Reinforcement Learning-based Product Delivery Frequency Control

Yang Liu, Zhengxing Chen, Kittipat Virochsiri,
Juan Wang, Jiahao Wu, Feng Liang
Facebook, Menlo Park, CA, USA, 94025
{yliu9, czxttkl, kittipat, juanw, jiahaowu, liangfeng}@fb.com

Abstract

Frequency control is an important problem in modern recommender systems. It dictates the delivery frequency of recommendations to maintain product quality and efficiency. For example, the frequency of delivering promotional notifications impacts daily metrics as well as the infrastructure resource consumption (*e.g.* CPU and memory usage). There remain open questions on what objective we should optimize to represent business values in the long term best, and how we should balance between daily metrics and resource consumption in a dynamically fluctuating environment. We propose a personalized methodology for the frequency control problem, which combines long-term value optimization using reinforcement learning (RL) with a robust volume control technique we termed "*Effective Factor*". We demonstrate statistically significant improvement in daily metrics and resource efficiency by our method in several notification applications at a scale of billions of users. To our best knowledge, our study represents the first deep RL application on the frequency control problem at such an industrial scale.

Introduction

Frequency control is a common and essential problem in recommender systems. A successful recommender system should have careful control of the recommendation delivering frequency in order to foster an engaging interaction with users while consuming infrastructure resources minimally. For example, social media platforms use promotional Emails to remind users of missed information but also aspire to minimize the delivery volume to avoid resource waste or creating spams (Gupta et al. 2016; Gupta, Liang, and Rosales 2017; Zhao et al. 2018a). Generally speaking, an increase of delivery frequency is able to improve product metrics transiently at the cost of infrastructure resources. However, too frequent recommendation may result in user fatigue in the long term (Ma, Liu, and Shen 2016; Gupta et al. 2016) and potential risks of shutting down of recommendation channels by users (Iqbal and Horvitz 2010).

Existing works converted the frequency control problem into constrained optimization, putting daily metrics and resource consumption into the objective and constraints separately to counteract with each other (Gupta et al. 2016; Gupta, Liang, and Rosales 2017; Zhao et al. 2018a). However, the daily metrics were still defined based on short term (*e.g.*, user activeness in one day) or based on heuristics. In the industry, companies care most about *accumulated* daily metrics and resource consumption over the long term, which we believe should be reflected in the optimization objective in a more principled way. In this light, frequency control should be a sequential decision problem as the user interacts with the recommender system continuously. We illustrate with the following motivating example where previous methods fall short. An inactive user has low interest to recommendations that an constrained optimization algorithm never considers it worth any system resources to deliver recommendation to the user. However, a sufficient amount of recommendations in multiple waves would change the user behavior, turning the user into a regular active one. Clearly, an algorithm that can plan its decisions sequentially will win in the long run.

This paper proposes a methodology for learning frequency control policies that optimize long-term accumulated measurements with the sequential nature of decisions in mind. We leverage reinforcement learning (RL) to learn the *value* of different frequencies, defined as the best possible accumulated daily metrics and resource consumption since the product is delivered at the frequency. The accumulation measurements represent the potential performance of a system in the long term and are closer to common business metrics companies aim to lift. RL has been demonstrated to be capable of learning long-term values of decisions in complex sequential-decision domains such as games (Mnih et al. 2015a) and robotics (Rajeswaran et al. 2017), where accumulated rewards are more important than the immediate reward at any single step. Take *Go* for an example. It is useful to evaluate a move based on whether it contributes to the final victory but less useful (and sometimes deceptive) to know whether the move brings short-term board advantage. Similarly, the frequency control can also be formulated as a sequential decision problem where a sequence of decisions on delivery frequency would collectively optimize long-term (accumulated) daily metrics and resource usage. In particular, we choose to apply Deep Q-Networks (DQN), an off-policy value-based reinforcement learning algorithm (Sutton and Barto 2018) to estimate the values of (user, frequency) pairs.

Another challenge we aspire to solve is to stabilize global delivery volume at deployment, for which we propose a simple yet robust technique termed "*Effective Factor*". In a large-scale industrial setting, we observed that the policy directly derived from the values learned by DQN could result in noticeable fluctuation in global delivery volume because the user and system behavior could shift day by day in the real world. While the data shift is not drastic generally, the change in delivery volume can be significant enough to trigger alerts from other monitoring systems present in modern platforms. Even recurrent training cannot help RL models catch up with the latest data shift because RL models train on user history data spanning a few weeks. Therefore, we propose Effective Factor which monitors global delivery volume and acts on top of DQN to stabilize output frequencies dynamically.

Contributions. Our study represents:

- Formulate the frequency control problem as a sequential decision problem.
- Propose to solve the frequency control problem by reinforcement learning and Effective Factor (dynamic volume control).
- Evaluate our method and show positive gains in both daily metrics and resource savings in several applications at a scale of billions of users.

To our best knowledge, our work has been the first deep RL-based algorithm for the frequency control problem that has been validated by experiments at a scale of billions of users. The rest of the paper is outlined as follows. We first introduce related works. Then, we propose our methodology in learning long-term values and stabilizing frequency control policies using "*Effective Factor*" (*Methodology* Section). We then evaluate our method in several real-world recommendation applications (*Applications and Experiments* Section). Importantly, the proposed methodologies can be easily applied to other frequency control cases with minor modifications.

Related Work

The works most relevant to ours come from large-scale social media platforms (*e.g.*, LinkedIn and Pinterest), which involved constrained optimization balancing between delivery volume and daily metrics (Gupta et al. 2016; Gupta, Liang, and Rosales 2017; Zhao et al. 2018a). Gupta *et al.* frames the frequency control problem as a Multi-Objective Optimization (MOO) problem (Gupta et al. 2016). The objective is to minimize the expected delivery volume as the sum of individual emails' delivery probabilities. The objective comes with the constraints mandating that positive/negative user experience should be above/below certain thresholds. Later, they extended the constraints to include sitewide engagement, such as the total number of active users on a platform (Gupta, Liang, and Rosales 2017). The solution can be obtained from a large-scale quadratic programming solver. However, their formulation indicates a strong independence assumption between emails, which is unable to represent the sequential effect of recommendation delivery in real-world applications. Zhao *et. al* proposed another constrained optimization technique which relaxes the independence assumption between emails (Zhao et al. 2018a). Under a total volume constraint, they searched for the best frequency f for each user u that contributes most $p(a|u, f)$, the conditional probability of u being active. $p(a|u, f)$ is directly predicted by supervised learning models based on user features and the delivery frequency. However, it remains as heuristics as for what constitutes the user activeness label. More importantly, the delivery decisions are still made independently, without considering that a sequence of decisions could change daily metrics differently .

Reinforcement learning (RL) has been successful in learning policies for sequential decision problems in various domains such as video games (Mnih et al. 2015a; Vinyals et al. 2019), board games (Silver et al. 2016), real-time bidding (Wu et al. 2018), and robotics (Rajeswaran et al. 2017). Its strength of capturing long-term effect also attracts researchers working on recommender systems to use RL for retrieving engaging items (Zhao et al. 2018b; Zheng et al. 2018; Cai et al. 2017). In the notification domain, the reinforcement learning approach has achieved success recently (Li 2019), in a similar motivation of viewing recommendation as a sequential decision problem. Gauci *et al.* reported successful applications of an RL-based policy in the notification sending framework to improve daily metrics (Gauci et al. 2018). However, the application exclusively focused on making send/drop decisions for individual notifications but did not consider how a learned policy would interact with infrastructure resources such as delivery volume.

There is a relevant branch of research on frequency capping for advertisements (Feldman et al. 2009; Farahat 2009; Shanahan and den Poel 2010). The goal of frequency capping for an ads server is to provide a supply-demand matching between advertisers who want to avoid repeated displays to the same user and users who arrive sequentially in an unknown pattern. However these studies share a different goal than us in that they put caps on frequencies per user rather than on the global delivery volume of a system.

Methodology

Problem Definition

As we stated in *Introduction*, the core idea of frequency control is to keep a right pace of recommendation delivery in order to maintain accumulated daily metrics as well as minimize resource consumption. While increasing delivery frequency can usually boost short-term product metrics, it is at the cost of infrastructure resource consumption and long-term user experience. Therefore, frequency control should be considered as a sequential decision problem where product delivery at a series of frequencies determines overall user experience and resource consumption together. We formalize the frequency control problem as a Markov Decision Process (MDP) (Bellman 1957), a classic framework for solving sequential decision problems. Specifically, the MDP is designed with the following ingredients:

- state space $\mathcal{S}$, which defines user features such as the user's profile and historical interactions with the product.
- action space $\mathcal{F}$, which defines the possible frequencies to deliver recommendation. In this paper, we focus on a finite action space with a set of predefined frequencies. Recommendation will be delivered at one of these predefined frequencies each time a decision is due.
- reward function $R : \mathcal{S} \times \mathcal{F} \rightarrow \mathbb{R}$, which denotes the quantitative measurement of daily metrics and resource consumption obtained by delivering recommendation $f \in \mathcal{F}$ times to the user with $s \in \mathcal{S}$. In our work, we use a linear reward function: $R(s, f) = m_u(s) - \epsilon f$. It encourages daily metrics $m_u(s)$ while penalizing linearly with the delivery frequency because we want to keep down resource consumption.
- transition function $T : \mathcal{S} \times \mathcal{F} \rightarrow \mathcal{S}$, which reflects how the user state would adapt after recommendation is delivered at frequency f to the user.

The optimal frequency control policy maximizes the accumulated discounted rewards since each user state s_t:

$$\pi^* = argmax_\pi \, \mathbb{E}_{s_i, f_i \sim \pi} \left[\sum_{i=t}^{\infty} \gamma^{i-t} R(s_i, f_i) \right], \quad (1)$$

where γ is a discount factor which balances the focus of the policy between near-term and longer-term rewards.

Deep Q-Network for Value Learning

Q-learning is a classic RL algorithm in which it seeks for π^* by first learning the state-action value, $Q^*(s_t, f_t)$:

$$Q^*(s_t, f_t) = max_\pi Q^\pi(s_t, f_t) \quad (2)$$

$$Q^\pi(s_t, f_t) = R(s_t, f_t) + \mathbb{E}_{s_i, f_i \sim \pi} \left[\sum_{i=t+1}^{\infty} \gamma^{i-t} R(s_i, f_i) \right] \quad (3)$$

$Q^*(s_t, f_t)$ is the maximal possible accumulated rewards that could be obtained by any policy. Intuitively, $Q^*(s_t, f_t)$ measures how promising delivering recommendation with f_t frequency to a user s_t would lead to the best long-term gain. Therefore, we treat $Q^*(s_t, f_t)$ as a good proxy of the long-term value of f_t for the user state s_t.

One can use the learned $Q^*(s_t, f_t)$ values to derive the optimal policy equivalent to Eqn. 1:

$$\pi^*(s_t) = argmax_{f_t} Q^*(s_t, f_t) \quad (4)$$

Q-learning uses the Bellman update to learn the Q^* function (with α as the learning rate):

$$\hat{Q}(s_t, f_t) = (1 - \alpha)\hat{Q}(s_t, f_t) + \alpha \left(R(s_t, f_t) + \max_{f_{t+1}} \hat{Q}(s_{t+1}, f_{t+1}) \right) \quad (5)$$

Since our state space contains real-world user features which can be high-dimensional, we choose to train with Deep Q-Network (DQN) (Mnih et al. 2015b) with deep neural networks as function approximation of $\hat{Q}(\cdot, \cdot)$. We use standard training techniques to train the Double Q-Learning (Hasselt 2010) such as Dueling Networks (Wang et al. 2016) in order to make learning easier and more stable (Gauci et al. 2018).

Effective Factor

Although in theory we can use a trained DQN and Eqn. 4 as a serving policy, it often results to jittering global delivery volume as real-world environments are shifting dynamically. The shifts in user behaviors or system resource usage are generally slow but still significant. We hypothesize the shifts come from temporal effects (*e.g.*, more traffic on the weekend or holidays than a weekday) and internal changes (*e.g.*, changes in the upstream/downstream services, throttles/constraints newly enforced into the system). In our real-world preliminary tests we found that the global delivery volume could fluctuate by a low but significant level ($10 \sim 20\%$) to trigger alerts from in-house monitoring systems.

While companies would like to see long-term gain in daily metrics and resource consumption, they also hope to stabilize global delivery volume. Recurrent training cannot help RL models catch up the latest user/system behavior shift because RL models train on user history data spanning a few weeks. It is also not easy to massage the reward function, particularly ϵ which we use to control the impact of resource consumption, to get a stable global delivery volume. Because the relationship between recommendation frequency and global resource consumption may not be easily defined as a linear function.

We propose a simple yet robust technique called "*Effect Factor*" on top of Q-values to stabilize global delivery volume in real time. The technique works for our cases where the action space has at least 3 predefined frequencies[1]. First, We define the maximal incremental value of Q-values of a state as:

$$\Delta Q^*(s) = max_{f'} Q^*(s, f') - min_{f'} Q^*(s, f') \quad (6)$$

Then, we define an adjustable scalar $EF \in [0, 1]$ and a frequency control policy based on EF and $\Delta Q^*(s)$:

$$\pi^{EF}(s) = \min \left\{ f \mid Q^*(s, f) \geq min_{f'} Q^*(s, f') + EF \cdot \Delta Q^*(s) \right\}, \quad (7)$$

where $\pi^{EF}(s)$ can be intuitively explained as finding the minimal frequency that achieves a sufficient level of the maximal incremental value.

We observed in practice that tuning EF has a predictable effect on daily metrics and resource consumption: for most states, as f increases, $Q^*(s, f)$ would be monotonically increasing, monotonically decreasing, or vary with a bell shape (See Fig. 1 for example). Although adjusting EF

[1]When the action space has only two possible actions, the practitioner can design a simpler heuristic.

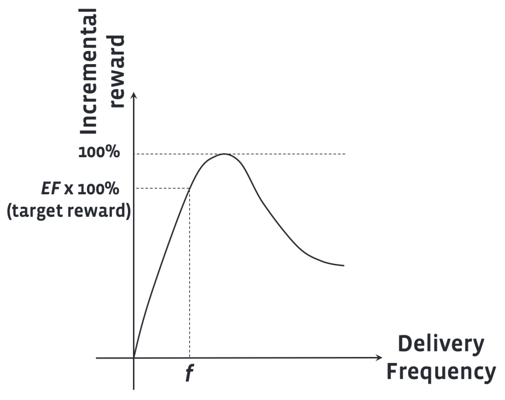

Figure 1: Illustration of the *Effect Factor* technique. The incremental reward (y-axis) by the selection of various delivery frequency (x-axis) is quantified by $[Q^*(s,f) - min_{f'} Q^*(s,f')] / \Delta Q^*(s)$. The returned delivery frequency for each user in the online serving is the minimal frequency that achieves $min_{f'} Q^*(s,f') + EF \cdot \Delta Q^*(s)$.

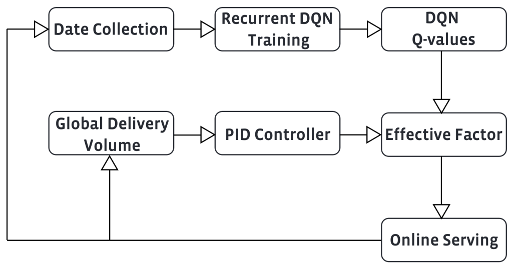

Figure 2: System Overview. The system consists of (1) a data pipeline to collect RL training data; (2) a recurring training pipeline of DQN models on a daily basis; (3) an effect factor controller, which reads predefined configurations, monitors real-time global volume, and adjusts dynamically; and (4) an online serving system. In the RL application, the clients provide either a fixed value of effect factor or a predefined target delivery volume. If latter, a PID controller will be applied to adjust the effect factor in online serving automatically.

could result to slow change for daily metrics and resource consumption of a single user, from a global view adjusting EF brings very immediate and monotonic change in terms of the overall delivery volume.

Since EF is a responsive and monotonic knob for tuning the global delivery volume, we employed a proportional-integral-derivative (PID) controller (Åström and Hägglund 1995) to dynamically adjust EF in order to stabilize the global delivery volume. To guide directions and magnitudes of adjustment, PID controllers calculate the proportional, integral, and derivative of the differences between a target delivery volume and actual delivery volumes. To run the PID controllers, a client team will first set a global target delivery volume in a configuration file. We have a service running continuously to aggregate and store the actual global delivery volume. A recurring job is periodically (*e.g.*, per 10-min) executed to check the difference between the actual and target delivery volume, based on which EF is dynamically adjusted. We observed that PID controllers worked as expected: whenever the actual global delivery volume exceeds the expected volume, PID controllers will lower the EF; if the actual volume goes in the opposite direction, PID controllers will increase the EF instead. We can also assign different values of EF for different user cohorts based on product needs.

It is worth noting that although the EF-intervened policy would deviate from the optimal policy specified in Eq. 4, EF is set in practice to cause only very mild deviation, and we can still see gains in metrics we would like to lift.

Applications and Experiments

We built a personalized frequency control system to support our RL-based methodology described in the *Methodology* Section. An overview of our system can be seen in Figure 2. We use the "notification scheduling" domain as a test bed for our RL-based methodology. We believe our method can be extended to other applications that need delivery frequency control in general. Notifications is an important recommendation channel in social network products to deliver missed information to users. While some notifications are triggered by user activities (*e.g.*, direct messages), we focus on those scheduled daily to target users for promotional purposes (*e.g.*, reminding users of missed content or suggesting new friends to connect). The notifications are scheduled in advance (*e.g.* one day ahead), at which time we need to determine how many notifications to schedule on the delivery day. Our scheduling frequency ranking platform is only responsible for determining the frequency of notification scheduling. The actual content of scheduled notifications is determined by downstream rankers, which can be seen as part of the MDP environment of which the frequency control policy has no direct control.

Training

To have a comprehensive evaluation of our methodology, we tested on five different notification types which serve different products and run in different channels such as email and mobile push. For business reasons, we will not reveal exact notification types but denote them as $notif\_type_1, \ldots, notif\_type_5$, each of which reaches billion-scale users. We allocated a 0.5% user segment traffic for training data collection. Whenever a scheduling decision was due for a user, one of six predefined frequencies was chosen to apply (*i.e.*, $\mathcal{F} = \{f_1, f_2, \ldots, f_6\}$). We joined data over user identities such that each user's complete scheduling history (30 tuples of $(s_t, f_t, R(s_t, f_t), s_{t+1})$) are included in the final training data. There are totally 50 ranking features used for the model training and prediction, including user-level features and user-notification interaction features. We used an open-source applied reinforcement learning platform *ReAgent* to train the DQN model (Gauci et al. 2018).

The hyperparameters (*e.g.*, neural network size, learning rate, *etc.*) of the DQN model are hand-picked without too much fine-tuning. The discount factor γ is tuned in the range from 0.25 to 0.9 based on the online experiment results. The daily metric measurement in the reward, $m_u(s)$, is itself a linear function of two different daily metrics. The linear weights of the daily metrics and ϵ appeared in the reward function, as well as the target global delivery volume, were hand-tuned after 2 to 3 experiment iterations of online A/B tests to achieve accepted trade-offs between daily metrics and resource consumption. In the first iteration, we began with $\epsilon = 0.005 \times daily\ metrics$ and searched its neighborhood. In the next iterations, we searched with greater granularity around the ϵ that gave us the most acceptable trade-off from the previous iterations. The agreement of trade-offs was reached by discussing with other stakeholders such as product managers.

Results

The online experiments in this study were performed on the 2-4% (around 50 million users for each notification type) of the whole monthly active users ($\sim$ 2.5 billion). We compared the performance between our RL-based approach and the current rule-based frequency control product policy. We observed that the RL-based approach has a significantly better efficiency ratio (=daily metrics / delivery volume) than a human-crafted rule-based approach across all the five notification types (Table 1). In addition, we tracked the two daily metrics included in $m_u(s)$ and the delivery volume, which showed consistent trends over time (See Figure 3 for such analysis from one notification type). This shows the possibility that we can improve individual metrics collectively by a linear reward shaping. In addition, we observed that reward functions without ϵ penalty results in lower delivery efficiency ($daily\ metrics\ /\ delivery\ volume$) and we have to end the experiment prematurely to avoid further negative impact.

To have a better understanding of our methodology, we made a case study on one chosen notification type with data tracked over 30 days. The global delivery volume during the experiment period is stable (not shown). Figure 4 shows the frequency distribution, which exhibited small fluctuation for each frequency. At the user level, we found that around 40% users always received the same predicted frequency number and the other users received fluctuated predicted frequency

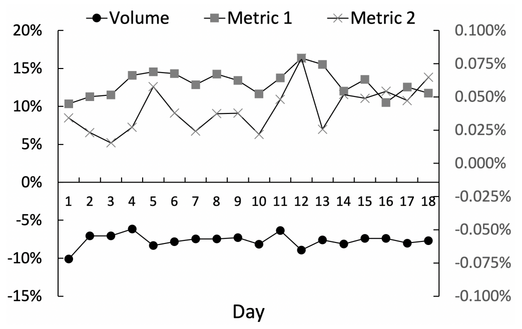

Figure 3: Time series of schedule volume and two daily metrics. The latter two are linearly combined in $m_u(s)$ in the reward function.

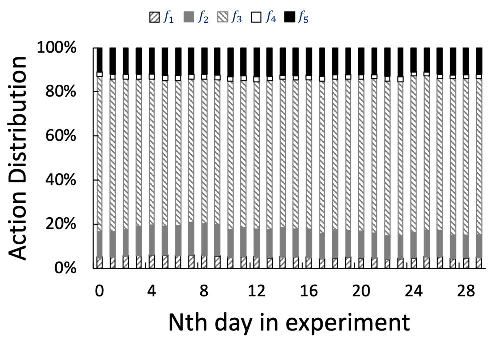

Figure 4: Global frequency distribution over the experiment period.

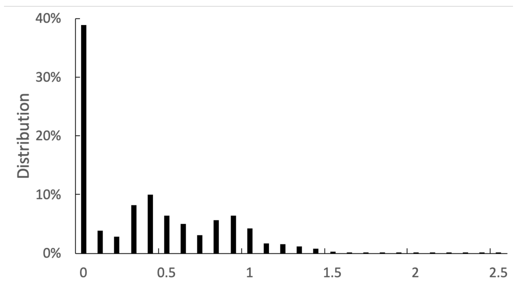

Figure 5: Standard deviation of frequencies per user over the 30 days. Around 40% users received a constant frequency while the rest received varied frequencies under our policy.

	$notif\_type_1$	$notif\_type_2$	$notif\_type_3$	$notif\_type_4$	$notif\_type_5$
delivery volume	-5.0%	-7.6%	+0.9%	-5%	+6.3%
daily metrics	+ 6.6%	+ 11.7%	+17.2%	Neutral	+12.3%
efficiency ratio	+12%	+19%	+16%	+5%	+6%

Table 1: Performance of our methodology compared to the current production setting. It achieves higher efficiency ratio (=daily metrics / delivery volume) in all notification types.

Activity cohort	High	Medium	Low
Delivery volume	-20.6%	-15.9%	-10.8%
Daily Metric 1	+12.3%	+16.5%	+3.8%
Daily Metric 2	+0.0419%	+0.0297%	+0.1089%

Table 2: User cohort analysis for one notification type. Results are compared to the current production policy.

over time, as indicated by the standard deviation of received frequencies per user (Figure 5). Breaking down users into three cohorts (highly, medium, and lowly active) according to their profiles *during* the experiment, we found reduction in the sending volume and improvement in individual daily metrics (the two used in the reward function) in all user cohorts (Table 2).

We also investigated the characteristics of EF. We found that once client teams set a reasonable target global delivery volume, EF were dynamically adjusted by PID controllers mostly in the range from 0.75 to 1.0, which indicates that the EF-modified policy is still close to the optimal policy learned by DQN.

Conclusion

In this paper, we proposed a methodology for the frequency control problem, combining long-term value learning by deep reinforcement learning and a global delivery volume control technique termed *Effective Factor*. The experiment results demonstrate that the proposed approach is able to improve daily metrics effectively while reducing infrastructure resource consumption in the notifications domain at the industrial scale. To our best knowledge, our work represents the first industrial application of deep reinforcement learning in the frequency control problem at a scale of billions of users.

Acknowledgments

We thank Yuankai Ge and Jason Gauci's support in this project. We thank all partners and client teams' collaboration and contributions.

References

Åström, K. J.; and Hägglund, T. 1995. *PID controllers: theory, design, and tuning*, volume 2. Instrument society of America Research Triangle Park, NC.

Bellman, R. 1957. A Markovian decision process. *Journal of mathematics and mechanics* 679–684.

Cai, H.; Ren, K.; Zhang, W.; Malialis, K.; Wang, J.; Yu, Y.; and Guo, D. 2017. Real-time bidding by reinforcement learning in display advertising. In *Proceedings of the Tenth ACM International Conference on Web Search and Data Mining*, 661–670.

Farahat, A. 2009. Privacy preserving frequency capping in internet banner advertising. In *Proceedings of the 18th international conference on World wide web*, 1147–1148.

Feldman, J.; Mehta, A.; Mirrokni, V.; and Muthukrishnan, S. 2009. Online stochastic matching: Beating 1-1/e. In *2009 50th Annual IEEE Symposium on Foundations of Computer Science*, 117–126. IEEE.

Gauci, J.; Conti, E.; Liang, Y.; Virochsiri, K.; He, Y.; Kaden, Z.; Narayanan, V.; Ye, X.; Chen, Z.; and Fujimoto, S. 2018. Horizon: Facebook's open source applied reinforcement learning platform. *arXiv preprint arXiv:1811.00260* .

Gupta, R.; Liang, G.; and Rosales, R. 2017. Optimizing Email Volume For Sitewide Engagement. In *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management*, 1947–1955.

Gupta, R.; Liang, G.; Tseng, H.-P.; Holur Vijay, R. K.; Chen, X.; and Rosales, R. 2016. Email volume optimization at LinkedIn. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 97–106.

Hasselt, H. V. 2010. Double Q-learning. In *Advances in neural information processing systems*, 2613–2621.

Iqbal, S. T.; and Horvitz, E. 2010. Notifications and awareness: a field study of alert usage and preferences. In *Proceedings of the 2010 ACM conference on Computer supported cooperative work*, 27–30.

Li, Y. 2019. Reinforcement learning applications. *arXiv preprint arXiv:1908.06973* .

Ma, H.; Liu, X.; and Shen, Z. 2016. User fatigue in online news recommendation. In *Proceedings of the 25th International Conference on World Wide Web*, 1363–1372.

Mnih, V.; Kavukcuoglu, K.; Silver, D.; Rusu, A. A.; Veness, J.; Bellemare, M. G.; Graves, A.; Riedmiller, M.; Fidjeland, A. K.; Ostrovski, G.; et al. 2015a. Human-level control through deep reinforcement learning. *nature* 518(7540): 529–533.

Mnih, V.; Kavukcuoglu, K.; Silver, D.; Rusu, A. A.; Veness, J.; Bellemare, M. G.; Graves, A.; Riedmiller, M.; Fidjeland, A. K.; Ostrovski, G.; et al. 2015b. Human-level control through deep reinforcement learning. *nature* 518(7540): 529–533.

Rajeswaran, A.; Kumar, V.; Gupta, A.; Vezzani, G.; Schulman, J.; Todorov, E.; and Levine, S. 2017. Learning complex dexterous manipulation with deep reinforcement learning and demonstrations. *arXiv preprint arXiv:1709.10087* .

Shanahan, J.; and den Poel, D. 2010. Determining optimal advertisement frequency capping policy via Markov decision processes to maximize click through rates. In *Proceedings of NIPS Workshop: Machine Learning in Online Advertising*, 39–45.

Silver, D.; Huang, A.; Maddison, C. J.; Guez, A.; Sifre, L.; Van Den Driessche, G.; Schrittwieser, J.; Antonoglou, I.; Panneershelvam, V.; Lanctot, M.; et al. 2016. Mastering the game of Go with deep neural networks and tree search. *nature* 529(7587): 484–489.

Sutton, R. S.; and Barto, A. G. 2018. *Reinforcement learning: An introduction*. MIT press.

Vinyals, O.; Babuschkin, I.; Czarnecki, W. M.; Mathieu, M.; Dudzik, A.; Chung, J.; Choi, D. H.; Powell, R.; Ewalds, T.; Georgiev, P.; et al. 2019. Grandmaster level in StarCraft II using multi-agent reinforcement learning. *Nature* 575(7782): 350–354.

Wang, Z.; Schaul, T.; Hessel, M.; Hasselt, H.; Lanctot, M.; and Freitas, N. 2016. Dueling network architectures for deep reinforcement learning. In *International conference on machine learning*, 1995–2003.

Wu, D.; Chen, X.; Yang, X.; Wang, H.; Tan, Q.; Zhang, X.; Xu, J.; and Gai, K. 2018. Budget constrained bidding by model-free reinforcement learning in display advertising. In *Proceedings of the 27th ACM International Conference on Information and Knowledge Management*, 1443–1451.

Zhao, B.; Narita, K.; Orten, B.; and Egan, J. 2018a. Notification Volume Control and Optimization System at Pinterest. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 1012–1020.

Zhao, X.; Xia, L.; Zhang, L.; Ding, Z.; Yin, D.; and Tang, J. 2018b. Deep reinforcement learning for page-wise recommendations. In *Proceedings of the 12th ACM Conference on Recommender Systems*, 95–103.

Zheng, G.; Zhang, F.; Zheng, Z.; Xiang, Y.; Yuan, N. J.; Xie, X.; and Li, Z. 2018. DRN: A deep reinforcement learning framework for news recommendation. In *Proceedings of the 2018 World Wide Web Conference*, 167–176.