BLOB : A Probabilistic Model for Recommendation that Combines Organic and Bandit Signals

Otmane Sakhi, Stephen Bonner, David Rohde, Flavian Vasile

BLOB : A Probabilistic Model for Recommendation that Combines Organic and Bandit Signals

Otmane Sakhi
Criteo AI Lab
Paris, France
o.sakhi@criteo.com

Stephen Bonner
Department of Computer Science, Durham University
Durham, UK
s.a.r.bonner@durham.ac.uk

David Rohde
Criteo AI Lab
Paris, France
d.rohde@criteo.com

Flavian Vasile
Criteo AI Lab
Paris, France
f.vasile@criteo.com

ABSTRACT

A common task for recommender systems is to build a profile of the interests of a user from items in their browsing history and later to recommend items to the user from the same catalog. The users' behavior consists of two parts: the sequence of items that they viewed without intervention (the organic part) and the sequences of items recommended to them and their outcome (the bandit part). In this paper, we propose *Bayesian Latent Organic Bandit model (BLOB)*, a probabilistic approach to combine the 'organic' and 'bandit' signals in order to improve the estimation of recommendation quality. The bandit signal is valuable as it gives direct feedback of recommendation performance, but the signal quality is very uneven, as it is highly concentrated on the recommendations deemed optimal by the past version of the recommender system. In contrast, the organic signal is typically strong and covers most items, but is not always relevant to the recommendation task. In order to leverage the organic signal to efficiently learn the bandit signal in a Bayesian model we identify three fundamental types of distances, namely action-history, action-action and history-history distances. We implement a scalable approximation of the full model using variational auto-encoders and the local re-paramerization trick. We show using extensive simulation studies that our method out-performs or matches the value of both state-of-the-art organic-based recommendation algorithms, and of bandit-based methods (both value and policy-based) both in organic and bandit-rich environments.

CCS CONCEPTS

• **Computing methodologies** → **Bayesian network models**; **Learning from implicit feedback**; • **Information systems** → **Recommender systems**.

KDD '20, August 23–27, 2020, Virtual Event, USA

ACM ISBN 978-1-4503-7998-4/20/08…$15.00
https://doi.org/10.1145/3394486.3403121

KEYWORDS

Latent variable models; Bayesian inference; Recommender systems

ACM Reference Format:
Otmane Sakhi, Stephen Bonner, David Rohde, and Flavian Vasile. 2020. BLOB : A Probabilistic Model for Recommendation that Combines Organic and Bandit Signals. In *26th ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD '20), August 23–27, 2020, Virtual Event, USA.* ACM, New York, NY, USA, 12 pages. https://doi.org/10.1145/3394486.3403121

1 INTRODUCTION

The recommender systems literature is somewhat bifurcated into two distinct branches. One branch concerns analysing logs of organic user sessions where similar items co-occur [1, 13, 20, 23]. A distinguishing feature of this research is that it focuses on logs of organic user sessions where users view variable numbers of (usually) related items in a shopping session.

A second branch of research explicitly (and entirely) focuses on the logs of the recommender system using the history of successful and unsuccessful recommendations in order to discover a good recommender system policy. This branch uses off policy learning in order to discover new policies with good actions [3, 6, 38]. This work is distinguished by its use of recommender system logs for training and its anonymous feature vector (usually called the context).

The purpose of this paper is twofold. Firstly, we pose a simple yet powerful model that combines these two distinct data sources in order to efficiently learn good recommendation policies. Secondly, we develop a fully probabilistic approach to recommendation and outline its benefits and consequences. The probabilistic formulation gives insights into user embedding creation and the alternative frameworks of value and policy learning.

The remainder of the paper is structured as follows: In Section 2 we introduce our probabilistic model of organic and bandit behaviour and discuss its properties. In Section 3 we describe the training of the model. In section 4 we apply our model to the RecoGym simulator [15, 33] and present results. Concluding remarks are made in Section 5.

2 PROBABILISTIC MODEL OF ORGANIC AND BANDIT SESSIONS

We develop a simple probabilistic model that allows us to build a representation of a user from a variable length organic sequence of items and then predict accurately how probable the user is to respond positively to each recommendation in the catalog.

Throughout this paper, we will make use of the notation introduced in Table 1. We use u to denote a user or a session, we use t time to denote sequential time and v to denote which product they viewed from 1 to P where P is the number of products. User u will also be given some recommendations (or actions) $a_{u,1}, ..., a_{u,n}$ again which can take values from 1 to P and we will observe a reward (or a click) for each of these recommendations $c_{u,1}, ..., c_{u,n}$. The organic part of the session are the items the user views without any encouragement from the recommender system i.e. $v_{u,1}, ...v_{u,T_u}$, the bandit part of the session refers to the recommender system log: $a_{u,1}, ..., a_{u,n_u}; c_{u,1}, ..., c_{u,n_u}$. Thus, the size of the organic dataset is U, the number of users, and the bandit dataset size is $\sum_u n_u = N$. We drop the u subscript and treat the bandit dataset as records with $n \in [1, ..., N]$.

In our model, the user's interest is described by a K dimensional variable $\boldsymbol{\omega}_u$ which can be interpreted as the user's interest in K topics. We then assume the following generative process for the organic views in each session:

$$\boldsymbol{\omega}_u \sim \mathcal{N}(\mathbf{0}_K, \boldsymbol{I}_K), \quad v_{u,1}, .., v_{u,T_u} \sim \text{categorical}(\text{softmax}(\boldsymbol{\Psi}\boldsymbol{\omega}_u + \boldsymbol{\rho}))$$

The organic embedding matrix $\boldsymbol{\Psi}$ is $P \times K$ and represents information about how items correlate in a users session organically (i.e. without any intervention from the recommender system). The P dimensional vector $\boldsymbol{\rho}$ is related to the items organic popularity.

Once this session is generated a recommendation or actions is made to user u denoted a_u and a reward or click will be observed c_u.

$$c_u | a_u, \boldsymbol{\beta}, \boldsymbol{\omega}, \boldsymbol{\kappa} \sim \text{Bernoulli}\{\text{sigmoid}(\boldsymbol{\beta}_{a_u}\boldsymbol{\omega}_u + \boldsymbol{\kappa}_{a_u})\}$$

The bandit embedding matrix $\boldsymbol{\beta}$ is $P \times K$ and represents information about how to personalise recommendations to a user u with a latent user representation $\boldsymbol{\omega}_u$.
The organic behavior is parameterized by $\boldsymbol{\Psi}, \boldsymbol{\rho}$ and the bandit behavior is parameterized $\boldsymbol{\beta}, \boldsymbol{\kappa}$ in order to relate the two we use the following matrix variate prior distribution of $\boldsymbol{\beta}$:

$$\boldsymbol{\beta} | \boldsymbol{\Psi} \sim \mathcal{MN}(s^+(w_a)\boldsymbol{\Psi}, s^+(w_b)\boldsymbol{\Psi}\boldsymbol{\Psi}^T, s^+(w_b)\frac{1}{P}\boldsymbol{\Psi}^T\boldsymbol{\Psi}).$$

Where $\mathcal{MN}(\cdot)$ is the matrix variate normal distribution[1] We will show how each of the three terms in the matrix variate normal allow us to include in our model one of the three fundamental differences of recommendation. The softplus function is defined:

$$s^+(w) = \log\{1 + \exp(w)\}.$$

We also put a prior on $\boldsymbol{\kappa}$ which is $P \times 1$:

$$\boldsymbol{\kappa} \sim \mathcal{N}(w_c, \boldsymbol{I}_P\sigma_\kappa^2).$$

[1]The matrix normal distribution can be defined by its connection to the multivariate normal. If $\boldsymbol{\beta} \sim \mathcal{MN}(\boldsymbol{M}, \boldsymbol{R}, \boldsymbol{S})$, where mean matrix $\boldsymbol{M}$ is $M \times N$, and $\boldsymbol{R}$ is $M \times M$ and $\boldsymbol{S}$ is $N \times N$ - then: $\text{vec}(\boldsymbol{\beta}) \sim \mathcal{N}(\text{vec}(\boldsymbol{M}), \boldsymbol{R} \otimes \boldsymbol{S})$. In this way the matrix variate normal has a more compact and restricted representation of the co-variance than the matrix variate normal. Here $\otimes$ denotes the Kronecker product.

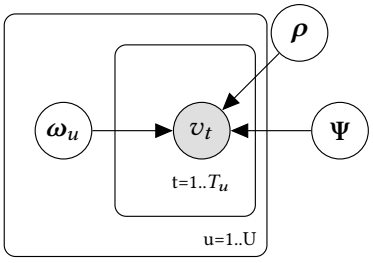

Figure 1: A graphical model of the organic behavior.

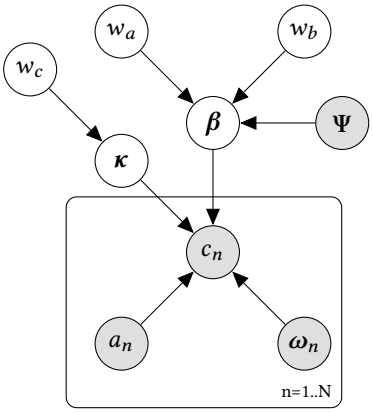

Figure 2: A graphical model of the bandit behavior.

The hyper-parameters w_a, w_b, w_c are also given normal priors:

$$w_a \sim \mathcal{N}(\mu_{w_{a_0}}, \sigma^2_{w_{a_0}}), \quad w_b \sim \mathcal{N}(\mu_{w_{b_0}}, \sigma^2_{w_{b_0}}), \quad w_c \sim \mathcal{N}(\mu_{w_{c_0}}, \sigma^2_{w_{c_0}}).$$

In this paper we will mostly consider the organic and bandit behavior as separate but related processes. A graphical model defining the organic portion of the model is given in Figure 1. This graphical model has a similar structure to the latent Dirichlet Allocation model (LDA) [5], the difference being that where we model $v \sim \text{categorical}\{\text{softmax}(\boldsymbol{\Psi}\boldsymbol{\omega} + \boldsymbol{\rho})\}$, LDA uses $v \sim \text{categorical}(\boldsymbol{\Psi}\boldsymbol{\omega})$ putting simplex constraints on $\boldsymbol{\Psi}$ and $\boldsymbol{\omega}$, similarly correlated topic models [21] use $v \sim \text{categorical}\{\boldsymbol{\Psi}\text{softmax}(\boldsymbol{\omega})\}$ where the simplex constraint is only on $\boldsymbol{\Psi}$. This model can also be viewed as a linear version of the Multi-VAE [23].

We will show that using variational autoencoders with the re-parameterization trick is an effective way to train the organic model.

The approach developed in this paper takes the organic model and estimates $\boldsymbol{\Psi}$ by maximum likelihood and $\boldsymbol{\omega}$ by posteior mean (denoted $\hat{\boldsymbol{\omega}}$) and then treats $\boldsymbol{\Psi}$ and $\hat{\omega}$ as observed in the bandit model. The graphical model is shown in Figure 2. In this probabilistic model we will develop full Bayesian inference of the $\boldsymbol{\beta}$, $\boldsymbol{\kappa}$, w_a, w_b and w_c. This is important because the bandit signal is very uneven. Lots of information is available on past actions that the previous recommender system favoured and little information or no information is available on many other actions, meaning that the posterior is tight in some regions but broad and highly influenced by the prior in others. We use variational approximations and the local re-parameterization trick in order to capture this complex structure.

Symbol	Dimension	Description
u	Scalar	A given user's id.
t	Scalar	sequential time.
P	Scalar	Total number of products.
K	Scalar	The size of the embedding.
$v_{u,t}$	Scalar	Product id for user u at time t.
$\boldsymbol{\omega}_u$	$K \times 1$	A given user's state.
$\boldsymbol{\Psi}$	$P \times K$	Organic embedding matrix.
$\boldsymbol{\Psi}_v$	$1 \times K$	Organic embedding for v.
$\boldsymbol{\beta}$	$P \times K$	Bandit embedding matrix.
$\boldsymbol{\beta}_v$	$1 \times K$	Bandit embedding for v.
$\boldsymbol{\rho}$	$P \times 1$	Item popularity intercept.
$\boldsymbol{\kappa}$	$P \times 1$	Item recommendability intercept.
T_u	Scalar	Session length for u.
N	Scalar	The size of the Bandit dataset.
U	Scalar	The number of user sessions.

Table 1: Notations and Definitions

We refer to the organic only component of the model as **BLO** (Bayesian Latent Organic) model (we apply maximum likelihood to $\boldsymbol{\Psi}, \boldsymbol{\rho}$ and integrate $\boldsymbol{\omega}$). The full model is referred to as **BLOB** (Bayesian Latent Organic Bandit Model).

2.1 Intuition for the model

The model presented embodies a fundamental implicit assumption in the traditional recommendation system, the assumption that auto-completion of a session results in good recommendations being made. This is one of the three fundamental distances of recommendation, the action-history distance.

2.1.1 The implicit assumption in traditional recommendation: good recommendations are (usually) similar to the items in the user's history. Algorithms in the recommendation literature look at items in a user's history and attempt to predict the final element in this session. The fraction of times that the predicted item is within the top K items in a held out data set is a key metric called precision@K that measures a models ability to "auto-complete" a users behavior. The organic performance is therefore computed:

$$P(v_{u,T_u}|v_{u,1},..,v_{u,T_u-1}).$$

Metrics such as NDCG, recall@K or log likelihood are computed on this auto-completion task.

However auto-completion is not the same as recommendation. In fact to reduce recommendation to auto-completion removes the opportunity for a recommender system to help a user discover new things which arguably is the primary objective of recommendation. That said, organic data is usually plentiful and this implicit assumption that recommendation as auto-completion certainly has some merit. We can state this assumption as, if:

$$P(V_{u,T_u} = v_a|v_{u,1},..,v_{u,T_u-1}) > P(V_{u,T_u} = v_b|v_{u,1},..,v_{u,T_u-1})$$

Then item v_a is probably better than item v_b as a recommendation i.e the following holds with high probability:

$$P(c = 1|A = v_a, v_{u,1},..,v_{u,T_u-1}) > P(c = 1|A = v_b, v_{u,1},..,v_{u,T_u-1})$$

Although this relationship often holds, it need not hold in every single instance. Maybe the user already knows about item v_a, maybe the recommendation for v_a is unattractive or maybe the reason the user never visited item v_b is lack of knowledge and it is actually a very valuable recommendation. We want our recommender system to make use of the organic relationship, but we also want to learn from the logs of the recommender system itself which records if the recommendations that we chose to deliver were successful or not. This "bandit feedback" is in some sense the true arbiter of if a recommendation is good or not, but the bandit signal is usually highly concentrated around what the previous version of the recommendation system judged to be a good recommendation, so it cannot reliably be used over the entire recommendation space. For example the organic session might contain information that two products (say) rice and a phone are rarely viewed together in the same organic session. However it probably will not contain many events where a phone is recommended to a user with rice in their history. If the recommender system is to infer that this is likely a poor recommendation, it must do so through a prior linking the bandit behavior to the organic behavior.

When deployed in a production recommender system the model operates in the following way. First a posterior over a user embedding is approximately calculated:

$$P(\boldsymbol{\omega}_u|v_{u,1},...,v_{u,T_u},\boldsymbol{\Psi},\boldsymbol{\rho})$$

A fast variational approximation can be made of $\boldsymbol{\omega}_u \sim \mathcal{N}(\boldsymbol{\mu}_{\omega_q}, \Sigma_{\omega_q})$ which gives both a mean and a variance (this can be done using either a variational EM algorithm or a variational autoencoder).

For our purposes we make the pragmatic compromise that we can summarise the user history with a posterior mean point estimate $\hat{\boldsymbol{\omega}} = \boldsymbol{\mu}_{\omega_q}$, this prevents numerical integration of $\boldsymbol{\omega}_u$ at recommendation time. Once this compromise is made it also makes sense to train the organic and bandit components separately. The probability of a click is given by:

$$P(c|\hat{\boldsymbol{\omega}},\boldsymbol{\beta},\boldsymbol{\kappa},a) = \text{sigmoid}(\boldsymbol{\beta}_a\hat{\boldsymbol{\omega}} + \boldsymbol{\kappa}_a)$$

The recommender system will then choose a recommendation that will optimise this reward (or a combination of reward and exploration - but the explore-exploit dilemma [22] is beyond the scope of this paper.

The organic parameters $\boldsymbol{\Psi}$ and $\boldsymbol{\rho}$ are not required in order to deliver a recommendation. They are used only to put a prior on the bandit embeddings.

We note parenthetically that due to the fact that once the user embedding $\hat{\boldsymbol{\omega}}$ is created the model is linear and we can exploit fast algorithms to quickly find the optimal recommendation over large catalogues [11, 26].

2.1.2 The organic user session. The organic user session model we propose can be understood in a number of ways. It can be viewed as a user item matrix factorization where the user has a latent interest in K topics - a discussion of this interpretation is given in the supplementary material.

It can also be viewed as an i.i.d. categorical process with a (usually) low-rank multivariate normal prior. The prior causes similar items to co-occur in a session with high probability. Because of this

assumptions seeing an item will always make it more likely to be viewed again. If we had a full rank model the user session would imply the law of large numbers where the next item prediction will converge to the empirical frequency. In practice the session history is short and the embedding size is much lower than the number of products, but the assumption remains that viewing an item makes the conditional probability for that same item increase (also and importantly the conditional probability that similar items will be viewed also increases).

This is a relatively strong assumption compared to powerful sequential models such as recurrent neural networks [13] which can model complex sequences. The simpler and stronger assumption made by BLO is reasonable in many settings and greatly simplifies learning.

2.1.3 The bandit session and the three distances in recommendation. The auto-complete assumptions as embodied in the recommendation research measures the similarity between the recommendation and the items in history. This is the first similarity or distance, the distance between the history and the action. The mean of the matrix normal $\mathbf{\Psi}$ embodies this assumption.

The second similarity in recommendation is the similarity between actions. That is if action a_1 and a_2 are similar then we expect that the responses to these actions to the same (or similar) users be correlated. This distance is encoded with the first (low rank) co-variance $\mathbf{\Psi}\mathbf{\Psi}^T$ in the matrix normal prior on $\boldsymbol{\beta}$.

The third similarity in recommendation is the similarity between users. If user u_1 and u_2 are similar then we expect the response to the same (or similar) action on these users to be correlated. This distance is encoded with the second co-variance $\mathbf{\Psi}^T\mathbf{\Psi}$ in the matrix normal prior on $\boldsymbol{\beta}$.

The effect of the first distance is to seed the recommendation using the organic similarities, the effect of the second and third is to borrow strength allowing the bandit signal to be used more effectively. Finally the parameters w_a and w_b control the strength of the influence of the first and second distance. The relative strength of the first distance and the second is an extremely important hyperparameter.

2.2 Value vs policy learning

The method proposed here is a value based method as it learns the value for every action and then can determine a decision rule using unconstrained optimisation. In this way it differs from alternative methods for learning from bandit feedback that have been recently proposed [3, 6, 38] which use policy learning.

Bayesian methods are inherently value based and bring the benefit of being able to synthesis data sources such as organic and bandit, they also produce uncertainty that is useful for explore-exploit strategies such as upper confidence bound and Thompson sampling [22]. From a purely statistical point of view principles such as the conditionality and the likelihood principle actually forbid the use of the propensity score [2, 12]. Given that training on bandit feedback is sometimes considered to be synonymous with using the inverse propensity score (IPS) it is worth reviewing some advantages of Bayesian value based methods.

It has been shown in [31], that under regularity conditions that apply in the recommendation case, the Bernstein-von Mises theorem applies, and that the Bayesian estimator is efficient $\sqrt{n}$ consistent and necessarily better than the IPS (or Horvitz-Thompson) estimator[2]. However note that a real recommender system log will be of sufficient dimensionality that even with terabytes of logs asymptotic theory is usually not relevant (i.e. priors will have real impacts).

It is also sometimes argued that the IPS score is necessary to apply in counterfactual settings due to the domain shift which occurs in causal settings [16]. However this argument does not apply when the model has enough capacity to accurately predict the value everywhere [37] and there is no need to constrain capacity to reduce estimator variance when applying Bayesian methods [28]. It seems that some of the positive aspects of value based methods have been overlooked due to criticisms that apply only in the non-Bayesian case.

Policy learning also suffers from some draw backs. Policy learning extends the principle of Statistical Learning Theory (SLT) to the counterfactual setting. The idea of SLT is that a decision rule is fit to the historical data from a constrained set. If a decision rule from a restricted set has good performance (low risk) then it is likely to also have low risk on out of sample data [41]. These analyses are based upon treating empirical risk or counterfactual risk as a statistic, but these are highly non-sufficient statistics and there is no ability to order decision rules that have the same empirical risk even when away from the data they are very different. The theory is heavily based on having a restricted set of decision rules, but restricting the set might exclude good decisions. Value based methods make no such restriction.

Extending SLT to the counterfactual setting requires some additional ideas because the consequences of decisions the new policy will make are not available. IPS based methods have been a recent research focus that extend the empirical risk minimisation to the counterfactual setting. Technical challenges are being addressed such as the fact that the variance of the decision rule can vary depending on how much it differs from the historical logging policy [39]. As well as the problem of propensity overfitting i.e. decision rules can achieve an estimated reward of 0 by avoiding past decisions (0 might be good or bad depending on how the reward is defined) causing decision rules either to cling to the old policy or to be driven away from it[3]. It is usually considered a better heuristic for the new policy to cling to the old one.

One simple method to control variance is to cap large weights [6] (necessarily associated with actions that are different to the logging policy). This method controls the bias-variance trade-off. Another method that more explicitly discourages deviation from the logging policy is to apply variance penalization [39] here rather than optimizing the conterfactual risk directly a penalized term is instead optimized, this penalization naturally goes up if the recommendations are rare under the logging policy (and hence have a high IPS weight).

[2] They additionally show that IPS based methods can have better frequentist properties than Bayesian estimators when these regulatory conditions break down.

[3] The self normalized importance sampling variant of IPS is one proposal to remove this sensitivity to the definition of the reward[36]

Many of the standard policy learning settings [4] have the property that the learnt policy will only deviate from the preferred decision of the logging policy in the face of considerable evidence. This is a good heuristic in cases where the logging policy is good, but can be a problem in other situations.

The potential strength of policy based approaches is due to the fact they do not use a model and they focus directly on the decision rule focusing optimisation and capacity on the parts of the problem that matters most. Bayesian value based methods cannot do this because the modelling step is made before and separately to the decision making step.

3 MODEL TRAINING

3.1 Organic session training: learning the organic embeddings

The log likelihood of the organic model has the form:

$$\log p(v_1, .., v_T, \boldsymbol{\omega}_u|\boldsymbol{\Psi}) = \left(\sum_t^T \boldsymbol{\Psi}_{v_t}\boldsymbol{\omega}_u + \boldsymbol{\rho}_{v_t}\right) - T\log\{\sum_p^P \exp(\boldsymbol{\Psi}_p\boldsymbol{\omega}_u + \boldsymbol{\rho}_p)\} + \log p(\boldsymbol{\omega}_u)$$

As the posterior on ω is intractable, we use a normal distribution $\boldsymbol{\omega}_u \sim \mathcal{N}(\boldsymbol{\mu}_{q_\omega}, \Sigma_{q_\omega})$ to approximate it, we get a variational lower bound of the form:

$$\mathcal{L} = \mathop{\mathbb{E}}_{q(\boldsymbol{\omega}_u)}[\log\ p(v_1, .., v_T, \boldsymbol{\omega}_u|\boldsymbol{\Psi}) - \log q(\boldsymbol{\omega}_u)] = \left(\sum_t^T \boldsymbol{\Psi}_{v_t}\boldsymbol{\mu}_{q_\omega} + \boldsymbol{\rho}_{v_t}\right) - T\mathop{\mathbb{E}}_{q(\boldsymbol{\omega}_u)}[\log\{\sum_p^P \exp(\boldsymbol{\Psi}_p\boldsymbol{\omega}_u + \boldsymbol{\rho}_p)\}] - \mathrm{KL}(\mathrm{q}(\boldsymbol{\omega}_\mathrm{u})|\mathrm{p}(\boldsymbol{\omega}_\mathrm{u})).$$

Where KL is a closed form KL divergence between the variational posterior and the prior (a multivariate standard normal distribution). We see that there is a problematic term associated with the denominator of the softmax. We use the re-parameterization trick [18] to overcome this term. It is also possible to use the Bouchard bound (which also enables an EM algorithm) and the log concave bound, both bounds can alleviate computational issues associated with the softmax sum [7], details of these lower bounds and the EM and simulated EM algorithm are given in the supplementary material.

3.1.1 Re-parameterization Trick. An effective approach to computing expectations with respect to the denominator of the softmax is to use the re-parameterization trick [18], which allows us to take a sample of $\boldsymbol{\omega}$ from the variational distribution and compute a noisy derivative of the lower bound. Within each iteration we proceed by simulating: $\boldsymbol{\epsilon}^{(s)} \sim \mathcal{N}(\mathbf{0}_K, \boldsymbol{I}_K)$, and then computing: $\boldsymbol{\omega}^{(s)} = L_{\Sigma_{q_\omega}}\boldsymbol{\epsilon}^{(s)} + \boldsymbol{\mu}_{q_\omega}$. Where $\boldsymbol{L}_{\Sigma_{q_\omega}}\boldsymbol{L}^T_{\Sigma_{q_\omega}} = \Sigma_{q_\omega}$, we can then optimize the noisy lower bound:

$$\mathcal{L}_{MC} = \left(\sum_t^T \boldsymbol{\Psi}_{v_t}\boldsymbol{\mu}_{q_\omega} + \boldsymbol{\rho}_{v_t}\right) - \mathrm{KL}(\mathrm{q}(\boldsymbol{\omega}_\mathrm{u})|\mathrm{p}(\boldsymbol{\omega}_\mathrm{u})) - T\log[\sum_p^P \exp\{\boldsymbol{\Psi}_p(L_{\Sigma_{q_\omega}}\boldsymbol{\epsilon}^{(s)} + \boldsymbol{\mu}_{q_\omega}) + \boldsymbol{\rho}_p\}]$$

Often Σ_{q_ω} is taken to be diagonal which makes computing $\boldsymbol{L}_{\Sigma_{q_\omega}}$ simply an element-wise square root.

A naive application of the algorithm discussed so far would have the number of variational parameters $\boldsymbol{\mu}_{q_\omega}, \Sigma_{q_\omega}$ growing with the number of user sessions. We propose instead to limit the number of parameters by the use of a variational auto-encoder [18]. This involves using a flexible function and optimizing it to do the job of the EM algorithm i.e.

$$\boldsymbol{\mu}_{q_\omega},\ \Sigma_{q_\omega} = f_\Xi(v_1, ...v_T),$$

Where any function e.g. a deep net can be used for $f_\Xi(\cdot)$ such as a deep or shallow neural network.

3.2 Bandit session training: learning the bandit embeddings

For every user we compute: $\hat{\boldsymbol{\omega}}_u = f(\boldsymbol{v}_u)$ (uncertainty over $\boldsymbol{\omega}_u$ is ignored and a point estimate taken). The hierarchical model has the form:

$$w_a \sim \mathcal{N}(\mu_{0_{w_a}}, \sigma^2_{0_{w_a}}),\ w_b \sim \mathcal{N}(\mu_{0_{w_b}}, \sigma^2_{0_{w_b}}),\ w_c \sim \mathcal{N}(\mu_{0_{w_c}}, \sigma^2_{0_{w_c}})$$

$$\boldsymbol{\kappa}' \sim \mathcal{N}(\mathbf{0}_P, \sigma^2_{\kappa_0}\boldsymbol{I}_P), \qquad \boldsymbol{\kappa} = \boldsymbol{\kappa}' + w_c$$

$$\boldsymbol{\beta}|\boldsymbol{\Psi}, w_a, w_b \sim \mathcal{MN}(s^+(w_a)\boldsymbol{\Psi}, s^+(w_b)\boldsymbol{\Psi}\boldsymbol{\Psi}^T, s^+(w_b)\frac{1}{P}\boldsymbol{\Psi}^T\boldsymbol{\Psi})$$

$$c_n|a_n, \boldsymbol{\beta}, \boldsymbol{\omega}, \boldsymbol{\kappa} \sim \text{Bernoulli}\{\text{sigmoid}(\boldsymbol{\beta}_{a_n}\boldsymbol{\omega}_n + \boldsymbol{\kappa}_{a_n})\}.$$

While β is a [P x K] random variable, we can leverage its low rank covariance matrix to transform the problem to infering a posterior on a [K x K] random variable. This reduces dramatically the training time as P, the size of the catalog items is usually very large compared with K. The low rank alternative parameterization of this distribution can be set as follows. Let:

$$\boldsymbol{\zeta} \sim \mathcal{MN}(\mathbf{0}_{K,K}, \boldsymbol{I}_K, \boldsymbol{I}_K).$$

If we let $\boldsymbol{L} = \text{chol}(\frac{1}{P}\boldsymbol{\Psi}^T\boldsymbol{\Psi})$ i.e. $\boldsymbol{L}\boldsymbol{L}^T = \frac{1}{P}\boldsymbol{\Psi}^T\boldsymbol{\Psi}$. A valid way to sample from a matrix variate normal gives:

$$\boldsymbol{\beta} = s^+(w_a)\boldsymbol{\Psi} + s^+(w_b)\boldsymbol{\Psi}\boldsymbol{\zeta}\boldsymbol{L}^T$$

As mentioned before, we treat the problem in a Bayesian way and approximate the posterior over all the parameters. We use variational inference to transform the problem into an optimization problem. We use a univariate normal variational approximation on w_a, w_b, w_c with means $\mu_{q_{w_a}}, \mu_{q_{w_b}}, \mu_{q_{w_c}}$ and variance $\sigma^2_{q_{w_a}}, \sigma^2_{q_{w_b}}, \sigma^2_{q_{w_c}}$. The variational approximation on κ is a diagonal covariance multivariate normal with mean given by $\boldsymbol{\mu}_{q_\kappa}$ and covariance given by $\text{diag}(\boldsymbol{\sigma}^2_{q_\kappa})$. Similarly we put a univariate normal variational approximation over each element of $\boldsymbol{\zeta}$ parameterized so that $\boldsymbol{\zeta}_{i,j}$ has mean $\mu_{q_{\zeta_{i,j}}}$ and variance $\sigma^2_{q_{\zeta_{i,j}}}$. This gives us $2(P + K^2 + 3)$ parameters to estimate. We denote Q as the Gaussian

[4] This includes having reward positive and no-reward zero, capping and variance penalization

variational posterior over all of the parameters, and P the prior and maximize :

$$\mathcal{L} = \mathop{\mathbb{E}}_{Q}[c_n \log \text{sigmoid}(\lambda_n) + (1 - c_n) \log\{1 - \text{sigmoid}(\lambda_n)\}] \quad (1)$$
$$- \frac{1}{N}\text{KL}(Q|P),$$

where:

$$\lambda_n = \boldsymbol{\beta}_{a_n}\hat{\boldsymbol{\omega}}_n + \boldsymbol{\kappa}_{a_n}$$
$$= s^+(w_a)\boldsymbol{\Psi}_{a_n}\hat{\boldsymbol{\omega}}_n + s^+(w_b)\{(\boldsymbol{L}\hat{\boldsymbol{\omega}}_n)^T \otimes \boldsymbol{\Psi}_{a_n}\}\text{vec}(\boldsymbol{\zeta}) + \boldsymbol{\kappa}_{a_n}$$

We use the local re-parameterization trick [17] which uses the Affine transform properties of multivariate Gaussian distribution to allow the re-parameterization trick to be employed on lower dimensions. This results in sampling at lower dimensions and more importantly makes the derivatives of the loss less noisy. To implement the local re-parameterization trick we draw random samples:

$$\boldsymbol{\epsilon}_{w_a} \sim \mathcal{N}(0,1), \quad \boldsymbol{\epsilon}_{w_b} \sim \mathcal{N}(0,1), \quad \boldsymbol{\epsilon}_{\text{lrt}} \sim \mathcal{N}(0,1), \quad \boldsymbol{\epsilon}_{\kappa} \sim \mathcal{N}(0,1).$$

with $\boldsymbol{R}_n = (\boldsymbol{L}\hat{\boldsymbol{\omega}}_n)^T \otimes \boldsymbol{\Psi}_{a_n}$, we can get a one dimentional noisy estimate of λ_n :

$$\hat{\lambda}_n = s^+(\mu_{q_{w_a}} + \epsilon_{w_a}\sigma_{q_{w_a}})\boldsymbol{\Psi}_{a_n}\hat{\boldsymbol{\omega}}_n$$
$$+ s^+(\mu_{q_{w_b}} + \epsilon_{w_b}\sigma_{q_{w_b}})(\boldsymbol{R}_n\,\text{vec}(\boldsymbol{\mu}_{q_\zeta}) + \|\boldsymbol{R}_n^T \odot \text{vec}(\boldsymbol{\sigma}_{q_\zeta})\|_2\epsilon_{\text{lrt}})$$
$$+ \boldsymbol{\mu}_{q_{\kappa a}} + \mu_{q_{w_c}} + \epsilon_\kappa\sqrt{\boldsymbol{\sigma}^2_{q_{\kappa a}} + \sigma^2_{q_{w_c}}}.$$

where $\|\cdot\|_2$ denotes the L_2 norm and $\odot$ element wise multiplication. We can optimize a noisy version of our objective :

$$\hat{\mathcal{L}}_n = c_n \log \text{sigmoid}(\hat{\lambda}_n) + (1 - c_n)\log\{1 - \text{sigmoid}(\hat{\lambda}_n)\} \quad (2)$$
$$- \frac{1}{N}\text{KL}(Q|P).$$

We call the solution of this optimization problem **BLOB-NQ** as we considered a Normal approximation for the posterior on ζ.

An alternative approach is to use a matrix variate normal distribution as the variational approximation of $\boldsymbol{\zeta}$ with mean matrix $\boldsymbol{\mu}_{q_\zeta}$ and the two covariance matrices given by: $\text{diag}(\boldsymbol{\sigma}^2_{q_{\zeta_1}})$ and $\text{diag}(\boldsymbol{\sigma}^2_{q_{\zeta_2}})$. This reduces the number of variational parameters used for representing the variance of the variational posterior. We thus need to estimate $2(P+3) + K^2 + 2K$ which is less then the previous approximation for $K \geq 2$. To apply the local re-parameterization trick let:

$$std_n = \sqrt{(\boldsymbol{\sigma}^2_{q_{\zeta_1}} \cdot \boldsymbol{\Psi}^2_{a_n})(\boldsymbol{\sigma}^2_{q_{\zeta_2}} \cdot (\boldsymbol{L}^T\hat{\boldsymbol{\omega}}_n)^2)}$$

$$\hat{\phi}_n = s^+(\mu_{q_{w_a}} + \epsilon_{w_a}\sigma_{q_{w_a}})\boldsymbol{\Psi}_{a_n}\hat{\boldsymbol{\omega}}_n$$
$$+ s^+(\mu_{q_{w_b}} + \epsilon_{w_b}\sigma_{q_{w_b}})\{\boldsymbol{\Psi}_{a_n}\boldsymbol{\mu}_{q_\zeta}\boldsymbol{L}^T\hat{\boldsymbol{\omega}}_n + std_n\boldsymbol{\epsilon}_{\text{lrt}}\}$$
$$+ \boldsymbol{\mu}_{q_{\kappa a}} + \mu_{q_{w_c}} + \epsilon_\kappa\sqrt{\boldsymbol{\sigma}^2_{q_{\kappa a}} + \sigma^2_{q_{w_c}}}.$$

A noisy estimate of the lower bound can then be computed by substituting $\hat{\phi}_n$ into Equation (2). We call its solution **BLOB-MNQ** as we use a Matrix Normal variational posterior.

In both approximations and when the objective is at its maximum, we can take a point estimate of the bandit embeddings:

$$\hat{\boldsymbol{\beta}} = s^+(\mu_{q_{w_a}})\boldsymbol{\Psi} + s^+(\mu_{q_{w_b}})\boldsymbol{\Psi}\boldsymbol{\mu}_{q_\zeta}\boldsymbol{L}^T.$$

The bandit embedding can be interpreted as a weighted sum of the organic embedding and the organic embedding multiplied by a $K \times K$ matrix that can adjust the bandit embeddings based on the bandit signal.

4 RESULTS

4.1 Organic Evaluation

We demonstrate that our method produces useful user representations on next item prediction using the RecoGym simulation environment [33]. RecoGym is a framework for simulating a recommender system and enables the simulation of A/B tests although here we simply use it to create organic sequences of item views and test the organic model's ability to do next item prediction. We split both the datasets into train and test so that sessions reside entirely in one of the two groups. We fit the model to the training set, we then evaluate by providing the model $v_1, ..v_{T_u-1}$ events and testing the model's ability to predict v_{T_u}.

The organic model was implemented using the PyTorch automatic differentiation package in Python [30] and trained using Stochastic Gradient Descent (SGD), specifically the RMSProp variant. We set the learning rate to 0.001 and tune the other hyperparameters, including L2 regularization, for each dataset based upon a validation set[5].

The various models are evaluated using recall at K (RC@K) and truncated discounted cumulative gain at K (DCG@K), which are defined below.

Let r_k be the kth highest value of $p(\boldsymbol{\omega}_{v_{T_u}}|v_1, ..v_{T_u-1})$. For all results presented in this paper, we set K to 5.

$$\text{RC@K} = \begin{cases} 1, & \text{if } v_{T_u} \in \{r_1, ..., r_K\}. \\ 0, & \text{otherwise.} \end{cases}$$

$$\text{DCG@K} = \sum_i \frac{2^{r_i \mathbf{1}\{v_{T_u} \in \{r_1, ..., r_K\}\}} - 1}{\log i + 1}.$$

We compute the average of these quantities over all sessions in the test set.

We consider two alternative methods for training the model:

- **Bouch/AE** - A linear variational auto-encoder using the Bouchard bound (see the supplementary material).
- **RT/AE** - A deep auto-encoder again using the re-parameterization trick. The deep auto-encoder consists of mapping an input of size P to three linear rectifier layers of K units each.

When we update the posterior over a user's latent variable representation at test time, we assess both using the auto-encoder denoted AE and using the 100 iterations of the EM algorithm denoted EM in the results.

When we compute next item predictions we consider both using a 100 sample Monte Carlo approximation denoted MC and just taking the mean as a point estimate denoted mean it uses only $\boldsymbol{\mu}_{q_\omega}$ (and correspondingly ignores Σ_q).

[5]Source code: https://github.com/criteo-research/blob). The RecoGym simulator allows reproducible results for all recommendation algorithms and policies.

Train Algorithm	Online Latent	Online Next Item	RC@5	DCG@5
Pop			0.020	0.016
ItemKNN			0.020	0.024
RNN			0.035	0.033
Bouch/AE	AE	MC	0.082	0.128
Bouch/AE	AE	mean	0.082	0.079
Bouch/AE	EM	MC	**0.117**	0.128
Bouch/AE	EM	mean	**0.117**	**0.130**
RT/AE	AE	MC	0.090	0.105
RT/AE	AE	mean	0.080	0.068
RT/AE	EM	MC	0.090	0.105
RT/AE	EM	mean	0.090	0.106

Table 2: Results on the testset of RecoGym dataset with 2000 products. For both metrics, a higher value is better.

To demonstrate the effectiveness of our approach, we present results from the following baseline approaches:

Popularity: Item popularity provides no personalization, but is nonetheless a strong baselines for certain recommendation tasks.

Item KNN: Item K Nearest Neighbors (KNN) involves computing the correlation matrix of the sample data adding the identity to prevent division by zero and then using these correlations as recommendations based on a user's most recent historical item. The limitations of this technique is that it ignores item popularity and multiple items in the user's history, but despite these limitations it is often a strong baseline.

Recurrent Neural Network: For this baseline, we make use of a recurrent neural network to learn a user representation by predicting the next item in the session. The model architecture we employ is similar to that of [13], in that we feed the output from an embedding layer into a Gated Recurrent Unit (GRU) [9] with 64 hidden units to learn the temporal dynamics of the user's session. The output from the GRU is then passed through a final softmax layer which gives the probability of the next item in the sequence. The network is trained to minimize the categorical cross-entropy over the training sessions via RMSProp.

For our organic experiment we use the RecoGym simulator with 2000 products and $\sigma_\omega = 0$, i.e. a static user state, we generate a training set of 100 sessions and a test set of 100 sessions, this results in 21852 and 19533 events for train and test respectively. The BLO models were all trained using 15000 epochs using the RMSProp algorithm, the embedding size was set to 10. The RNN was trained with K=200 for 5000 epochs (it performed slightly worse with a training run of 25000). The results are shown in Table 2. BLO is much better than the baselines at standard organic recommender systems metrics. However if being able to build an adequate model of organic behaviour is sufficient for building a recommender system depends on if the organic behaviour is aligned with bandit behaviour. This requires using RecoGym for its intended purpose simulating A/B tests and varying the agreement between the organic behavior and bandit behavior using the provided flips parameter.

4.2 The Complete Model - Organic and Bandit

4.2.1 Experimental Setup. Unfortunately no real world dataset exhibits the required properties (both organic and bandit behavior) moreover no real world dataset including counterfactual datasets allow us to evaluate the quality of a recommender systems recommendations reliably. For this reason for the complete dataset we do our evaluations completely in the RecoGym simulator. A strong advantage of the simulation environment is that not only can we compute offline organic metrics but we can also simulate A/B tests.

Another advantage of the RecoGym simulator that simulates both organic and bandit behaviour is that algorithms from the traditional organic part of recommender systems research and bandit algorithms can be compared side by side. We consider traditional organic algorithms like ItemKNN [10] along side our organic Bayesian Latent Organic model (BLO) and sophisticated deep learning approaches such as the MultiVAE [24]. In the case of bandit algorithms we can test value based logistic regression as well as the policy based contextual bandit. In order to apply any bandit algorithm we need to perform feature engineering in order to transform the history consisting of item views into a vector of history. For the logistic regression we elect to make a P dimensional feature vector crossed with the action also of size P giving P^2 features. Similarly the contextual bandit is a linear model that maps the P dimensional vector of historical counts to a P dimensional action space.

We are interested to see how the recommender system responds to different logging policies, we therefore test it using a good logging policy based on the session popularity. That is the probability $1 - \epsilon$ is shared proportionally to the items in a users history we use considerable exploration ($\epsilon = 0.3$). We are interested in the (common) case where we have plentiful organic data so we set RecoGym to have 20000 organic sessions. Finally we are interested in situations where the next item prediction is an optimal recommendation and cases where the organic signal alone is misleading to recommendation quality. This connection between the organic and the bandit signal is controlled with the *flips* parameter in RecoGym. The flips parameter permutes the behavior of two actions.

A unique feature of RecoGym is that we are able to simulate both organic and bandit feedback, this means we are able to compare algorithms that operate on the bandit signal (both policy and value based) with algorithms that operate on the organic signal. We consider the following baselines:

Logistic regression (bandit, value): Perhaps the simplest way to process a bandit signal. We regress the reward on features derived from the users history and the recommended action. In order to deliver the recommendation we predict the reward for every action and select the highest.

Contextual bandit (bandit, policy): The contextual bandit is a policy based method that maps a context to a recommendation in one-of-n coding a vector of length P. The algorithm is trained using counterfactual risk minimization using the IPS score logged by RecoGym without any clipping or variance penalty.

Session ItemKNN (organic): This organic algorithm operates by determining for each session if an item was present or absent, from this dataset a correlation matrix is computed. At recommendation is delivered by computing the average correlations for each

item in history as a single vector and then taking the maximum. We take the whole session into account rather than the most recent item (unlike most recent ItemKNN used above).

Multi-VAE (organic): A state of the art deep learning recommendation algorithm similar to the organic portion of the model presented here except the model is non-linear and uses some non-standard heuristics such as "beta-annealing".

BLO (organic): The organic portion of the model developed here. We set the embedding size to be K=20 and use a linear variational auto-encoder. This is implemented in PyTorch. A learning rate of 0.0001 is used with 1000 epochs and an embedding size of $K = 20$.

BLOB (organic and bandit combined): The complete model developed here. We use priors: $w_a \sim \mathcal{N}(-1, 1^2)$ $w_b \sim \mathcal{N}(-6, 1^2)$ $w_c \sim \mathcal{N}(-4.5, 10^2)$ $\kappa \sim \mathcal{N}(w_c, 0.01^2 \boldsymbol{I})$. We consider both the normal variational approximation NQ and the matrix normal variational approximation MNQ. The bandit layer is implemented using TensorFlow with a learning rate of 0.001 and 800 epochs for the P=100 and 1200 epochs for the P=1000, with a batch size of 1024 and using the RMSprop training algorithm.

Random: The actions are recommended randomly. A weak baseline but useful to calibrate performance.

4.2.2 Experimental Results. The first experiment considers the catalog size to be P=100, the number of user sessions to be 1000, the simulated A/B test is done over 4000 users and the logging policy being session popularity with epsilon greedy exploration (epsilon=0.3). This means that the bandit signal will resemble that found in real systems with a strong signal around some actions favoured by the previous version of the recommender system (session popularity policy - a decent baseline) and a weak signal over much of the remaining action space. Results are shown in Table 3.

In the Flips=0 scenario RecoGym is configured so that next item prediction based on organic data is a perfect proxy for delivering good recommendations. As a consequence all the organic based methods do well including the BLO (organic), both our methods that combine organic and bandit BLOB-NQ and BLOB-MNQ and the Multi-VAE baseline, the Session ItemKNN baseline while organic does not perform well.

When the Flips=50 scenario RecoGym internally permutes 50 actions behavior this means that next item prediction is now a poor proxy of recommendation performance. We see this as all purely organic based agents now perform poorly indeed the connection between organic and bandit is reduced to the point that Session ItemKNN, the Multi VAE and BLO all perform worse than random. It is in this case that the value of our BLOB model is demonstrated as both BLOB NQ and BLOB MNQ perform strongly.

For the purely signal bandit algorithms the value based Log Reg and the policy based CB perform similarly to each other and with Flips=0 and Flips=50. They perform a little better than random (except for CB Flips=50) demonstrating that there is some usable signal in the bandit feed back but are far from state of the art especially in the Flips=0 case where ignoring the organic signal profoundly limits recommendation quality. In the Flips=50 case the pure bandit approaches outperform the purely organic algorithms but the combined approach performs significantly better giving a click through rate of 1.57% for the BLOB NQ compared to 1.21% for the logistic regression.

Importantly the BLOB NQ and BLOB MNQ outperform or equal the other methods in the Flips=0 setting and outperform the other methods in the Flips=50 setting.

Table 3: Simulated A/B test results on the RecoGym simulator using: P=100, U=1000, organic only sessions=20 000.

Agent	Type	CTR (%) Flips=0	CTR (%) Flips=50
Log Reg	(bandit)	1.37	1.21
CB	(bandit)	1.37	1.09
ItemKNN	(organic)	1.39	0.92
MultiVAE	(organic)	**2.43**	0.76
BLO	(organic)	**2.42**	0.76
BLOB-NQ	(combined)	**2.42**	**1.57**
BLOB-MNQ	(combined)	**2.40**	**1.56**
Random		1.09	1.11

The second experiment considers the same setup but with $P = 1000$, we also increase the number of epochs on the bandit component of the model to 1200. Results are shown in Table 4.

Again we see that the methods that use the organic data either the purely organic or the combined BLOB methods we propose perform work well when Flips=0, but when Flips=500 the purely organic methods fall in performance to little above random yet the combined methods BLOB-MNQ and BLOB-NQ continue to perform well beating all other baselines.

The policy based contextual bandit shows a small improvement over the value based logistic regression in the Flips=0 case although this advantage vanishes when Flips=500, this is may be due to the fact that the contextual bandit "clings" to the logging policy and the session popularity logging policy is better in the case where Flips=0.

Table 4: Simulated A/B test results on the RecoGym simulator using: P=1000, U=1000, organic only sessions=20 000.

Agent	Type	CTR (%) Flips=0	CTR (%) Flips=500
Log Reg	(bandit)	1.26	1.30
CB	(bandit)	1.38	1.29
ItemKNN	(organic)	1.39	0.87
MultiVAE	(organic)	**2.43**	1.15
BLO	(organic)	**2.42**	1.13
BLOB-NQ	(combined)	**2.40**	1.51
BLOB-MNQ	(combined)	**2.39**	**1.62**
Random		1.13	1.12

5 CONCLUSION

We focus on a particular recommendation task, one where a user profile is defined by a history of items in a catalog and the recommendation task is to recommend items from the same catalog. Our model is able to learn both from the organic signal and the bandit

signal jointly beating baselines in a range of settings by exploiting the three fundamental distances of recommendation action-history, action-action and history-history.

We use computational techniques which allow allow large scale Bayesian inference suitable for Recommendation with large catalogs. The local re-parameterization trick was particularly valuable in reducing the variance in our optimisation problem.

BLOB is able to perform well both in situations where next item prediction is a good proxy for recommendations and situations where it is poor. Meeting the performance of pure organic algorithms in settings where the organic signal is sufficient and exceeding all baselines organic and bandit (policy) and bandit (value). This strongly validates the value of Bayesian methods to infer in the cases of a signal of varying strength and their practical value thanks to modern developments in Bayesian deep learning.

There are many possible extension to this work, one is to produce end to end training i.e. training both the organic and bandit component simultaneously. To apply this approach would require a more complicated training procedure. We also expect there are other useful ways to combine organic and bandit signal, perhaps based on models that avoid the softmax and sigmoid transform such as LDA for the organic and using the approach out lined in [25] for the Bandit. Avoiding softmax and sigmoid transforms has both computational advantages and can increase interpretability.

REFERENCES

[1] Gediminas Adomavicius and Alexander Tuzhilin. 2005. Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions. *IEEE transactions on knowledge and data engineering* 17, 6 (2005), 734–749.

[2] James O Berger, Robert L Wolpert, MJ Bayarri, MH DeGroot, Bruce M Hill, David A Lane, and Lucien LeCam. 1988. The likelihood principle. *Lecture notes-Monograph series* 6 (1988), iii–199.

[3] Alina Beygelzimer and John Langford. 2009. The offset tree for learning with partial labels. In *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 129–138.

[4] David M. Blei and John D. Lafferty. 2005. Correlated Topic Models. In *Advances in Neural Information Processing Systems 18 [Neural Information Processing Systems, NIPS 2005, December 5-8, 2005, Vancouver, British Columbia, Canada]*. 147–154. http://papers.nips.cc/paper/2906-correlated-topic-models

[5] David M Blei, Andrew Y Ng, and Michael I Jordan. 2003. Latent dirichlet allocation. *Journal of machine Learning research* 3, Jan (2003), 993–1022.

[6] L. Bottou, J. Peters, J. Quiñonero-Candela, D. Charles, D. Chickering, E. Portugaly, D. Ray, P. Simard, and E. Snelson. 2013. Counterfactual reasoning and learning systems: The example of computational advertising. *The Journal of Machine Learning Research* 14, 1 (2013), 3207–3260.

[7] Guillaume Bouchard. 2007. Efficient bounds for the softmax function, applications to inference in hybrid models. (2007).

[8] Olivier Cappé and Eric Moulines. 2009. On-line expectation–maximization algorithm for latent data models. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)* 71, 3 (2009), 593–613.

[9] Kyunghyun Cho, Bart van Merrienboer, Çaglar Gülçehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2014. Learning Phrase Representations using RNN Encoder-Decoder for Statistical Machine Translation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing, EMNLP 2014, October 25-29, 2014, Doha, Qatar, A meeting of SIGDAT, a Special Interest Group of the ACL*. ACL, 1724–1734. http://aclweb.org/anthology/D/D14/D14-1179.pdf

[10] James Davidson, Benjamin Liebald, Junning Liu, Palash Nandy, Taylor Van Vleet, Ullas Gargi, Sujoy Gupta, Yu He, Mike Lambert, Blake Livingston, and others. 2010. The YouTube video recommendation system. In *Proceedings of the fourth ACM conference on Recommender systems*. 293–296.

[11] Aristides Gionis, Piotr Indyk, Rajeev Motwani, and others. 1999. Similarity search in high dimensions via hashing. In *Vldb*, Vol. 99. 518–529.

[12] Miguel A Hernan and James M Robins. 2010. Causal inference. (2010).

[13] Balázs Hidasi and Alexandros Karatzoglou. 2018. Recurrent Neural Networks with Top-k Gains for Session-based Recommendations. In *Proceedings of the 27th ACM International Conference on Information and Knowledge Management, CIKM 2018, Torino, Italy, October 22-26, 2018*, Alfredo Cuzzocrea, James Allan, Norman W. Paton, Divesh Srivastava, Rakesh Agrawal, Andrei Z. Broder, Mohammed J. Zaki, K. Selçuk Candan, Alexandros Labrinidis, Assaf Schuster, and Haixun Wang (Eds.). ACM, 843–852. DOI: http://dx.doi.org/10.1145/3269206.3271761

[14] Tommi Jaakkola and Michael Jordan. 1997. A variational approach to Bayesian logistic regression models and their extensions. In *Sixth International Workshop on Artificial Intelligence and Statistics*, Vol. 82. 4.

[15] O. Jeunen, D. Rohde, F. Vasile, and M. Bompaire. 2020. Joint Policy-Value Learning for Recommendation. In *Proc. of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining (KDD '20)*.

[16] Fredrik Johansson, Uri Shalit, and David Sontag. 2016. Learning representations for counterfactual inference. In *International Conference on Machine Learning*. 3020–3029.

[17] Diederik P Kingma, Tim Salimans, and Max Welling. 2015. Variational dropout and the local reparameterization trick. In *Advances in Neural Information Processing Systems*. 2575–2583.

[18] Diederik P. Kingma and Max Welling. 2014. Auto-Encoding Variational Bayes. In *2nd International Conference on Learning Representations, ICLR 2014, Banff, AB, Canada, April 14-16, 2014, Conference Track Proceedings*, Yoshua Bengio and Yann LeCun (Eds.). https://openreview.net/group?id=ICLR.cc/2014

[19] David A. Knowles and Tom Minka. 2011. Non-conjugate Variational Message Passing for Multinomial and Binary Regression. In *Advances in Neural Information Processing Systems 24*, J. Shawe-Taylor, R. S. Zemel, P. L. Bartlett, F. Pereira, and K. Q. Weinberger (Eds.). Curran Associates, Inc., 1701–1709.

[20] Yehuda Koren and Robert Bell. 2015. Advances in collaborative filtering. In *Recommender systems handbook*. Springer, 77–118.

[21] John D. Lafferty and David M. Blei. 2006. Correlated Topic Models. In *Advances in Neural Information Processing Systems 18*, Y. Weiss, B. Schölkopf, and J. C. Platt (Eds.). MIT Press, 147–154. http://papers.nips.cc/paper/2906-correlated-topic-models.pdf

[22] Tor Lattimore and Csaba Szepesvári. 2018. Bandit algorithms. *preprint* (2018), 28.

[23] Dawen Liang, Rahul G. Krishnan, Matthew D. Hoffman, and Tony Jebara. 2018. Variational Autoencoders for Collaborative Filtering. In *Proceedings of the 2018 World Wide Web Conference on World Wide Web, WWW 2018, Lyon, France, April 23-27, 2018*, Pierre-Antoine Champin, Fabien L. Gandon, Mounia Lalmas, and Panagiotis G. Ipeirotis (Eds.). ACM, 689–698. DOI: http://dx.doi.org/10.1145/3178876.3186150

[24] D. Liang, R. G. Krishnan, M. D Hoffman, and T. Jebara. 2018. Variational autoencoders for collaborative filtering. In *Proc. of the 2018 World Wide Web Conference (WWW '18)*. International World Wide Web Conferences Steering Committee, ACM, 689–698.

[25] Alberto Lumbreras, Louis Filstroff, and Cédric Févotte. 2018. Bayesian mean-parameterized nonnegative binary matrix factorization. *arXiv preprint arXiv:1812.06866* (2018).

[26] Yury A Malkov and Dmitry A Yashunin. 2018. Efficient and robust approximate nearest neighbor search using hierarchical navigable small world graphs. *IEEE transactions on pattern analysis and machine intelligence* (2018).

[27] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed Representations of Words and Phrases and their Compositionality. In *Advances in Neural Information Processing Systems 26*, C. J. C. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K. Q. Weinberger (Eds.). Curran Associates, Inc., 3111–3119.

[28] Radford M Neal. 2012. *Bayesian learning for neural networks*. Vol. 118. Springer Science & Business Media.

[29] Tui H Nolan and Matt P Wand. 2017. Accurate logistic variational message passing: algebraic and numerical details. *Stat* 6, 1 (2017), 102–112.

[30] Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. 2017. Automatic differentiation in PyTorch. In *NIPS-W*.

[31] Ya'acov Ritov, Peter J Bickel, Anthony C Gamst, Bastiaan Jan Korneel Kleijn, and others. 2014. The Bayesian Analysis of Complex, High-Dimensional Models: Can It Be CODA? *Statist. Sci.* 29, 4 (2014), 619–639.

[32] Herbert Robbins and Sutton Monro. 1951. A stochastic approximation method. *The annals of mathematical statistics* 22, 3 (1951), 400–407.

[33] David Rohde, Stephen Bonner, Travis Dunlop, Flavian Vasile, and Alexandros Karatzoglou. 2018. RecoGym: A Reinforcement Learning Environment for the problem of Product Recommendation in Online Advertising. In *REVEAL workshop, ACM Conference on Recommender Systems 2018*.

[34] David Rohde and Matt P Wand. 2016. Semiparametric mean field variational Bayes: General principles and numerical issues. *The Journal of Machine Learning Research* 17, 1 (2016), 5975–6021.

[35] Francisco JR Ruiz, Michalis K Titsias, Adji B Dieng, and David M Blei. 2018. Augment and reduce: Stochastic inference for large categorical distributions. *arXiv*

preprint arXiv:1802.04220 (2018).

[36] Tobias Schnabel, Adith Swaminathan, Ashudeep Singh, Navin Chandak, and Thorsten Joachims. 2016. Recommendations As Treatments: Debiasing Learning and Evaluation. In *Proceedings of the 33rd International Conference on International Conference on Machine Learning - Volume 48 (ICML'16)*. 1670–1679.

[37] A. Storkey. 2009. When training and test sets are different: characterizing learning transfer. *Dataset shift in machine learning* (2009), 3–28.

[38] Adith Swaminathan and Thorsten Joachims. 2015. Batch learning from logged bandit feedback through counterfactual risk minimization. *Journal of Machine Learning Research* 16 (2015), 1731–1755.

[39] A. Swaminathan and T. Joachims. 2015. Batch learning from logged bandit feedback through counterfactual risk minimization. *Journal of Machine Learning Research* 16, 1 (2015), 1731–1755.

[40] Michalis Titsias. 2016. One-vs-each approximation to softmax for scalable estimation of probabilities. In *Advances in Neural Information Processing Systems*. 4161–4169.

[41] Vladimir Vapnik. 2013. *The nature of statistical learning theory*. Springer science & business media.

6 SUPPLEMENTARY MATERIAL

6.1 Approximating expectations under the log softmax

The variational lower bound of BLO (and BLOB) contains a log softmax term. An alternative to using the re-parameterization trick is to use The Bouchard bound which removes the need for Monte Carlo methods. The Bouchard bound introduces a further approximation and additional variational parameters a, ξ but produces an analytical bound:

$$\begin{aligned}
\mathcal{L} \geq \mathcal{L}_{\text{Bouch}} &= \left(\sum_t^T \boldsymbol{\Psi}_{v_t} \boldsymbol{\mu}_{q_\omega} + \boldsymbol{\rho}_{v_t}\right) \\
&- T[a + \sum_p^P \frac{\boldsymbol{\Psi}_p \boldsymbol{\mu}_{q_\omega} + \boldsymbol{\rho}_p - a - \xi_p}{2} + \log(1 + e^{\xi_p}) \\
&+ \lambda_{\text{JJ}}(\xi_p)\{(\boldsymbol{\Psi}_p \boldsymbol{\mu}_{q_\omega} + \boldsymbol{\rho}_p - a)^2 + \boldsymbol{\Psi}_p \Sigma_{q_\omega} \boldsymbol{\Psi}_p^T - \xi_p^2\}] \\
&- \frac{K}{2}\log(2\pi) - \frac{1}{2}\{\boldsymbol{\mu}_{q_\omega}^T \boldsymbol{\mu}_{q_\omega} + \text{trace}(\Sigma_{q_\omega})\} + \frac{1}{2}\log|2\pi e \Sigma_{q_\omega}|.
\end{aligned}$$

Because the Bouchard bound causes the softmax to decompose into a sum we can avoid the expensive normalization by subsampling some of the terms in the softmax.

$$\begin{aligned}
\hat{\mathcal{L}}_{\text{Bouch}}(v_1, ..., v_T, n_1, ...n_S, \Xi, \boldsymbol{\Psi}) &= \left(\sum_t^T \boldsymbol{\Psi}_{v_t} \boldsymbol{\mu}_{q_\omega} + \boldsymbol{\rho}_{v_t}\right) \\
&- T[a + \frac{P}{S}\sum_{s'=1}^S \frac{\boldsymbol{\Psi}_{n_{s'}} \boldsymbol{\mu}_{q_\omega} + \boldsymbol{\rho}_{n_{s'}} - a - \xi_{n_{s'}}}{2} + \log(1 + e^{\xi_{n_{s'}}}) \\
&+ \lambda_{\text{JJ}}(\xi_{n_{s'}}) \times \{(\boldsymbol{\Psi}_{n_{s'}} \boldsymbol{\mu}_{q_\omega} + \boldsymbol{\rho}_{n_{s'}} - a)^2 + \boldsymbol{\Psi}_{n_{s'}} \Sigma_{q_\omega} \boldsymbol{\Psi}_{n_{s'}}^T - \xi_{n_{s'}}^2\}] \\
&- \frac{K}{2}\log(2\pi) - \frac{1}{2}\{\boldsymbol{\mu}_{q_\omega}^T \boldsymbol{\mu}_{q_\omega} + \text{trace}(\Sigma_{q_\omega})\} + \frac{1}{2}\log|2\pi e \Sigma_{q_\omega}|.
\end{aligned}$$

where $v_1, ..., v_T$ are the items associated with the session and $n_1, ...n_S$ are $S < P$ negative items randomly sampled, and $\lambda_{\text{JJ}}(\cdot)$ is the Jaakola and Jordan function [14]:

$$\lambda_{\text{JJ}}(\xi) = \frac{1}{2\xi}\left(\frac{1}{1 + e^{-\xi}} - \frac{1}{2}\right).$$

This algorithm is similar to the word2vec algorithm [27] but without any non-probabilistic heuristics.

6.2 Log concavity bound

The log concave bound [4, 7, 35] also breaks the log softmax into a sum

$$\begin{aligned}
\log p(v_1, .., v_T, \boldsymbol{\omega}_u | \boldsymbol{\Psi}) &= \left(\sum_t^T \boldsymbol{\Psi}_{v_t} \boldsymbol{\omega}_u + \boldsymbol{\rho}_{v_t}\right) \\
&- T \log\{\sum_p^P \exp(\boldsymbol{\Psi}_p \boldsymbol{\omega}_u + \boldsymbol{\rho}_p)\} - \frac{K}{2}\log(2\pi) - \frac{1}{2}\boldsymbol{\omega}_u^T \boldsymbol{\omega}_u \\
&\geq \left(\sum_t^T \boldsymbol{\Psi}_{v_t} \boldsymbol{\omega}_u + \boldsymbol{\rho}_{v_t}\right) \\
&- T\phi\{\sum_p^P \exp(\boldsymbol{\Psi}_p \boldsymbol{\omega}_u + \boldsymbol{\rho}_p)\} + T\log\phi + T - \frac{K}{2}\log(2\pi) - \frac{1}{2}\boldsymbol{\omega}_u^T \boldsymbol{\omega}_u \\
&= L_{\log}
\end{aligned}$$

$$\mathcal{L}_{log} = E_{q(\omega)}[L_{\log}] - \mathrm{KL}(Q,P) = \left(\sum_t^T \boldsymbol{\Psi}_{v_t}\boldsymbol{\mu}_{q_\omega} + \boldsymbol{\rho}_{v_t}\right)$$
$$- T\phi\{\sum_p^P \exp(\boldsymbol{\Psi}_p\boldsymbol{\mu}_{q_\omega} + \boldsymbol{\rho}_p + \frac{1}{2}\boldsymbol{\Psi}_p\Sigma_{q_\omega}\boldsymbol{\Psi}_p^T)\} + \log\phi + 1$$
$$- \mathrm{KL}(Q,P).$$

A fast noisy version of the bound is:

$$\hat{\mathcal{L}}_{log}(v_1,..,v_T,n_1,n_{S_{\text{neg}}}) = \left(\sum_t^T \boldsymbol{\Psi}_{v_t}\boldsymbol{\mu}_{q_\omega} + \boldsymbol{\rho}_{v_t}\right) - \mathrm{KL}(Q,P)$$
$$- T\frac{P}{S_{\text{neg}}}\phi\{\sum_{s'}^{S_{\text{neg}}} \exp(\boldsymbol{\Psi}_{n_{s'}}\boldsymbol{\mu}_{q_\omega} + \boldsymbol{\rho}_{n_{s'}} + \frac{1}{2}\boldsymbol{\Psi}_{n_{s'}}\Sigma_{q_\omega}\boldsymbol{\Psi}_{n_{s'}}^T)\} + T\log\phi + T$$

Finally the one vs each bound [40] also breaks the log softmax into a sum without introducing any variational parameter whatsoever.

We can also use a variational auoto-encoders for a, ξ in the case of the Bouchard bound and ϕ in the case of the log concave bound to prevent variational parameters growing with the size of the dataset. This is similar to the augment and reduce approach [35] but has no requirement to be in complete data exponential family form.

The computational impact of turning the log softmax into a sum computationally is driven by P and GPU size. If P is small compared to the GPU it may be preferable to avoid using any additional approximations and compute the full softmax using the reparameterization trick.

6.3 The EM Algorithm - an alternative to the VAE

6.3.1 Standard EM algorithm. If the parameters $\boldsymbol{\Psi}, \boldsymbol{\rho}$ are already known then the posterior over the user embedding $\boldsymbol{\omega}$ may be calculated by optimizing the lower bound using the following variational EM algorithm. The EM algorithm exploits the fact that the Bouchard bound is quadratic and conjugate to the Gaussian distribution. The algorithm here is the *dual* of the one presented in [7] as we assume the embedding $\boldsymbol{\Psi}$ is fixed and $\boldsymbol{\omega}$ is updated where the algorithm they present does the opposite. The EM algorithm consists of cycling the following update equations:

$$\Sigma_{q_\omega}^{-1} = I_k + 2T\sum_p \lambda_{\text{JJ}}(\xi_p)\boldsymbol{\Psi}_p^T\boldsymbol{\Psi}_p,$$

$$\boldsymbol{\mu}_{q_\omega} = \Sigma_{q_\omega}\left((\sum_t^T \boldsymbol{\Psi}_{v_t}^T) - T\left[\sum_p^P\{\frac{1}{2} + 2(\boldsymbol{\rho}_p - a)\lambda_{\text{JJ}}(\xi_p)\}\boldsymbol{\Psi}_p^T\right]\right),$$

$$a = \frac{-1 + \frac{P}{2} + \sum_p 2\lambda_{\text{JJ}}(\xi_p)(\boldsymbol{\Psi}_p\boldsymbol{\mu}_{q_\omega} + \boldsymbol{\rho}_p)}{2\sum_p \lambda_{\text{JJ}}(\xi_p)},$$

$$\xi_p = h(\boldsymbol{\Psi}_p, \boldsymbol{\rho}_p, a, \Sigma_{q_\omega}, \boldsymbol{\rho}_q) = \sqrt{\boldsymbol{\Psi}_p\Sigma_{q_\omega}\boldsymbol{\Psi}_p^T + (\boldsymbol{\Psi}_p\boldsymbol{\mu}_{q_\omega} + \boldsymbol{\rho}_p - a)^2}.$$

6.3.2 Fast online EM algorithm. We further note that the EM algorithm is (with the exception of the a variational parameter) a fixed point update (of the natural parameters) that decomposes into a sum. The terms in the sum come from the softmax in the denominator. After substituting a co-ordinate descent update of a with a gradient descent step update, then the entire fixed point update becomes a sum:

$$(\Sigma_{q_\omega}^{-1})^{\text{new}} = I_k + 2\sum_p \lambda_{\text{JJ}}(h(\boldsymbol{\Psi}_p, \boldsymbol{\rho}_p, a, \Sigma_{q_\omega}, \boldsymbol{\rho}_q))\boldsymbol{\Psi}_p^T\boldsymbol{\Psi}_p,$$

$$(\Sigma_{q_\omega}^{-1}\boldsymbol{\mu}_{q_\omega})^{\text{new}} = (\sum_t^T \boldsymbol{\Psi}_{v_t}^T)$$
$$- T\left[\sum_p^P\{\frac{1}{2} + 2(\boldsymbol{\rho}_p - a)\lambda_{\text{JJ}}\{h(\boldsymbol{\Psi}_p, \boldsymbol{\rho}_p, a, \Sigma_{q_\omega}, \boldsymbol{\rho}_q)\}\}\boldsymbol{\Psi}_p^T\right]$$

$$a^{\text{new}} = a + \frac{-1 + \frac{P}{2}}{2}$$
$$+ \sum_p \lambda_{\text{JJ}}\{h(\boldsymbol{\Psi}_p, \boldsymbol{\rho}_p, a, \Sigma_{q_\omega}, \boldsymbol{\rho}_q)\}$$
$$\times (\boldsymbol{\Psi}_p\boldsymbol{\mu}_{q_\omega} + \boldsymbol{\rho}_p) - a\lambda_{\text{JJ}}\{h(\boldsymbol{\Psi}_p, \boldsymbol{\rho}_p, a, \Sigma_{q_\omega}, \boldsymbol{\rho}_q)\}$$

That is the EM algorithm can be written:

$$\left((\Sigma_{q_\omega}^{-1})^{\text{new}}, (\Sigma_{q_\omega}^{-1}\boldsymbol{\mu}_{q_\omega})^{\text{new}}, a^{\text{new}}\right) = \sum_p^P g(\boldsymbol{\Psi}_p, \boldsymbol{\rho}_p, \Sigma_{q_\omega}^{-1}, \Sigma_{q_\omega}^{-1}\boldsymbol{\mu}_{q_\omega}, a).$$

As noted in [8] when an EM algorithm can be written as a fixed point update over a sum, then the Robbins Monro algorithm can be applied. Allowing updates of the form (p is chosen randomly):

$$(\Sigma_{q_\omega}^{-1})^{(\text{s})}, (\Sigma_{q_\omega}^{-1}\boldsymbol{\mu}_{q_\omega})^{(\text{s})}, a^{(\text{s})}$$
$$= (1 - \Delta_s)\left((\Sigma_{q_\omega}^{-1})^{(\text{s}-1)}, (\Sigma_{q_\omega}^{-1}\boldsymbol{\mu}_{q_\omega})^{(\text{s}-1)}, a^{(\text{s}-1)}\right)$$
$$+ \Delta_s g(\boldsymbol{\Psi}_p, \boldsymbol{\rho}_p, (\Sigma_{q_\omega}^{-1})^{(\text{s}-1)}, (\Sigma_{q_\omega}^{-1}\boldsymbol{\mu}_{q_\omega})^{(\text{s}-1)}, a^{(\text{s}-1)}).$$

where Δ is a slowly decaying Robbins Monro sequence ([32]) with $\Delta_1 = 1$ (meaning no initial value of $(\Sigma_{q_\omega}^{-1})^{(0)}, (\Sigma_{q_\omega}^{-1}\boldsymbol{\mu}_{q_\omega})^{(0)}, a^{(0)}$) is needed. For large P this algorithm is many times faster than the generic EM algorithm. Note that (unusually) the Robbins Monro algorithm is applied to the softmax of a large categorical variable and not to individual records under a conditionally independent assumption.

There are other variational bounds that may be considered for this problem most notably the tilted bound [19]. For the tilted bound the known fixed point algorithms are not guaranteed to be stable and are not always stable in practice [29, 34] so extra methods such as line searches would need to be considered.the tilted bound also does not decompose into a sum. We do not further consider alternative bounds.

The computational cost of this algorithm depends on the number of products P linearly and the embedding size K cubicly, if P and K are modest it can take less than a second making it potentially deployable at prediction time. In practice we found the cost of large P might be prohibitive due to the sums over all P embeddings,

in these cases a variational auto-encode described in the next section, is to be preferred.

6.4 Next Item Prediction

The predictive distribution required to do next item prediction is also not trivial in this case, i.e. approximating:

$$p(v_{u,T+1}|v_{u,1},..,v_{u,T}) = \int p(v_{u,T+1}|\boldsymbol{\omega},\boldsymbol{\Psi},\boldsymbol{\rho})p(\boldsymbol{\omega}|v_{u,1},.v_{u,T})d\boldsymbol{\omega}_u$$

is not trivial even if $p(\boldsymbol{\omega}|v_{u,1},..v_{u,T_u})$ is approximated with a Gaussian distribution $\boldsymbol{\omega}_u|v_1,..v_T \sim \mathcal{N}(\boldsymbol{\mu}_{q_\omega},\Sigma_{q_\omega})$. We are interested in computing:

$$p(v_{n+1}|v_1,...v_n) \approx \mathop{\mathbb{E}}_{q(\boldsymbol{\omega})}\left[\frac{\exp(\boldsymbol{\Psi}_v\boldsymbol{\omega}+\boldsymbol{\rho})}{\sum_{v'}\exp(\boldsymbol{\Psi}_{v'}\boldsymbol{\omega}+\boldsymbol{\rho})}\right].$$

We considered using a Monte Carlo based approximation, first by drawing S samples:

$$\boldsymbol{\omega}^{(s)} \sim \mathcal{N}(\boldsymbol{\mu}_{q_\omega},\Sigma_{q_\omega}),$$

$$p(v_{n+1}|v_1,...v_n) \approx \frac{1}{S}\sum_s^S \frac{\exp(\boldsymbol{\Psi}_v\boldsymbol{\omega}^{(s)}+\boldsymbol{\rho})}{\sum_{v'}\exp(\boldsymbol{\Psi}_{v'}\boldsymbol{\omega}^{(s)}+\boldsymbol{\rho})},$$

as well as using a number of fast approximations such as:

$$p(v_{n+1}|v_1,...v_n) \approx \frac{\exp(\boldsymbol{\Psi}_v\boldsymbol{\mu}_{q_\omega}+\boldsymbol{\rho})}{\sum_{v'}\exp(\boldsymbol{\Psi}_{v'}\boldsymbol{\mu}_{q_\omega}+\boldsymbol{\rho})}.$$

while we investigated more complex approximations (such as normalizing the exponential of the lower bound) we did not find they helped in practice.