Recurrent Latent Variable Networks for Session-Based Recommendation

Panayiotis Christodoulou

Cyprus University of Technology

paa.christodoulou@edu.cut.ac.cy

27/8/2017

Overview

Motivation

- Recommender Systems (RS) constitute a key part of modern e-commerce websites.
- Recent study on RS has mainly focused on matrix factorization (MF) methods and neighborhood search-type models.
- Session-based recommendation cannot be properly addressed by such conventional methodologies.
- In session-based RS, recommendation is based only on the actions of a user during a specific browsing session.
- This gives rise to hard challenges that stem from the unavailability of rich user profiles (data sparsity).
- On the other hand, user session data constitute action sequences potentially entailing rich, complex, and subtle temporal dynamics.

Motivation

- We posit that, effective extraction of these underlying dynamics may facilitate addressing the problems stemming from data sparsity.
- In the past, Markov Decision Process (MDP) models have been used to this end, with only limited success.
- Recently, the Deep Learning breakthrough, especially, advances in Recurrent Neural Network (RNN) models, has inspired new, immensely promising lines of research in the field.
- Characteristically, Hidasi et al. (2016) employed gated recurrent unit (GRU) RNNs that are presented with data regarding the items a user selects and clicks on during a given session.

Proposed Approach: Main Rationale

- We seek to increase the capability of RNN-driven session-based RS to ameliorate the negative effects of data sparsity.
- To this end, we attempt to account for uncertainty, by means of Bayesian inference.
- Our proposed approach is founded upon the fundamental assumption that the hidden units of the postulated RNNs constitute latent variables imposed some appropriate prior distribution.
- Then, we use the training data to infer the corresponding posteriors.
- For scalability, we employ amortized variational inference (AVI):
- (i) Parameterizes the inferred posteriors via conventional neural networks (inference networks); and (ii) casts inference as optimization.

Proposed Approach: ReLaVaR

ReLaVaR Formulation

- We frame the session-based recommendation problem as a sequence-based prediction problem.
- We denote as $\{\boldsymbol{x}_i\}_{i=1}^{n}$ a user session; here, $\boldsymbol{x}_i$ is the ith clicked item, which constitutes a selection among m alternatives, and is encoded in the form of an 1-hot representation.
- We formulate session-based recommendation as the problem of predicting the score vector $\boldsymbol{y}_{i+1} = [y_{i+1,j}]_{j=1}^{m}$ of the available items with respect to the following user action, where $y_{i+1,j} \in \mathbb{R}$ is the predicted score of the jth item.
- At each time point, we select the top-k items (as ranked via $\boldsymbol{y}$) to recommend to the user.

Proposed Approach: ReLaVaR

ReLaVaR Formulation

- Formally, the recurrent units activation vector, $\boldsymbol{h}$, of a GRU are updated at time i according to the following expression:

$$\boldsymbol{h}_i = (1 - \boldsymbol{z}_i) \cdot \boldsymbol{h}_{i-1} + \boldsymbol{z}_i \cdot \hat{\boldsymbol{h}}_i \tag{1}$$

where

$$\boldsymbol{z}_i = \tau(\boldsymbol{W}_z \boldsymbol{x}_i + \boldsymbol{U}_z \boldsymbol{h}_{i-1}) \tag{2}$$

$$\hat{\boldsymbol{h}}_i = \tanh(\boldsymbol{W} \boldsymbol{x}_i + \boldsymbol{U}(\boldsymbol{r}_i \cdot \boldsymbol{h}_{i-1})) \tag{3}$$

$$\boldsymbol{r}_i = \tau(\boldsymbol{W}_r \boldsymbol{x}_i + \boldsymbol{U}_r \boldsymbol{h}_{i-1}) \tag{4}$$

and $\tau()$ is the logistic sigmoid function.

- ReLaVaR extends upon these design principles by rendering the developed GRU model amenable to Bayesian inference.

Proposed Approach: ReLaVaR

ReLaVaR Formulation

- We consider that the component recurrent unit activations are stochastic latent variables; we impose a simple prior distribution:

$$p(\boldsymbol{h}_i) = \mathcal{N}(\boldsymbol{h}_i|\boldsymbol{0}, \boldsymbol{I}) \tag{5}$$

- Then, we seek an efficient means of inferring the corresponding posterior distributions, given the available training data.
- We draw inspiration from AVI; we postulate that the sought posteriors, $q(\boldsymbol{h})$, approximately take the form of Gaussians with means and isotropic covariance matrices parameterized via GRU networks.

$$q(\boldsymbol{h}_i; \boldsymbol{\theta}) = \mathcal{N}(\boldsymbol{h}_i|\boldsymbol{\mu}_{\boldsymbol{\theta}}(\boldsymbol{x}_i), \sigma^2_{\boldsymbol{\theta}}(\boldsymbol{x}_i)\boldsymbol{I}) \tag{6}$$

Proposed Approach: ReLaVaR

ReLaVaR Formulation

- Hence, we have:

$$[\boldsymbol{\mu}_{\boldsymbol{\theta}}(\boldsymbol{x}_i), \log \sigma^2_{\boldsymbol{\theta}}(\boldsymbol{x}_i)] = (1 - \boldsymbol{z}_i) \cdot [\boldsymbol{\mu}_{\boldsymbol{\theta}}(\boldsymbol{x}_{i-1}), \log \sigma^2_{\boldsymbol{\theta}}(\boldsymbol{x}_{i-1})] + \boldsymbol{z}_i \cdot \hat{\boldsymbol{h}}_i \quad (7)$$

where

$$\boldsymbol{z}_i = \tau(\boldsymbol{W}_z \boldsymbol{x}_i + \boldsymbol{U}_z [\boldsymbol{\mu}_{\boldsymbol{\theta}}(\boldsymbol{x}_{i-1}), \log \sigma^2_{\boldsymbol{\theta}}(\boldsymbol{x}_{i-1})]) \quad (8)$$

$$\hat{\boldsymbol{h}}_i = \tanh(\boldsymbol{W} \boldsymbol{x}_i + \boldsymbol{U}(\boldsymbol{r}_i \cdot [\boldsymbol{\mu}_{\boldsymbol{\theta}}(\boldsymbol{x}_{i-1}), \log \sigma^2_{\boldsymbol{\theta}}(\boldsymbol{x}_{i-1})])) \quad (9)$$

and

$$\boldsymbol{r}_i = \tau(\boldsymbol{W}_r \boldsymbol{x}_i + \boldsymbol{U}_r [\boldsymbol{\mu}_{\boldsymbol{\theta}}(\boldsymbol{x}_{i-1}), \log \sigma^2_{\boldsymbol{\theta}}(\boldsymbol{x}_{i-1})]) \quad (10)$$

while $[\boldsymbol{\xi}, \boldsymbol{\zeta}]$ denotes the concatenation of vectors $\boldsymbol{\xi}$ and $\boldsymbol{\zeta}$.

- Then, the values of the latent variables (stochastic unit activations) $\boldsymbol{h}_i$ are drawn as samples from the inferred posterior density (6).

Proposed Approach: ReLaVaR

ReLaVaR Formulation

- We impose a suitable Multinoulli distribution over the generated output variables of our model, $\boldsymbol{y}_{i+1}$, conditional on the corresponding latent vectors, $\boldsymbol{h}_i$.

$$p(y_{i+1,j} = 1|\boldsymbol{h}_i) \propto \tau(\boldsymbol{w}_y^j \cdot \boldsymbol{h}_i) \tag{11}$$

  where $\boldsymbol{W}_y = [\boldsymbol{w}_y^j]_{j=1}^m$ are trainable parameters of the output layer of the model.

- This selection can be viewed as giving rise to a pointwise ranking criterion, with the negative conditional log-likelihood from (11), L_s, corresponding to the Categorical Cross-Entropy loss function.

Proposed Approach: ReLaVaR

ReLaVaR Training Algorithm

- Let us consider a training dataset $\mathcal{D}$, which comprises a number of click sequences (sessions), pertaining to a multitude of users.
- AVI for ReLaVaR consists in maximization of the evidence lower-bound (ELBO) expression:

$$\log p(\mathcal{D}) \geq \sum_i \Big\{ -\mathrm{KL}\big[q(\boldsymbol{h}_i;\boldsymbol{\theta})||p(\boldsymbol{h}_i)\big] - \mathbb{E}[L_s] \Big\} \tag{12}$$

- Here, $\mathrm{KL}\big[q||p\big]$ is the KL divergence between $q(\cdot)$ and $p(\cdot)$.
- Unfortunately, the posterior expectation $\mathbb{E}[L_s]$ cannot be computed analytically; hence, its gradient becomes intractable.
- Approximating this expectation via Monte-Carlo (MC) samples would result in estimators with unacceptably high variance.

Proposed Approach: ReLaVaR

ReLaVaR Training Algorithm

- We ameliorate this problem by means of a smart reparameterization of the MC samples drawn from the Gaussian posterior density:

$$\boldsymbol{h}^{\gamma} = \boldsymbol{\mu}_{\boldsymbol{\theta}}(\cdot) + \sigma_{\boldsymbol{\theta}}(\cdot)\,\boldsymbol{\epsilon}^{\gamma} \tag{13}$$

  where $\boldsymbol{\epsilon}^{\gamma}$ is white random noise with unitary variance, i.e. $\boldsymbol{\epsilon}^{\gamma} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I})$.

- By adopting this reparameterization, the drawn posterior MC samples are expressed as differentiable functions of the sought parameter sets, $\boldsymbol{\theta}$, and some random noise variable, *with low variance.*
- Then, we employ the gradient of the reparameterized ELBO [via (13)], in the context of *Adagrad*, to train the model.

Proposed Approach: ReLaVaR

Prediction Generation

- Recommendation generation in the context of a user session can be performed by computing the predicted ratings $\boldsymbol{y}_{i+1}$, and selecting the top-k of them to recommend to the user.
- To this end, we sample the latent variables $\boldsymbol{h}_i$ from the corresponding variational posterior distributions, under a standard MC-type rationale.

Experiments

- We exploit the RecSys Challenge 2015 benchmark.
- This comprises click-stream data pertaining to user sessions with an e-commerce website.
- We split the available data into one set comprising 7,966,257 sessions (with a total of 31,637,239 click actions), and another one comprising the remainder 5,324 sessions (with a total of 71, 222 click actions); we use the former for training and the latter for testing.
- Both sets entail a total of 37,483 items that a user may select to click on. Thus, we are dealing with a very sparse dataset.
- Our source codes have been developed in Python/Theano.

Experiments

ReLaVaR Model Configuration

- We perform Glorot Normal initialization of the model parameters.
- For regularization purposes, we apply Dropout on the component GRUs, with a rate equal to 0.5.
- To facilitate convergence, Nesterov momentum is additionally applied during training.
- We do not perform any tedious data augmentation procedure or model pretraining.

Experiments

Evaluation Metrics

- Recall@20 metric expresses the frequency at which the desired item in the test data appears in the 20 highest ranked items suggested by the evaluated approach.
- MRR@20 describes the average predicted score of the desired items in the test data, with the score values set to zero if the desired item does not make it to the top-20 list of ranked items.

Experiments

Table: Best performance results of the evaluated methods.

Method	Model Size	Recall@20	MRR@20
BPR-MF	-	0.2574	0.0618
GRU w/ BPR Loss	1000	0.6322	0.2467
GRU w/ TOP1 Loss	1000	0.6206	0.2693
M2	100	0.7129	0.3091
M4	1000	0.6676	0.2847
ReLaVaR	1500	0.6507	0.3527

Experiments

Table: ReLaVaR model performance for different selections of the employed loss function, L_s (results correspond to best network configuration).

Loss Function	TOP1	Cross-Entropy
# Latent Units	1000	1500
Step Size	0.1	0.05
Momentum Weight	0	0
Recall@20	0.6250	0.6507
MRR@20	0.2727	0.3527

Experiments

Table: Comparison of computational times (in seconds), for various selections of the employed loss function. Results pertain to network configurations yielding best accuracy.

Method	Network Size	Training time (total)	Prediction time per click event (average)
GRU w/ BPR Loss	1000 Units	48692.48	0.683
GRU w/ TOP1 Loss	1000 Units	44716.60	0.627
ReLaVaR w/ TOP1 Loss	1000 Units	42357.84	0.595
ReLaVaR w/ Cross-Entropy Loss	1500 Units	60109.86	0.844

Experiments

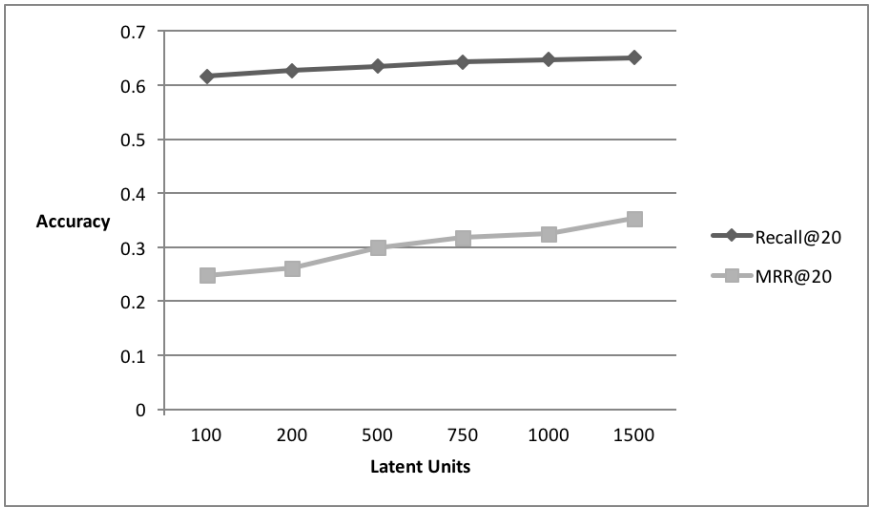

Figure: ReLaVaR performance fluctuation with the number of latent variables.

Experiments

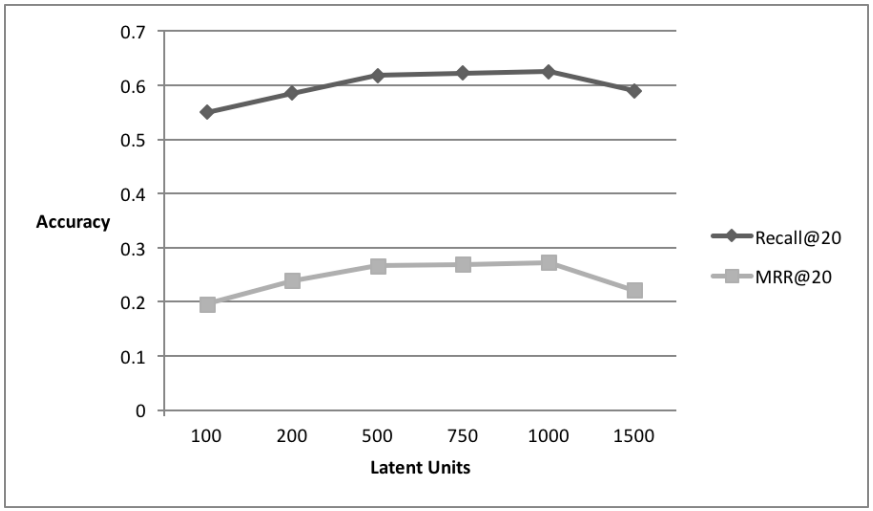

Figure: ReLaVaR performance fluctuation with the number of latent variables: Use of the TOP1 loss function.

Questions...?

DOSSIER-Cloud
DevOpS-based Software
engIneERing for the cloud

The project leading to this research has received funding from the European Unions Horizon 2020 research and innovation programme under grant agreement No 692251.