Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence (AAAI-16)

Collective Noise Contrastive Estimation for Policy Transfer Learning

Weinan Zhang[†], **Ulrich Paquet**[‡], **Katja Hofmann**[‡]
[†]University College London, [‡]Microsoft Research

Abstract

We address the problem of learning behaviour policies to optimise online metrics from heterogeneous usage data. While online metrics, e.g., click-through rate, can be optimised effectively using exploration data, such data is costly to collect in practice, as it temporarily degrades the user experience. Leveraging related data sources to improve online performance would be extremely valuable, but is not possible using current approaches. We formulate this task as a policy transfer learning problem, and propose a first solution, called collective noise contrastive estimation (collective NCE). NCE is an efficient solution to approximating the gradient of a log-softmax objective. Our approach jointly optimises embeddings of heterogeneous data to transfer knowledge from the source domain to the target domain. We demonstrate the effectiveness of our approach by learning an effective policy for an online radio station jointly from user-generated playlists, and usage data collected in an exploration bucket.

Introduction

Interactive systems, such as web search engines, news recommender systems, and online radio stations, face the problem of finding a behaviour policy (e.g., which search results to show, news items to recommend, or songs to play) that optimises user satisfaction, typically measured in terms of some online metrics (e.g., positive feedback ratio, click-through rate) (Zhao, Zhang, and Wang 2013; Li et al. 2010). Contextual bandit approaches allow learning such behaviour policies directly from user interactions while balancing exploration (of new policies) and exploitation (of known good policies) (Li et al. 2010). However, learning a policy from scratch, by exploring and exploiting feedback obtained from real users, is a dangerous game: by the time a good policy is learned, all users would have abandoned the system. One is therefore forced to collectively leverage other available data sources to bootstrap a good initial policy. Conversely, much work has focused on learning from explicitly labelled data (Koren, Bell, and Volinsky 2009; Mnih and Salakhutdinov 2007), or user-provided example policies (Chen et al. 2012a). However, these approaches do not directly optimise online performance, and it is currently not clear how to bridge the gap between these approaches.

In this paper we formalise the task of learning a behaviour policy from heterogeneous data sources as a *policy transfer learning* problem. Given a set of exploration data, and additional heterogeneous data sources (e.g., user-provided example policies), how can the additional data be used to guide the policy learned from exploration data? As a running example, and for the empirical part of our work, we focus on the particularly challenging task of policy learning for online radio stations. An online radio station (referred to as *agent*) like that of Xbox Music is driven by a policy that stochastically streams music according to a probability $P(\text{next track}|\text{context})$.[1] Whilst listening to music, the listener provides partial feedback by skipping songs (low reward), or fully listening to songs (high reward). More formally, consider a context space I (e.g., the currently playing artist), actions in set J (e.g., artists to play next), and a reward signal r (user satisfaction with action j given context i). Following standard contextual bandit formulations (Li et al. 2010; 2015), in each round the environment selects the context i and the correponding reward vector $\overrightarrow{r_i}$ across all actions J, and provides i to the agent. The agent selects an action j, and sends it to the environment. The environment reveals $\overrightarrow{r_i}[j]$, the reward for action j only, in response (denoted as r for simplicity). Our goal is to learn an agent behaviour policy that has high expected reward.

We assume two types of data sources available for learning a policy with binary reward $r \in \{1, -1\}$: first, positive examples $D_P = \{(i, j, r = 1)\}$ – these are user-provided positive examples of good policies, here obtained in the form of user generated playlists; second, exploration data in the form of $(i, j, r, p_D(j|i))$, where $p_D(j|i)$ is the probability of selecting the action j given context i under the data collection policy.[2]

[1]For the purpose of this work we focus on individual transitions and model these as independent of longer-term history. Dependency on history could be extended by considering longer sequences as context, or through extension to the full MDP formulation. Also note that we focus on finding a good policy using existing data, corresponding to the "exploration first" scenario that is particularly relevant in practical applications.

[2]We assume exploration data was collected before training using an arbitrary data collection policy. Solutions for estimating $p_D(j|i)$ for unknown or non-stochastic exploration policies are proposed in (Langford, Strehl, and Wortman 2008; Strehl et al. 2010).

Given these datasets, our goal is then to learn a policy $P(j|i, r = 1)$ to maximise the expected policy value V, particularly, the expected reward directly calculated from the user feedback $V = \mathbb{E}_i[\mathbb{E}_{P(j|i,r'=1)}[\overrightarrow{r_i}[j]]]$. Within this framework, two optimisation objectives are studied: one is the joint data generation likelihood of the observations from two data sources; the second links the data generation to the unbiased expectation of the policy value with inverse propensity scores (IPS). Modelled as a softmax selection policy, the costly calculation of the object gradient is overcome via collective noise contrastive estimation (collective NCE) on both domains. We find the proposed solution to be highly efficient.

Our empirical study based on two real-world music usage data sources from Xbox Music shows excellent performance in terms of both data generation likelihood and expected policy value from successfully transferring knowledge from user generated playlists to their radio listening behaviour.

Contributions. (i) To our knowledge, this is the first work to leverage related data sources to optimise online metrics within a novel framework of policy transfer learning. (ii) Based on our framework, two optimisation objectives are studied and solved by efficient collective NCE. In particular, the second objective directly enables the optimisation of the IPS policy value via lower bound maximisation. (iii) This is the first work that extends the NCE algorithms to the applications of real-world interactive recommender systems (here: music recommendation).

Related Work

Online Learning for Recommender Systems. In interactive recommender system applications, users interact with the recommended items and the recommender system could perform online learning from user feedback to refine its recommendation to the specific user (Zhao, Zhang, and Wang 2013). These applications are ideally suited for contextual bandit algorithms, a type of reinforcement learning problem that focuses on balancing exploitation and exploration (Li et al. 2010; Zhao, Zhang, and Wang 2013). Seen as a reinforcement learning problem, (Liebman, Saar-Tsechansky, and Stone 2015) proposed an approach to learning sequences of songs for online radio stations with the focus on longer-term learning and planning. Our work also focuses on context-aware sequence transitions but leverages heterogeneous feedback for transfer learning.

Offline Policy Evaluation. As pointed out in (Li 2015), direct online evaluation and optimisation is expensive and risky. However, it is cheap and risk-free if policy can be optimised and evaluated using historic data that was previously collected using another policy. (Li et al. 2011) proposed to use historic data for unbiased offline evaluation using experience replay and rejection sampling. Prerequisites of this approach are that the exploration policy is known, and that it has sufficiently explored all actions in the support of the evaluated policy (Langford, Strehl, and Wortman 2008). For cases where historic data is collected using a biased (non-uniform) or non-stationary policy, (Dudík et al. 2012) suggested an adaptive rejection sampling approach. For case where the exploration policy is unknown, an evaluation scheme with the estimated propensity scores and a lower bound of the data observation probability was proposed in (Strehl et al. 2010).

Transfer Learning for Reinforcement Learning. Transfer learning has been proven to work on a variety of problems (Pan and Yang 2010), including reinforcement learning (Ramon, Driessens, and Croonenborghs 2007; Taylor and Stone 2007; 2009). As pointed out in (Taylor and Stone 2009), there are different goals of transfer learning in reinforcement learning. Out of these, our goal is the "jumpstart", i.e., to achieve high initial performance on the target task (online radio) by using data from a different task (user generated playlists). We specifically focus on the questions of how knowledge can be transferred in a policy learning setting, and whether such transferred knowledge lead to better-performing policies.

Model

Softmax-based Stochastic Policy

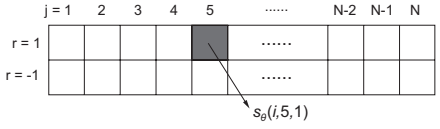
Our goal is to learn an effective behaviour policy for streaming online radio. Following (Chen et al. 2012a), we define our radio policy using a softmax distribution (corresponding to the "single-point model" in (Chen et al. 2012a)). However, differently from this earlier work, we model the joint distribution of action and reward. This extension is the basis for integrating heterogeneous usage data as will be shown below. Specifically, for the given context i, the probability of the softmax policy playing a track of the artist j and receiving reward r is parameterised as

$$P_\theta(j, r|i) = e^{s_\theta(i,j,r)} \Big/ \sum_{r'} \sum_{j'} e^{s_\theta(i,j',r')} . \quad (1)$$

Here, $s_\theta(i, j, r)$ is the scoring function that measures the utility of playing a track by artist j given context i, as parameterised by θ (detailed in Eq. (2)). Context i is an index into a discrete set of states; for simplicity we index the currently playing artist. The partition function sums over all possible artists, which is around 10^6 in Xbox Music. An intuitive implementation[3] of the scoring function is

$$s_\theta(i, j, r) = r \cdot w_i^T w_j + b_j, \quad (2)$$

with $\theta = (w_i, w_j, b_j)$. This formulation represents each artist by an embedding $w_j \in \mathbb{R}^N$. It resembles a basic neural probabilistic language model (Mnih and Teh 2012) with the current words (i.e., the current artist) and the next word (the next artist) embedded as latent feature vectors. The bias term b_j captures the popularity of artist j in the training data.



Our model can be understood as selecting a cell (j, r) in a two-row grid as illustrated above. The predicted user feedback $P_\theta(r|i, j) = 1/(1 + e^{-2rw_i^T w_j})$ is a logistic function

[3]Extensions of our method to other scoring functions, e.g., incorporating features like user types and artist genres, are straightforward (Chen et al. 2012b).

that normalises to one over the two rows in the above grid. It *only* depends on the inner product of the artists' latent feature vectors, and as such, models the feedback on the quality of the transition. To recover the playlist-only scenario in (Chen et al. 2012a), we can condition on $r = 1$. Because in this setting $P(r|i) = P(r)$ (the desired reward is independent of the context) and $P(r)$ is constant, we obtain

$$P_\theta(j|i, r = 1) = e^{s_\theta(i,j,1)} \,/\, \textstyle\sum_{j'} e^{s_\theta(i,j',1)} \,. \tag{3}$$

We use this formulation to model estimates for the playlist data, while the formulation as the joint density (Eq. (2)) is used to model radio data. Based on this model, we next define our objectives.

Objectives

In our work, we explore two supervised learning objectives. The first is the data generation likelihood, which is normally used in playlist generation tasks (Chen et al. 2012a). The second is the expected policy value, which is commonly used in reinforcement learning (Li 2015; Sutton and Barto 1998).

Objective 1: Data Generating Likelihood Maximisation. The data generation setting is similar to the text generation process for language models (Lavrenko and Croft 2001). For *playlist data*, given the training data $D_P = \{(i, j, 1)\}$, we want to find the optimal policy parameterised by θ to maximise the model's average likelihood $\mathcal{L}_P(P_\theta)$ to generate each triple in D_P:

$$\mathcal{L}_P(P_\theta) = \textstyle\prod_{(i,j,1)\in D_P} P_\theta(j|i, r = 1) \,. \tag{4}$$

For *radio data*, given the training data $D_R = \{(i, j, r)\}$, we want to find the optimal θ to maximise the model's likelihood $\mathcal{L}_D(\theta)$ to generate D_R:

$$\mathcal{L}_R(P_\theta) = \textstyle\prod_{(i,j,r)\in D_R} P_\theta(j, r|i) \,. \tag{5}$$

With datasets D_P and D_R, we can find θ to collectively optimise their combined likelihoods through

$$\begin{aligned} &\max_\theta \frac{\alpha}{|D_R|} \log \mathcal{L}_R(P_\theta) + \frac{1-\alpha}{|D_P|} \log \mathcal{L}_P(P_\theta) \\ =&\max_\theta \frac{\alpha}{|D_R|} \sum_{(i,j,r)\in D_R} \log \frac{e^{s_\theta(i,j,r)}}{\sum_{r'} \sum_{j'} e^{s_\theta(i,j',r')}} + \\ &\frac{1-\alpha}{|D_P|} \sum_{(i,j,1)\in D_P} \log \frac{e^{s_\theta(i,j,1)}}{\sum_{j'} e^{s_\theta(i,j',1)}} \,, \end{aligned} \tag{6}$$

with α being the combination hyperparameter to control the relative importance of the radio task. We emphasise that our goal is high performance on the task of generating radio data. The hypothesis underlying the combined objective is that training on a combination of playlist and radio data can improve performance through smoothing and regularisation (c.f., Eq. (16)).

Objective 2: IPS Policy Value Maximisation. Much like reinforcement learning, where the policy takes an action (samples an artist) according to the state (context) and then observes the action reward (user feedback), we can estimate the expected reward of the policy P_θ, i.e., $\mathbb{E}_i[\mathbb{E}_{P_\theta(j|i,r'=1)}[\overrightarrow{r_i}[j]]]$, based on the historic radio data D_R. As D_R was generated from an underlying data collection policy, denoted as P_D, it is necessary to eliminate the data bias from P_D using inverse propensity score (IPS) to perform the unbiased estimation of the expected policy value $\hat{V}_{\text{ips}}(P_\theta)$.

$$\begin{aligned} \hat{V}_{\text{ips}}(P_\theta) &= \frac{1}{|D_R|} \sum_{(i,j,r)\in D_R} \frac{r P_\theta(j|i, r' = 1)}{P_D(j|i)} \\ &= \frac{1}{|D_R|} \sum_{(i,j,r)\in D_R} \frac{r}{P_D(j|i)} \frac{e^{s_\theta(i,j,1)}}{\sum_{j'} e^{s_\theta(i,j',1)}} \end{aligned} \tag{7}$$

The data policy P_D in this evaluation scheme should not have smaller support than the test policy P_θ. In order to ease the parameter gradient calculation we define an approximation of the IPS policy value:

$$\tilde{V}_{\text{ips}}(P_\theta) = \frac{1}{|D_R|} \sum_{(i,j,r)\in D_R} \frac{r}{P_D(j|i)} \log \frac{e^{s_\theta(i,j,1)}}{\sum_{j'} e^{s_\theta(i,j',1)}} \,. \tag{8}$$

Function $\log x$ monotonously increases w.r.t. x and $x > \log x$ for any $x \in \mathbb{R}^+$, and $P_\theta(j|i, r' = 1)$ is nonnegative. Assuming $r \geq 0$ through a linear shift in the reward, then $\hat{V}_{\text{ips}}(P_\theta) > \tilde{V}_{\text{ips}}(P_\theta)$, i.e. $\tilde{V}_{\text{ips}}(P_\theta)$ is a **lower bound** to the true IPS policy value. With this IPS variant $\tilde{V}_{\text{ips}}(P_\theta)$, we can find θ to collectively maximise $\tilde{V}_{\text{ips}}(P_\theta)$ on radio dataset and the likelihood $\mathcal{L}_P(P_\theta)$ on playlist dataset by

$$\begin{aligned} &\max_\theta \alpha \tilde{V}_{\text{ips}}(P_\theta) + \frac{1-\alpha}{|D_P|} \log \mathcal{L}_P(P_\theta) \\ =&\max_\theta \frac{\alpha}{|D_R|} \sum_{(i,j,r)\in D_R} \frac{r}{P_D(j|i)} \log \frac{e^{s_\theta(i,j,1)}}{\sum_{j'} e^{s_\theta(i,j',1)}} + \\ &\frac{1-\alpha}{|D_P|} \sum_{(i,j,1)\in D_P} \log \frac{e^{s_\theta(i,j,1)}}{\sum_{j'} e^{s_\theta(i,j',1)}} \,. \end{aligned} \tag{9}$$

Maximising $\tilde{V}_{\text{ips}}(P_\theta)$ maximises a lower bound on $\hat{V}_{\text{ips}}(P_\theta)$ and transfers knowledge from the playlist data.

Parameter Updating with Noise Contrastive Estimation

Let $J_P^{(i,j,1)}(\theta)$ denote an individual term in the rightmost (playlist) sum in Eqs. (6) and (9). In order to maximise the objectives in (6) and (9), the gradient with respect to θ is

$$\frac{\partial}{\partial\theta} J_P^{(i,j,1)}(\theta) = \frac{1-\alpha}{|D_P|} \cdot \frac{\partial}{\partial\theta} \log \frac{e^{s_\theta(i,j,1)}}{\sum_{j'} e^{s_\theta(i,j',1)}} \tag{10}$$

for each playlist transition observation $(i, j, 1)$ for both objectives. For each radio transition observation (i, j, r), where $J_R^{(i,j,r)}(\theta)$ denotes a term in the leftmost (radio) sum in Eqs. (6) and (9), the gradient with respect to θ is

$$\frac{\partial}{\partial\theta} J_R^{(i,j,r)}(\theta) = \frac{\alpha}{|D_R|} \cdot \frac{\partial}{\partial\theta} \log \frac{e^{s_\theta(i,j,r)}}{\sum_{r'} \sum_{j'} e^{s_\theta(i,j',r')}} \tag{11}$$

for objective 1, and

$$\frac{\partial}{\partial\theta} J_R^{(i,j,r)}(\theta) = \frac{\alpha}{|D_R|} \cdot \frac{r}{P_D(j|i)} \frac{\partial}{\partial\theta} \log \frac{e^{s_\theta(i,j,1)}}{\sum_{j'} e^{s_\theta(i,j',1)}} \tag{12}$$

for objective 2. In the above three equations, the gradient calculation on the softmax function

$$\frac{\partial}{\partial\theta}\log\frac{e^{s_\theta(i,j,r)}}{\sum_{j'}e^{s_\theta(i,j',r)}}=\frac{\partial s_\theta(i,j,r)}{\partial\theta}-\mathbb{E}_{P_\theta(j'|i,r)}\Big[\frac{\partial s_\theta(i,j',r)}{\partial\theta}\Big] \tag{13}$$

needs to iterate over all possible artist j', which is a very costly operation. In our work, we follow (Mnih and Teh 2012) to leverage the noise contrastive estimation (NCE) (Gutmann and Hyvärinen 2012) to employ an efficient approximated gradient calculation.

The basic tenet of NCE is to define a loss function to quantify how likely the policy will separate a data point (i,j,r) from k noise data points $\{(i,j_m,r)\}_{m=1}^k$ with j_m generated from a known noise probabilistic distribution $P_n(j_m)$. The log probability of distinguishing data from noise is $\mathcal{L}_{\text{NCE}}^{(i,j,r)}(\theta)=$

$$\log\frac{P_\theta(j|i,r)}{P_\theta(j|i,r)+kP_n(j)}+\sum_{m=1}^{k}\log\frac{kP_n(j_m)}{P_\theta(j_m|i,r)+kP_n(j_m)}, \tag{14}$$

and its gradient

$$\frac{\partial}{\partial\theta}\mathcal{L}_{\text{NCE}}^{(i,j,r)}(\theta)=\frac{kP_n(j)}{e^{s_\theta(i,j,r)}+kP_n(j)}\frac{\partial s_\theta(i,j,r)}{\partial\theta}-\sum_{m=1}^{k}\frac{e^{s_\theta(i,j_m,r)}}{e^{s_\theta(i,j_m,r)}+kP_n(j_m)}\frac{\partial s_\theta(i,j_m,r)}{\partial\theta} \tag{15}$$

can be efficiently calculated. It is proven in (Gutmann and Hyvärinen 2012; Mnih and Teh 2012) that when $k\to+\infty$, the gradient $\frac{\partial}{\partial\theta}\mathcal{L}_{\text{NCE}}^{(i,j,r)}(\theta)\to\frac{\partial}{\partial\theta}\log\frac{e^{s_\theta(i,j,r)}}{\sum_{j'}e^{s_\theta(i,j',r)}}$ in Eq. (13). With NCE as a tool to efficiently calculate the parameter gradients, we perform SGD to update our model parameters, i.e., w_i, w_j, b_j in Eq. (2), with η learning rate, as $\theta\leftarrow\theta+\eta\frac{\partial}{\partial\theta}\mathcal{L}_{\text{NCE}}^{(i,j,r)}(\theta)$.[4]

During training, the model is fed the data points (i,j,r) from the two data sources and the two NCE components are trained on playlist and radio data collectively. We call the approach Collective NCE.

Regularisation. Our basic regularisation includes the L2 norm of the latent feature vector w_j of each artist, as well as the artist popularity bias b_j. Moreover, in order to transfer the knowledge from the source domain (playlist) to the target domain (radio), we propose a regularisation term to push the latent feature vectors of the same artist on the two domains towards each other.

$$\begin{aligned}\text{Loss}=&-(\text{Objective 1 or 2})+\lambda_1\sum_j(||w_j^{\langle p\rangle}||_2^2+{b_j^{\langle p\rangle}}^2)\\&+\lambda_1\sum_j(||w_j^{\langle r\rangle}||_2^2+{b_j^{\langle r\rangle}}^2)+\lambda_2\sum_j||w_j^{\langle p\rangle}-w_j^{\langle r\rangle}||_2^2\end{aligned} \tag{16}$$

In practice, this kind of "soft" bounding of an artist's feature vectors across the two data sources performs much better than the "hard" bounding that would be obtained by setting λ_2 to infinity. This is because the behaviour underlying playlist and radio data are different. We preserve such differences to obtain a good fit with the data while enabling knowledge transfer.

[4]See the supplementary material for a detailed derivation at http://research.microsoft.com/apps/pubs/?id=258572.

Experiments and Results

We report on two experiments, designed to empirically assess our approach. The first (*data generation*) aligns with our objective 1, and measures to what degree a generative model of the radio data can be learned from different data sources, and whether knowledge transfer may be possible. The second experiment (*policy learning*) assesses performance in terms of our key metric, the IPS-weighted (unbiased) estimate of actual online performance.

Dataset Description. We test the proposed models on two proprietary datasets collected from Xbox Music, an online commercial music radio service. Both datasets are generated via a uniform sampling from the artists and playlists/radio episodes with no fewer than 10 observations from the original huge data log. All data was collected in 2014.

The first dataset, D_P contains the playlists of tracks generated by users. It contains 722,741 track transitions from 20,262 user generated playlists with 1,808 artists. The second dataset, D_R consists of 97,605 track transition sequences with 1,440 artists generated from the radio system and the users' feedback on each transition in the form of normal listening transitions and skips. The two datasets share 1,034 artists due to the different data distributions although they are generated from the same radio service. Both for the playlist dataset and radio dataset, we randomly sample 10,000 transitions as the validation data and test data respectively, while the remainder is used as training data. For the test radio dataset, in order to protect sensitive information, we balance the numbers of normal listening and skip observations. Note that although the reported experiment results are based on a small portion of the whole data on Xbox Music, the set of all artists is of order 10^6, making the estimation intractable in brute-force manner as was applied in (Chen et al. 2012a).

Training Setting. For the playlist data, user feedback on every transition is regarded as positive, scored as $+1$. This means that playlists are considered as user-provided positive example policies. For the radio data, the positive user feedback, i.e., normal listening, is scored as $+1$, while the negative user feedback, i.e., the skip, is scored as 0.[5] The latent feature vectors are all in 32 dimensions[6]. Both training datasets are randomly shuffled before being fed into the model to avoid spurious effects due to sequential correlations.

During training, for each observation (i,j,r), k noise artists $\{j_m\}_{m=1}^k$ are sampled to calculate the NCE gradient as shown in Eq. (15). The probability of sampling a noise artist is proportional to the artist's frequency (popularity) in

[5]In the training stage, we shift the negative reward from -1 to 0 in order to make the lower bound in Eq. (8) hold for all cases. In the test stage, the negative reward is still -1.

[6]We also conducted experiments with 10D and 64D latent feature vectors. We found the 32D feature vector setting to achieve a good balance between effectiveness and efficiency.

the training data. We tune hyperparameters on the validation set, before performance is evaluated on the test data.

Evaluation. For the data generation task, our goal is to learn a good generative model of the radio data. Following (Chen et al. 2012a), we assess this goal in terms of the averaged log-likelihood per transition, i.e., the value of the first term (without α) in Eq. (6) on the test data. For the policy value maximisation, we follow (Strehl et al. 2010), and adopt IPS-weighted unbiased evaluation, i.e., we measure the value of Eq. (7).

Since our main goal is to model and learn radio policies, we report the performance corresponding to the radio component (based on D_R) of objectives 1 and 2, and evaluate whether the knowledge transferred from user-generated playlist data can successfully improve radio quality. For objective 2, in order to perform unbiased evaluation, the policy's recommendations for a given context are restricted to the artists that are in the support of the data collection policy[7]. The data collection policy applies a pseudo-random selection from a support set of similar artists and its propensity scores are estimated per artist-pair from the training data.

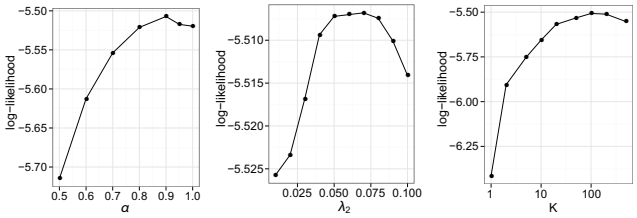
Compared Algorithms. Besides our proposed collective NCE solution (denoted as NCE-COLLECTIVE), we compare another 5 baseline policies. The RANDOM policy uniformly samples an artist from the available (restricted) artist set. The POPULARITY policy samples artists with probability proportional to their frequency in the training data. These two policies are context-free. NCE-RADIO is our NCE model trained only on the radio dataset, which corresponds to the special case of $\alpha = 1$ in Eqs. (6) and (9). To check whether a model learned purely from the playlist data can be directly adopted to model the radio data, we trained NCE-PLAYLIST with playlist data only, i.e., $\alpha = 0$ in Eqs. (6) and (9), and tested it on the radio task. For the IPS policy learning task, we added a further policy that is often applied in practice, SAMEARTIST, which only plays tracks from the same (currently playing) artist. We also experimented with the importance sampling solution for log-softmax gradient calculation, as proposed in (Bengio, Senécal, and others 2003), but found it to be highly unstable in our experiments, just as reported in (Mnih and Teh 2012).

Data Generation Results. The log-likelihood performance of generating the radio test data from the compared policies is provided in Table 1. We can see the obvious impact of the NCE-RADIO policy, which substantially outperforms all baseline non-NCE policies. This indicates the feasibility of leveraging the NCE algorithm for efficient softmax gradient calculation in recommender system training. More importantly, NCE-COLLECTIVE further improves the log-likelihood from NCE-RADIO, which shows the effectiveness of transferring the knowledge of artist similarity learned from the playlist data to the radio data. In addition, NCE-PLAYLIST, which is trained with only playlist data, performs worse than RANDOM. This means that simply moving the learned parameters from the playlist domain to the radio domain does not work at all.

[7]Without this restriction, the policy might learn to simply avoid recommending the artists associated with negative reward, but fail to capture the artists' real similarity.

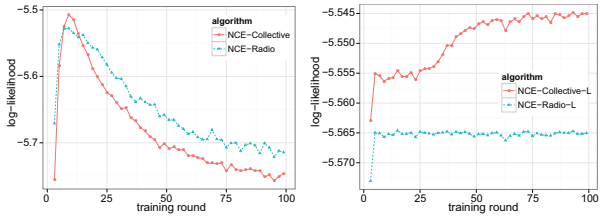
Table 1: Data generation performance comparison.

Algorithm	log-likelihood
Random	-7.7932
Popularity	-5.8009
NCE-Playlist	-10.3978
NCE-Radio	-5.5197
NCE-Collective	**-5.5072**



(a) α in Eq. (6) (b) λ_2 in Eq. (16) (c) k in Eq. (14)

Figure 1: Log-likelihood of generating the test data with different hyperparameters.



(a) NCE without λ_1 reg. (b) NCE with λ_1 reg.

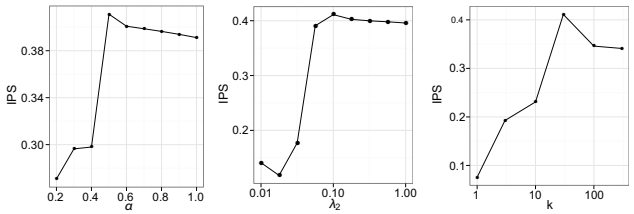
Figure 2: Log-likelihood convergence with respect to the training rounds.

Figure 1 shows the log-likelihood performance w.r.t. the tuning of the three hyperparameters of our collective NCE model. We observe that the log-likelihood performance changes smoothly as the hyperparameter values change and it is insensitive to the variations in hyperparameter values around the peak regions. This shows that the optimal hyperparameters can be soundly obtained. Specifically, we obtain the empirically optimal settings of radio-task weight $\alpha = 0.9$, inter-domain regularisation term $\lambda_2 = 0.07$, and noise-data ratio $k = 100$. There is a slight performance drop when $k > 100$, which is also reported in (Mnih and Teh 2012). The reason could be some artists sampled as noise points are actually preferred by the user when performing a large noise sampling.

Figure 2 further shows that the NCE algorithm convergences over training rounds. Here NCE-X algorithms set $\lambda_1 = 0$ in Eq. (16), while NCE-X-L algorithms sets $\lambda_1 = 0.05$. We can observe that NCE-X algorithms overfit easily although they reach peak performance quickly. On the contrary, NCE-X-L algorithms relatively smoothly approach the optimal performance (at the expense of more training rounds), which shows a better model generalisation ability using λ_1 regularisation.

Table 2: IPS based policy evaluation comparison.

Algorithm	IPS Value
Random	0.0687
Popularity	0.0747
SameArtist	-0.3088
NCE-Playlist	0.0695
NCE-Radio	0.3912
NCE-Collective	**0.4111**



(a) α in Eq. (9) (b) λ_2 in Eq. (16) (c) k in Eq. (14)

Figure 3: IPS on test data with different hyperparameters.

Policy Learning. Performance in terms of the IPS-based policy value is presented in Table 2. It is clear that the NCE-RADIO and NCE-COLLECTIVE policies substantially improve over alternative policies, with the IPS value of 0.3912 and 0.4111, respectively. This shows that our approach is able to learn very good radio policies in IPS-based policy learning setting. Again, NCE-COLLECTIVE further outperforms NCE-RADIO in term of the IPS value, which verifies that the collective training on two domains helps learn good artist transition for online radio policies. The context-free policies RANDOM and POPULARITY obtain substantially lower IPS values, which shows the importance of the selecting artists in context in the radio scenario, instead of relying on, e.g., artist popularity. SAMEARTIST obtains a negative IPS score of -0.3088, which is surprising given that radio policies of this type are quite common in industry. It shows that users may get bored after listening to several tracks from the same artist (Crossen, Budzik, and Hammond 2002).

Figure 3 shows how the IPS values vary w.r.t. three hyperparameters in our model. We find the empirically optimal settings as the radio-task weight $\alpha = 0.5$, the inter-domain regularisation term $\lambda_2 = 0.1$, and the NCE noise sample number $k = 30$. For both hyperparameters α and λ_2, there is an obvious IPS ascent when tuning their values from 0 (brute-force or no transfer learning) to the optimal values (an appropriate degree of transfer), which shows the impact of the knowledge transfer in our scenario. After the hyperparameter values pass the optimal values, the performance drop is relatively gradual, which makes it easy to generalise model performance to the test data.

Artist Embedding Illustration. To illustrate the performance of our method, we train an NCE-COLLECTIVE model with 2-dimensional artist latent feature vectors and present the embedded artists in Figure 4. We can see that the artists are clustered consistently with their genres, although such genre information is never used in our model. Furthermore, some artists are heavily played, thus lose the transition

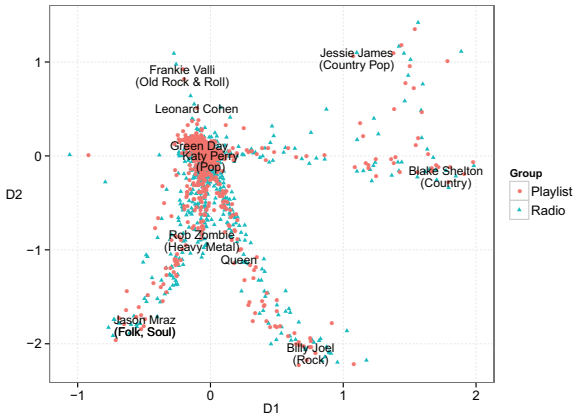


Figure 4: The 2D artist embedding trained by NCE-COLLECTIVE for objective 1, with some typical artists annotated. It is obvious that (i) artists with different genres get different clusters. For example, Jason Mraz (folk, soul) is on the left-bottom, Blake Shelton (country) is on the right side, etc. Some high-popularity stars with frequent playing everywhere, e.g., Green Day and Katy Perry, are embedded in the center cluster. (ii) The embedding landscapes of playlist data and radio data are quite similar, which suggests the feasibility of transferring the knowledge from the playlist side to the radio side.

preference even if they have informative genres, such as the famous rock band Green Day and pop singer Katy Perry, as shown by the embedding in the centre cluster. Moreover, some example recommended radio streams with the annotated artists in Figure 4 as the seed artist are provided in our supplementary material.

Conclusion

In this paper, we proposed a policy transfer learning framework for collectively training behaviour policies of interactive systems from heterogeneous usage data. To our knowledge, this is the first work applying noise contrastive estimation techniques to this setting, and we show that this framework can be used to effectively optimise a lower bound on an online performance objective (IPS policy value). At the same time, our method allows the joint training of two NCE models to transfer knowledge between different types of usage data. Our empirical results are obtained in the challenging domain of learning policies for online radio stations. Crucially, we show that our approach of optimising a lower bound on the IPS training objective results in excellent behaviour policies. We also show that models learned on playlist data only (as done in previous work) do not result in good online performance. However, best performance is achieved when transferring knowledge from playlist data to the radio policy using our joint optimisation approach. For future work, we aim to investigate different choices of sampling distribution for obtaining negative items in NCE training. Also, our focus has been on the "explore first" training scenario where playlist and exploration data are fixed before learning. Moving to policies that explicitly balance explo-

ration and exploitation is another promising direction.

Acknowledgements

We sincerely thank Thore Graepel and Noam Koenigstein for many helpful discussions, and Nir Nice and the Microsoft Recommendations Team for supporting this project.

References

Bengio, Y.; Senécal, J.-S.; et al. 2003. Quick training of probabilistic neural nets by importance sampling. In *AISTATS Conference*.

Chen, S.; Moore, J. L.; Turnbull, D.; and Joachims, T. 2012a. Playlist prediction via metric embedding. In *KDD*, 714–722. ACM.

Chen, T.; Zhang, W.; Lu, Q.; Chen, K.; Zheng, Z.; and Yu, Y. 2012b. Svdfeature: a toolkit for feature-based collaborative filtering. *The Journal of Machine Learning Research* 13(1):3619–3622.

Crossen, A.; Budzik, J.; and Hammond, K. J. 2002. Flytrap: intelligent group music recommendation. In *ACM IUI*, 184–185.

Dudík, M.; Erhan, D.; Langford, J.; and Li, L. 2012. Sample-efficient nonstationary policy evaluation for contextual bandits. In *Proceedings of the UAI*.

Gutmann, M. U., and Hyvärinen, A. 2012. Noise-contrastive estimation of unnormalized statistical models, with applications to natural image statistics. *The Journal of Machine Learning Research* 13(1):307–361.

Koren, Y.; Bell, R.; and Volinsky, C. 2009. Matrix factorization techniques for recommender systems. *Computer* (8):30–37.

Langford, J.; Strehl, A.; and Wortman, J. 2008. Exploration scavenging. In *Proceedings of the 25th international conference on Machine learning*, 528–535. ACM.

Lavrenko, V., and Croft, W. B. 2001. Relevance based language models. In *Proceedings of the 24th annual international ACM SIGIR conference on Research and development in information retrieval*, 120–127. ACM.

Li, L.; Chu, W.; Langford, J.; and Schapire, R. E. 2010. A contextual-bandit approach to personalized news article recommendation. In *Proceedings of the 19th international conference on World wide web*, 661–670. ACM.

Li, L.; Chu, W.; Langford, J.; and Wang, X. 2011. Unbiased offline evaluation of contextual-bandit-based news article recommendation algorithms. In *Proceedings of the fourth ACM international conference on Web search and data mining*, 297–306. ACM.

Li, L.; Chen, S.; Kleban, J.; and Gupta, A. 2015. Counterfactual estimation and optimization of click metrics in search engines: A case study. In *Proceedings of the 24th International World Wide Web Conference (WWW'14), Companion Volume*.

Li, L. 2015. Offline evaluation and optimization for interactive systems. In *Proceedings of the Eighth ACM International Conference on Web Search and Data Mining*, 413–414. ACM.

Liebman, E.; Saar-Tsechansky, M.; and Stone, P. 2015. DJ-MC: A reinforcement-learning agent for music playlist recommendation. In *AAMAS*, 591–599.

Mnih, A., and Salakhutdinov, R. 2007. Probabilistic matrix factorization. In *Advances in neural information processing systems*, 1257–1264.

Mnih, A., and Teh, Y. W. 2012. A fast and simple algorithm for training neural probabilistic language models. In *Proceedings of the 29th International Conference on Machine Learning (ICML-12)*, 1751–1758.

Pan, S. J., and Yang, Q. 2010. A survey on transfer learning. *Knowledge and Data Engineering, IEEE Transactions on* 22(10):1345–1359.

Ramon, J.; Driessens, K.; and Croonenborghs, T. 2007. Transfer learning in reinforcement learning problems through partial policy recycling. In *ECML*. Springer. 699–707.

Strehl, A.; Langford, J.; Li, L.; and Kakade, S. M. 2010. Learning from logged implicit exploration data. In *Advances in Neural Information Processing Systems*, 2217–2225.

Sutton, R. S., and Barto, A. G. 1998. *Introduction to reinforcement learning*. MIT Press.

Taylor, M. E., and Stone, P. 2007. Cross-domain transfer for reinforcement learning. In *Proceedings of the 24th international conference on Machine learning*, 879–886. ACM.

Taylor, M. E., and Stone, P. 2009. Transfer learning for reinforcement learning domains: A survey. *J. Mach. Learn. Res.* 10:1633–1685.

Zhao, X.; Zhang, W.; and Wang, J. 2013. Interactive collaborative filtering. In *Proceedings of the 22nd ACM international conference on Conference on information & knowledge management*, 1411–1420. ACM.