with the environment to take advantage of unlabeled data based on input from the environment, using the provided MDP definition. Like other semi-supervised algorithms, the proposed model comprises two phases to its training.

Phase 1 (Supervised Learning): 25%, 50%, 75%, and 100% of the available labeled data are used in each of four training settings to create a U-Net-like model. Following phase 2, the SSRAL training phase, these will be compared to see how reliable the approach is with fewer labeled data.

In phase 2, the environment that interacts with the agent is the trained model. For the purpose of avoiding needless information loss due to the final sigmoid activation layer, the state space will be defined as the logit output of the model.

By the end of the first phase, the state space of the labeled training data and its original labels are used to create a set of expert demonstrations from the environment. Using the greatest margin IRL, the reward function R∗ is estimated from the expert's behavior.

$$\mathbb{E}[\sum_{t=0}^{\infty} \gamma^t R(s_t)^* | \pi^*] \geq \mathbb{E}[\sum_{t=0}^{\infty} \gamma^t R(s_t)^* | \pi] \quad \forall \pi$$

In phase 2, the policy is assessed by the critic using the approximation of the reward function. The reward function itself is not updated iteratively.

Phase 2 (SA2L): Based on the current strategy, which begins with random initialization, a subset of the unlabeled data is formed and utilized to fine-tune the segmentation network. The long-term payoff at this point is the validation accuracy r that results. The A2C network's policy and value function are trained to optimize the long-term reward (r) and the short-term reward (R∗ (s) supplied by the IRL trained in phase 1. R∗ (s) is used to stabilize learning, which was unstable when it was solely based on r. Using both R∗ (s) and r, the value function is computed via temporal difference (TD) methods [17].

$$\delta^{\pi\theta} = r \cdot R^*(s) + \gamma V^{\pi\theta}(s') - V^{\pi\theta}(s)$$

The policy gradient serves as the basis for the actor's iterative update.

$$\theta = \theta + \alpha \nabla_\theta \log \pi_\theta(s,a) Q_w(s,a)$$

where an objective assessment of the advantage function is represented by the real TD error. It should be noted that this approach describes a clear relationship between the ultimate performance and the iterative updates, in contrast to many of the earlier active learning algorithms.

Methodology

Brain Strom Optimization

Driven by the process of human creative idea generation, or brainstorming, Shi first put forth a promising swarm-based metaheuristic called BSO. The original BSO has an easy-to-implement design and is simple to use. In recent years, a variety of challenging issues, such as distributed flow shops, knowledge spillover concerns, and real-parameter numerical optimization, have been successfully tackled by BSO and its derivatives. Extensive testing has confirmed that BSO has strong performance to offer an exceptional balance between exploration and exploitation capabilities.

BSO starts with a population made up of several candidate people, each of whom represents a potential solution to the optimization issue. After that, it searches for solutions through three stages: clustering, creating, and selecting. Using a clustering strategy, the population is divided into multiple different clusters during the clustering phase. The center individual for each cluster is the best individual within it; the other individuals are considered to be the regular ones. During the generating phase, one or two people from clusters are used to create new individuals. Every freshly generated person is matched with an already-existing person. A selection approach is used in the selecting phase to store the superior person from the paired individuals and save it for the following population. Ultimately, the three stages are repeated until the ideal solution is obtained and a termination requirement is met.

Recommendation Phase

In observational studies, propensity scoring is a statistical method that is frequently used to quantify the efficacy of an intervention by taking into account the factors that predict getting the therapy. The likelihood that an action will be selected by the behavior policy is frequently equal to the propensity score of that action in the context of reinforcement learning (Chen et al., 2019a). Propensity score-based off-policy correction in reinforcement learning and importance sampling (IS) are similar techniques. Both approaches aim to compensate for the difference between the behavior-generating (data-generating) policy and the target policy. The expected value under a distribution can be estimated using samples from that distribution and the IS method.The suggested system's flow is depicted in figure 4.6.

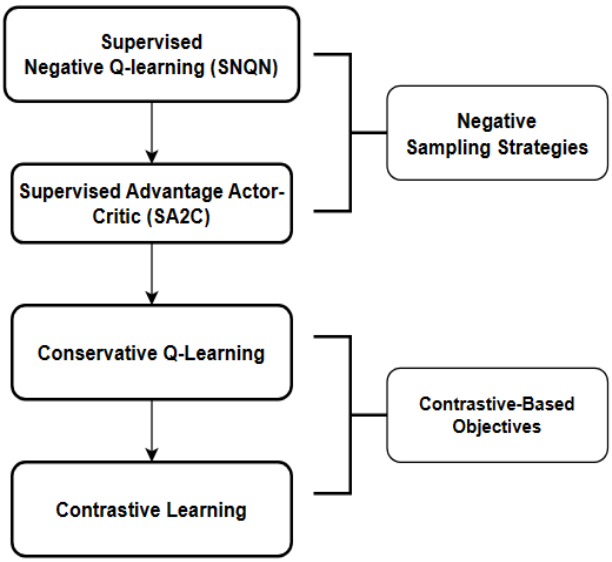

Figure 4.6 Flow of Proposed System

Inaccurate computation of propensity scores and advantage function estimates can lead to bias. Furthermore, the previously cited data shows that instability and possible divergence can result from large volatility in the propensity scores or estimated advantage function. Overly optimistic Q-value estimates could be the cause of this instability, which is a particular kind of learning process instability. The aforementioned elements add two families of enhancements to this learning algorithm framework: To address challenges associated with off-policy training, (1) apply conservative Q-Learning Kumar et al. (2020); (2) add a contrastive learning target to enhance the quality of learnt representations even more.

Conservative Q-Learning

Issues may arise when off-policy modifications are made using IS or propensity ratings. A problem arises when there is a large difference between the target policy and the behavior policy, specifically the high variation of IS. This happens as a result of the IS ratio become unnecessarily high or low. The propensity score method may face comparable difficulties. As a result, as seen in Figures 3 and 4, the high variance may cause instability in the learning process, which in turn may cause the Q-function to diverge. We propose that if the benefit function and propensity scores are not computed appropriately, then estimating them both may cause bias. Errors in modeling, estimate, or function approximation can all lead to this bias. Furthermore, the previously cited data shows that instability and possible divergence can result from large volatility in the propensity scores or estimated advantage function. Overly optimistic Q-value estimates could be the cause of this instability, which is a particular kind of learning process instability. Q-values that are overestimated may result in inaccurate learning and poor policy performance.

Constructed by Kumar et al. (2020), Conservative Q-Learning (CQL) aims to resolve the overestimation issue that arises often while using Q-learning. With probable out-of-distribution and in-distribution actions (activities that are included in the dataset) taken into account, the main objective of CQL is to minimize an upper bound on the policy's expected value. By decreasing the subsequent goal, this is accomplished.

$$L_{CQL}(\theta) = E_{(s,a,r,s')\sim D}\left[\left(Q_\theta(s,a) - r - \gamma E_{a'\sim\pi_{\emptyset}(a'|s')}[Q_{\theta'}(s',a')]\right)x^2\right] + \alpha E_{s\sim D}[\log \sum_a \exp Q_\theta(s,a) - E_{a\sim\pi_\beta(a|s)}[Q(s,a)]]$$

The variables D, θ, □′, ϕ, γ, and α represent the fixed dataset, ϕ, the policy parameters, γ, the discount factor, and α, the temperature parameter that controls the trade-off between the Q-function reduction and the conservative regularization. Figure 4.7 shows the model architecture for the training process as well as the connection between Q-learning and the transformer model with recommended objectives.

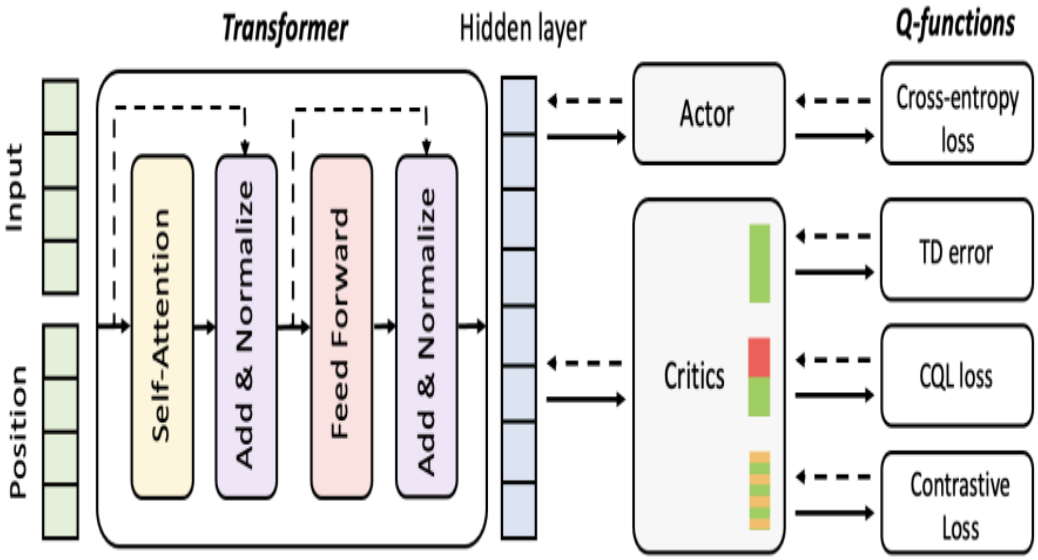

Figure 4.7 Proposed architectural framework

This method is based on the idea that these missed interactions indicate a collection of things the user is not interested in. The quality of learned representations is then enhanced through the application of contrastive learning.

Contrastive Learning with Temporal Augmentations:

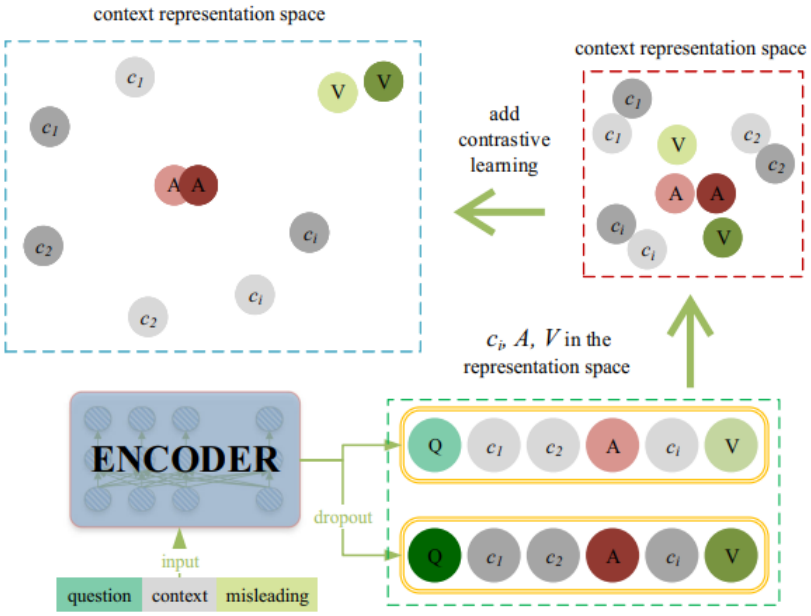

Figure 4.8 Contrastive Learning Method.

Learning effective representations is aided by the widely used loss function in contrastive learning, InfoNCE. Figure 4.8 illustrates the contrastive learning method's strategy. Q stands for question phrases. Context is

indicated by C, other sentences in the context are indicated by ci, the sentences containing the answer are indicated by A, and deceptive sentences are shown by V. A pair of positive samples produced via dropout are represented by two sentences of comparable colors. The contrastive learning module may successfully separate the response sentence from other sentences, such as deceptive statements, when the model encodes hostile data. Positive sample pairs (xj, yj) and a set of negative samples (yj, k) are used to compute the objective.

$$L_{InfoNCE} = -\frac{1}{M}\sum_{j=1}^{M}\log\frac{\exp\left(f(x_j,y_j)\right)}{\exp\left(f(x_j,y_j)\right)+\sum_{k=1}^{K}\exp\left(f(x_j,y_{j,k})\right)} \quad (12)$$

The similarity between xj and a negative sample is measured by f(xj, yj,k), where f(xj,yj) is the similarity function between the representations of xj and yj. The number of positive sample pairs is denoted by yj, k, and M. For every pair of positive samples, K is the number of negative samples.. In order to learn meaningful representations, the InfoNCE loss seeks to maximize similarity between positive pairings and minimize similarity between negative pairs. The best way to improve model performance is to combine it with contrastive learning. With a recommender system that uses a static dataset, this approach is especially helpful in situations where online interaction is too expensive or impossible. The ultimate goal of optimization becomes:

$$L = L_{CE} + \omega L_Q + L_{CO} + \alpha L_{CQL} \quad (13)$$

where LQ is the Q-learning, or TD loss, LCE is the cross-entropy loss, LCO is the contrastive objective, and LCQL is the conservative Q-learning objective.

The suggested algorithm, known as CQL-BSO, effectively combines CQL with BSO to enhance BSO's performance for the examined problem. The CQL-BSO steps are listed in detail below.

First, 90% of the population should be created at random, and 10% should be generated using the SQN-based approach.

2. Based on selection probabilities, choose an update strategy for each individual and use various strategies to update the population.

3. To further enhance its quality, carry out local searches for every individual.

4. Adjust the selection probabilities by QL and start a new iteration if the termination condition is not satisfied; if it is, the algorithm ends.

5. Enter the QL area. Update the Q-table, award, and system state. After choosing the new action and carrying it out to modify the probability, move on to step (2).

Results and Discussion

Dataset

Explain the experiment carried out on five real-world datasets in this section to assess the effectiveness of the suggested techniques, BSO and CQL. Use of electronic data has allowed for the extraction of useful information. Patient electronic medical data is a difficult endeavor. Two distinct datasets related to diabetes are used to simulate the suggested model. The "PIMA INDIAN Diabetes Database" is a dataset gathered from Indian health organizations, while the electronic version of the "Hospital Frankfurt Germany" is another diabetes dataset. These datasets were gathered from Kaggle and comprise the Frankfurt dataset, which has 2000 cases, and the PIMA dataset, which contains 768 instances with 9 attributes that are based on the target class of diabetes. Table 4.1, which is provided below, discusses and describes each quality.

Table 4.1 Feature information about the diabetes disease

Sr.no.	Attribute	Unit	Ranges
1	Age	Year	01-120
2	Family History	Yes (1), No(0)	0,1
3	Glucose	Mg/DI	37-380
4	Skin Thickness	Mm	0-210
5	Blood Pressure (BP)	Mm, Hg	90-190
6	Pregnancies	Number (0 -9)	0-8
7	Insulin	uU/ml	0-764
8	BMI	Kg/m^2	14-80.6
9	Diagnosis result	Positive (1) , Negative (0)	1, 1

Baselines

In order to evaluate the performance of the method in this paper, three baseline methods were used.

SASRec (Wang & Julian, 2018) is a self-attention-based sequential model, which uses an attention mechanism to identify relevant items for predicting the next item.

SNQN, SA2C (Xin et al., 2022) Baselines SNQN performs a naive negative sampling; SA2C includes advantage estimations to re-weight the Q-values.

SASRec_AC(Xin et al., 2020) self-supervised reinforcement learning for sequential recommendation tasks. All models uses SASRec model as base model and use actor-critic framework.

4.5.3 Evaluation Protocols

The purpose of the experiment was to evaluate how well the suggested model could diagnose diabetes. The information was first gathered from sensors and sent to the feature database via IoMT. Similar to this, unstructured data about the patient—such as lab reports, inquiries, observations, and medical histories—were transformed into structured format in order to undergo additional pre-processing. Additionally, the diabetes datasets from Pima Indians and Hospital Frankfurt Germany were used to train the diabetes illness prediction model. Xin et al. (2022) proposes a data split that is used to test the performance of the suggested approaches through an adaptation of cross-validation. Two metrics are used to assess the quality of recommendations: top-k Normalized Discounted Cumulative Gain (NDCG@k) and top-k Hit Ration (HR@k). A recall-based statistic called HR @ K determines if the ground truth item appears in the top k positions of the list of recommendations. The recommendation list's top spots are given greater scores by the rank-sensitive NDCG metric.

The suggested model's performance as a recommendation system is assessed using precision, recall, and F1 score. The precision of a forecast can be defined as the ratio of correct positive predictions to total positive predictions. Recall is defined as the percentage of positive actuals to positive right predictions. The weighted harmonic average of Precision and Recall is the F1− score. The following is the calculating formula:

$$\text{Precision} = \frac{TP}{TP + FP}$$

$$\text{Recall} = \frac{TP}{TP + FN}$$

$$\text{F1} = \frac{2 \times \text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}$$

$$\text{Accuracy} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{TN} + \text{FP} + \text{FN}}$$

where FP is the number of normal points that were mistakenly recognized as anomaly points, FN is the number of anomaly points that were mistakenly identified as normal points, and TP is the number of anomaly points that were successfully detected.

Tabular Results

The contrastive aim is only applied across data batches as positive and negative items in this case; there is no negative action sampling. The suggested method uses both the contrastive and conservative aims and accepts negative action sampling. Throughout the studies, there has been a consistent pattern whereby the baseline techniques, SA2C and SNQN, initially achieve high accuracy but quickly lose performance as Q-learning diverges. The performance comparison of several models on the diabetes datasets from Pima Indians and Hospital Frankfurt, Germany, is shown in tables 4.2 and 4.3.

Table 4.2 Performance comparison of different models on Hospital Frankfurt Germany diabetes dataset

Model	Acc (%)	Pre (%)	Rec (%)	F1 (%)
SASRec	69.5	0.53	0.69	0.58
SASRec_AC	65.1	0.62	0.72	0.68
SNQN	79.2	0.75	0.83	0.6
SA2C	82.3	0.78	0.79	0.75
SA2C Smooth Enabled	87.6	0.86	0.85	0.81
Proposed	95.2	0.92	0.91	0.93

Table 4.3 Performance comparison of different models on Pima Indians diabetes dataset

Model	Acc (%)	Pre (%)	Rec (%)	F1 (%)
SASRec	75.2	0.62	0.62	0.62
SASRec_AC	74.1	0.71	0.72	0.71
SNQN	80.5	0.69	0.69	0.76
SA2C	72.2	0.75	0.75	0.82
SA2C Smooth Enabled	85.7	0.83	0.83	0.87
Proposed	93.3	0.96	0.96	0.97

Results of accuracy, precision, recall, and F1 score are displayed in Figures 4.9, 4.10, 4.11, and 4.12, which assess the suggested model's performance as a recommendation system. These findings indicate that

higher performance can be attained and stays constant with more negative samples, in contrast to baseline techniques SNQN and SA2C, which demonstrate performance degradation and divergence.

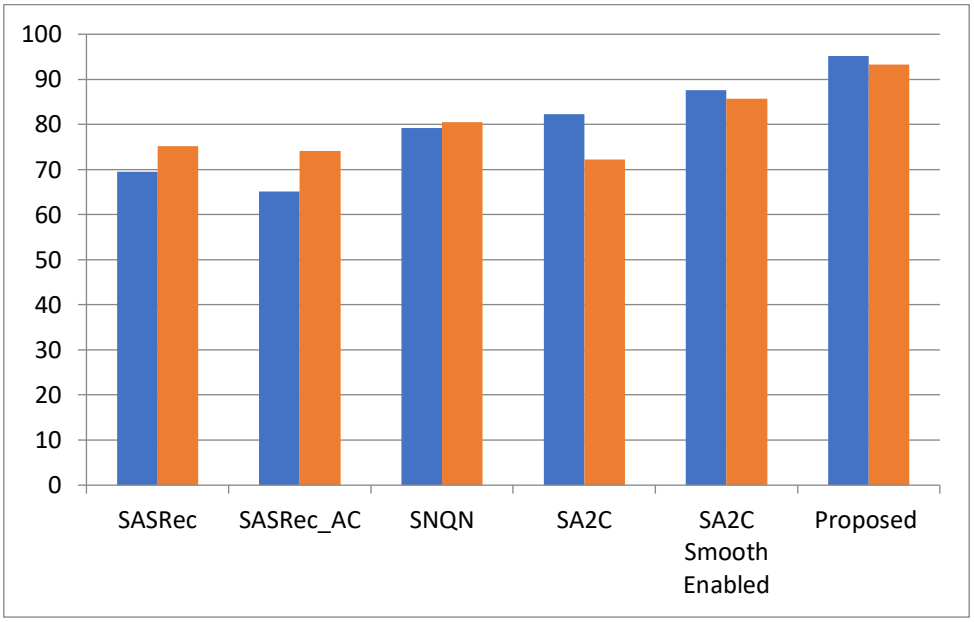

Figure 4.9 Accuracy

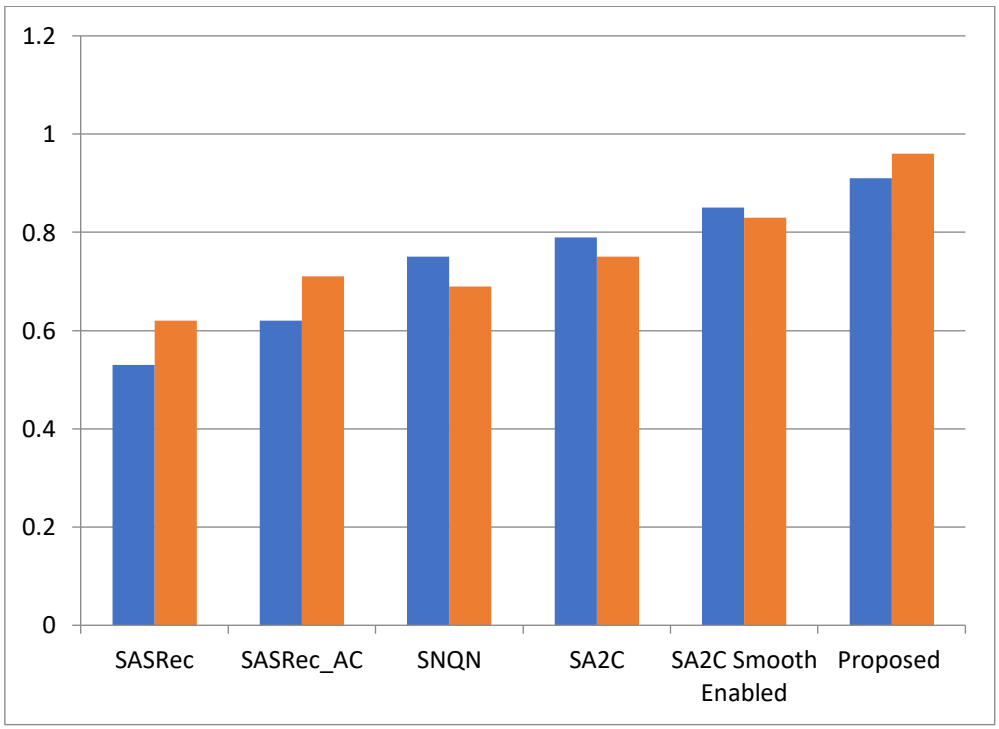

Figure 4.10 Precision

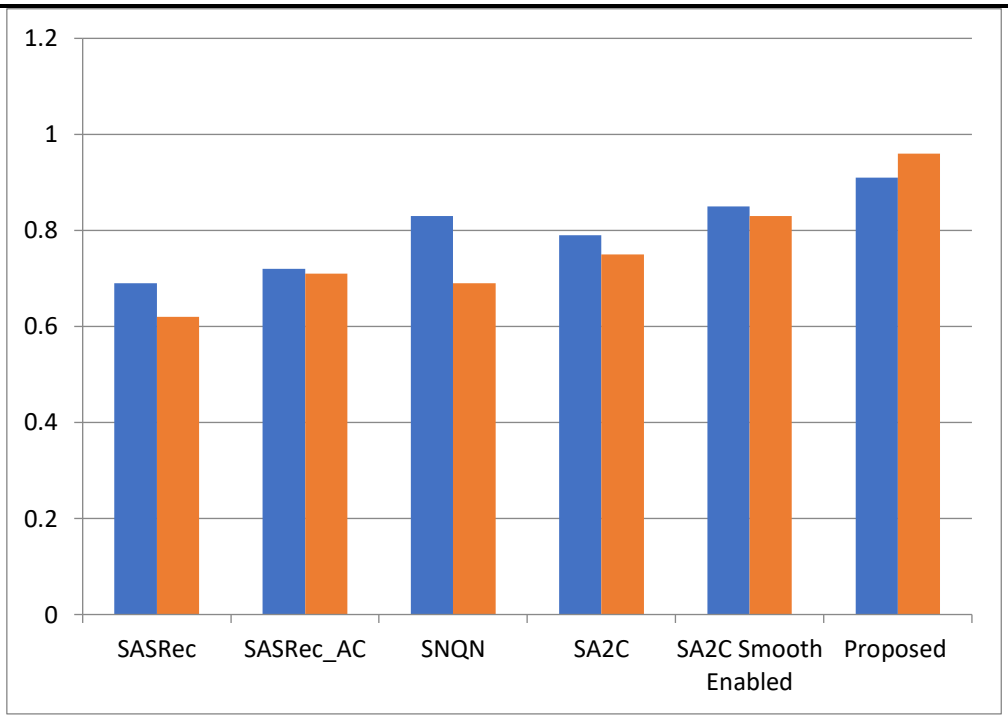

Figure 4.11 Recall

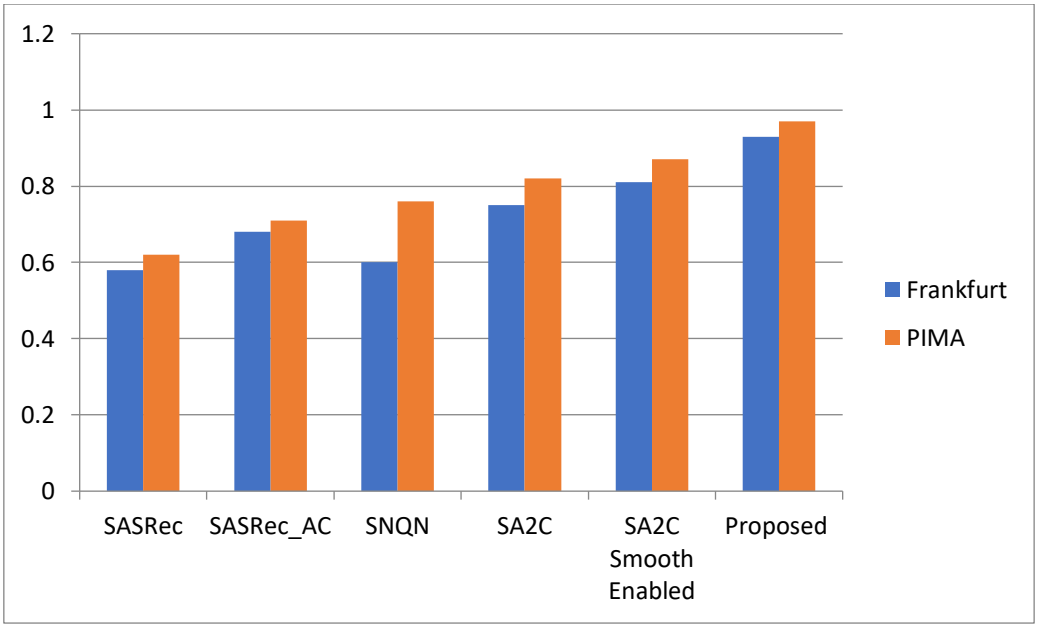

Figure 4.12 F1Score

Across all studied datasets, the proposed technique performs better than the baseline models. Combining improvements to the negative sampling process with a sequential contrastive learning aim routinely yields better results than the baselines. A new study reveals how well contrastive learning may be integrated into recommender systems. This method enhances the learning capacity of the Q-function in the contrastive embedding space by offering richer representations of states and actions. As such, it makes it possible to distinguish between states and actions more precisely. Because Q-learning is conservative, it creates a useful equilibrium by avoiding overestimating Q-values, which may otherwise result in less-than-ideal policies. This Q-learning adjustment protects against too optimistic assumptions about the rewards that come with doing

particular things. It is also found that adding negative action sampling greatly improves the model's overall performance and guarantees stability during RL training.

The suggested approach training framework can successfully enhance recommendation performance, as evidenced by the experimental findings. It is difficult to foresee human diseases, especially multimodal diabetes, in order to provide more effective and timely care. Multidisciplinary diabetes is a potentially fatal disorder that affects many vital human body parts. A suggested methodology is provided to rapidly and effectively forecast and suggest interdisciplinary diabetes disease in individuals. In addition to effectively predicting and recommending whether or not the patient has multidisciplinary diabetic illness, the suggested SHRS-M3DP model can also determine the impact of the following human body parts: Ultimately, the analysis of this research revealed that, when compared to previously published methods, the suggested model's overall performance is an impressive 99.6%.

Conclusion

Our study provides new perspectives on how well contrastive learning can be incorporated into recommender systems. The learning ability of the Q-function in the contrastive embedding space is enhanced by this method, which offers richer representations of states and actions. As such, it makes it possible to distinguish between states and actions more precisely. Furthermore, Q-learning's conservative character creates a useful equilibrium by avoiding overestimation of Q-values, which may otherwise result in less-than-ideal policies. This Q-learning adjustment protects against too optimistic assumptions about the rewards that come with doing particular things.Furthermore, we found that adding negative action sampling improves the model's overall performance and guarantees stability during RL training. This combination represents a significant gain in our understanding of reinforcement learning and makes a significant contribution to the subject, although not being revolutionary.

References

[1] Afsar, M. M., Crump, T., and Far, B. (2022). Reinforcement learning based recommender systems: A survey. ACM Computing Surveys, 55(7):1–38.

[2] Zhou, K., Wang, H., Zhao, W. X., Zhu, Y., Wang, S., Zhang, F., Wang, Z., and Wen, J.-R. (2020). S3-rec: Self-supervised learning for sequential recommendation with mutual information maxi mization. In Proceedings of the 29th ACM International Conference on Information and Knowledge Management, CIKM ’20, page 1893–1902, New York, NY, USA. Association for Computing Machinery.

[3] Xin, X., Karatzoglou, A., Arapakis, I., and Jose, J. M. (2022). Supervised advantage actor-critic for recommender systems. In Proceedings of the Fifteenth ACM International Conference on Web Search and Data Mining, WSDM ’22, page 1186–1196, New York, NY, USA. Association for Computing Machinery.

[4] Xie, X., Sun, F., Liu, Z., Wu, S., Gao, J., Ding, B., and Cui, B. (2020). Contrastive learning for sequential recommendation. https://arxiv.org/abs/2010.14395.

[5] Xin, X., Karatzoglou, A., Arapakis, I., and Jose, J. (2020). Self-supervised reinforcement learning for recommender systems. In Proceedings of the 43th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR ’20).

[6] Tang, J., Drori, Y., Chang, D., Sathiamoorthy, M., Gilmer, J., Wei, L., Yi, X., Hong, L., and Chi, E. H. (2023). Improving training stability for multitask ranking models in recommender systems. arXiv preprint arXiv:2302.09178.

[7] Li, J., Zhang, W., Wang, T., Xiong, G., Lu, A., and Medioni, G. (2023). Gpt4rec: A generative framework for personalized recommendation and user interests interpretation.

[8] Kumar, A., Zhou, A., Tucker, G., and Levine, S. (2020). Conservative q-learning for offline reinforcement learning. In Larochelle, H., Ranzato, M., Hadsell, R., Balcan, M., and Lin, H., editors, Advances in Neural Information Processing Systems, volume 33, pages 1179–1191. Curran Associates, Inc.

[9] He, K., Fan, H., Wu, Y., Xie, S., and Girshick, R. B. (2020). Momentum contrast for unsupervised visual representation learning. In 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2020, Seattle, WA, USA, June 13-19, 2020, pages 9726–9735. Computer Vision Foundation / IEEE.

[10] Gilmer, J., Ghorbani, B., Garg, A., Kudugunta, S., Neyshabur, B., Cardoze, D., Dahl, G., Nado, Z., and Firat, O. (2021). A loss curvature perspective on training instability in deep learning. arXiv preprint arXiv:2110.04369.

[11] Lixin Zou, Long Xia, Zhuoye Ding, Jiaxing Song, Weidong Liu, and Dawei Yin. 2019. Reinforcement Learning to Optimize Long-term User Engagement in Recommender Systems. arXiv preprint arXiv:1902.05570 (2019)

[12] Christakopoulou, K., Xu, C., Zhang, S., Badam, S., Potter, T., Li, D., Wan, H., Yi, X., Le, Y., Berg, C., Dixon, E. B., Chi, E. H., and Chen, M. (2022). Reward shaping for user satisfaction in a reinforce recommender.

[13] Chen, T., Kornblith, S., Norouzi, M., and Hinton, G. (2020). A simple framework for contrastive learning of visual representations. In International conference on machine learning, pages 1597 1607. PMLR.

[14] Chang, J., Gao, C., Zheng, Y., Hui, Y., Niu, Y., Song, Y., Jin, D., and Li, Y. (2021). Sequential recommendation with graph neural networks. In Proceedings of the 44th international ACM SIGIR conference on research and development in information retrieval, pages 378–387.

[15] Gao, T., Yao, X., and Chen, D. (2021). Simcse: Simple contrastive learning of sentence embeddings. In Moens, M., Huang, X., Specia, L., and Yih, S. W., editors, Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, EMNLP 2021, Virtual Event / Punta Cana, Dominican Republic, 7-11 November, 2021, pages 6894–6910. Association for Computational Linguistics.