Amortized inference through normalized nonnegative models

Cyril J. Stark
Center for Theoretical Physics
Massachusetts Institute of Technology
Cambridge, MA 02139
cyril@mit.edu

Abstract

At the example of movie recommendation, we demonstrate the use of normalized nonnegative models (NNM) for top-down amortized inference. More precisely, we extract hierarchies of questions from NNMs. These hierarchies of questions have the following property. The answer of a user to the first top-level question locates the representation of the user within one chamber C_1 of a very coarse-grained partitioning of an underlying effective sample space. The second question depends on C_1. The answer of the user to the second question locates her representation in a smaller chamber C_2 of a partitioning of C_1. Thus, answers to short lists of *interpretable* questions allow for the approximate inference of user representations. This type of inference is amortized because we only compute an appropriate NNM once, and use it subsequently in terms of *fast* top-down inference.

1 Introduction

In this paper we demonstrate the use of *normalized nonnegative models* [1] (NNM) for top-down amortized inference [2, 3, 4]. We explain the value of this class of models in the context of movie recommendation [5]. However, we hope that our explanations will be clear enough to convince the reader that NNM-based top-down inference can be applied far beyond movie recommendation.

Before we go into the details about NNMs we need to introduce the *system-state-measurement* paradigm which makes a clear distinction between the 'state of a system' and the 'measurement method' used to probe that system. This paradgim is used almost everywhere in science and engineering. The paper [1] proposes to adopt that paradigm in the context of item recommendation. In item recommendation, the *system* is that abstract part of our mind that evaluates whether we like or dislike an item. Each person has individual preferences and therefore, each person's system in manifested differently. In recommendation, we use *states* to capture those individual manifestations: we assign to each person a state that specifies that person's system (i.e., that person's individual preferences). The *measurements* that we perform on the system are questions like "Do you like the movie *Despicable Me*?". Each measurement/question probes a person's state/movie taste. By performing multiple measurements of a person's taste, we get an idea of that person's state/preferences.

In the natural sciences and engineering, the system-state-measurement paradigm is often adopted by modeling systems in terms of a *sample space*, by modeling states in terms of *probability distributions* on that sample space, and by modeling measurements in terms of *random variables*. Hence, to describe data in this manner, we need to determine the following building blocks. Firstly, we need to compute an effective sample space. Secondly, we need to compute a probability distribution for each person to describe that person's preferences. Thirdly, we need to compute a random variable[1] for

[1] I.e., a function mapping the sample space to some alphabet.

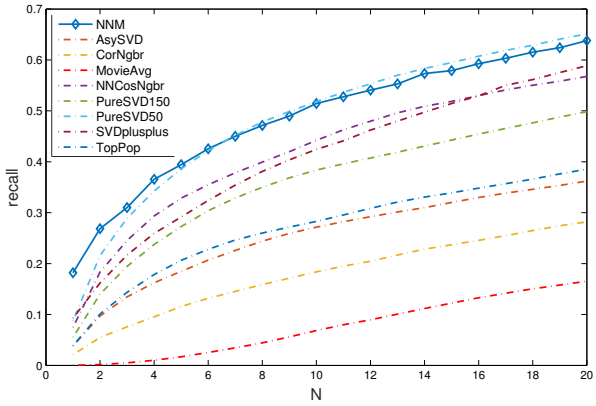

Figure 1: Recall at N [6] for MovieLens 1M; see [7] for description of other methods.

each item i to describe questions like "Do you like item i?". In the definition of NNMs we simply convexly relax the third of those building blocks so that the search for NNMs can be addressed conveniently in terms alternating optimization.

The main strength of NNMs are highly interpretable user and item representations [1]. These distinguished representations allow us to use NNMs for top-down amortized inference. Once we have computed an NNM, we can find approximate representations of *new* users (that have not yet rated any movie) by asking them a *short* list of *intuitive* questions. The answer the user provides to the first question locates the support of the user's state in a chamber C_1 of a very coarse grained partitioning of the underlying effective sample space. The second question locates the support of the user's state in a chamber C_2 of a partitioning of C_1. The third question locates the user's state in a chamber C_3 of a partitioning of C_2, etc. Hence, said short list of question realizes a method for top-down inference. This method realizes amortized inference because it uses the previously computed NNM. We demonstrate the practicability of top-down inference with NNMs by computing those lists of questions for the MovieLens 1M dataset. The use of NNMs (and the resulting top-down inference) is justified through their *high predictive power* (see figure 1; MAE on MovieLens 1M is 0.643).

Of course, topic models like those based on latent Dirichlet allocation [8] also allow for the extraction of hierarchical orderings of items. Those hierarchical orderings are other natural candidates for top-down inference. We believe, however, that our approach is more elementary in that, for instance, we do not need to assume generative processes like the nested Chinese restaurant process [9, 10] or nested hierarchical Dirichlet processes [11].

2 Probability theory recap

According to Kolmogorov, a random experiment with finite sample space is described by the following triple: Firstly, a sample space $\Omega = \{\omega_1, ..., \omega_D\}$. The elements ω_j denote elementary events. Secondly, a probability distribution $\vec{p} \in \mathbb{R}_+^D$ with $\sum_j (\vec{p})_j = 1$, i.e., $\vec{p}$ is an element of the probability simplex Δ. Thirdly, a random variable $\hat{E}$, i.e., a function $\hat{E} : \Omega \to \{1, ..., Z\}$ for some alphabet size $Z \in \mathbb{N}$. We denote by $\mathbb{P}[\hat{E} = z]$ the probability of the event $\{\omega \in \Omega | \hat{E}(\omega) = z\}$. Therefore, $\mathbb{P}[\hat{E} = z] = \mathbb{P}[\hat{E}^{-1}(z)] = \sum_{\omega \in \hat{E}^{-1}(z)} p_\omega$ where $\hat{E}^{-1}(z) \subseteq \Omega$ denotes the pre-image of z under the map $\hat{E}$. This probability can be expressed in terms of indicator vectors. For that purpose we define $\vec{E}_z$ by $\left(\vec{E}_z\right)_j = 1$ if $\omega_j \in \hat{E}^{-1}(z)$, and $\left(\vec{E}_z\right)_j = 0$ otherwise. It follows that $\mathbb{P}[\hat{E} = z] = \vec{E}_z^T \vec{p}$.

3 Normalized nonnegative models

We adopt the system-state-measurement paradigm. The *system* we are interested in is that part of our mind that determines the outcome to the question "Do you like item i?" ($i \in [I]$). For each

user $u \in [U]$ this system is in some state described by a distribution $\vec{p}_u \in \Delta$ on some unknown sample space $\Omega = \{\omega_1, ..., \omega_D\}$ representing the system. Each question "Do you like item i?" is modeled in terms of a random variable $\hat{E}_i$ with alphabet $[Z]$ ($Z = 5$ for 5-star ratings). We denote by $\mathbb{P}_u[\hat{E}_i = z]$ the probability for user u to rate item i with $z \in [Z]$. Thus,

$$\mathbb{P}_u[\hat{E}_i = z] = \mathbb{P}_u[\hat{E}_i^{-1}(z)] = \vec{E}_{iz}^T \vec{p}_u \tag{1}$$

where $\vec{p}_u$ models the state of user u and where $\vec{E}_{iz} \in \{0,1\}^D$ is defined by

$$\left(\vec{E}_{iz}\right)_j = \begin{cases} 1, & \text{if } \omega_j \in \hat{E}_i^{-1}(z) \\ 0, & \text{otherwise.} \end{cases} \tag{2}$$

By (2), $\sum_z \vec{E}_{iz} = (1, ..., 1)^T$. We denote by $\mathcal{M}_0$ the set of all valid descriptions $(\vec{E}_1, ..., \vec{E}_Z)$ of a random variable $\hat{E}$, i.e., $\mathcal{M}_0 = \left\{(\vec{E}_1, ..., \vec{E}_Z) \in \{0,1\}^{D \times Z} \middle| \sum_z \vec{E}_{iz} = (1, ..., 1)^T \right\}$. Allowing for stochastic mixtures of elements of $\mathcal{M}_0$ we arrive at the *convex relaxation*

$$\mathcal{M} = \left\{(\vec{E}_1, ..., \vec{E}_Z) \in \mathbb{R}_+^{D \times Z} \middle| \sum_z \vec{E}_{iz} = (1, ..., 1)^T \right\}$$

of $\mathcal{M}_0$. In the remainder, the tuple of vectors $\left((\vec{p}_u)_u, (\vec{E}_{iz})_{iz}\right)$ denotes a *normalized nonnegative model* (NNM) if $\vec{p}_u \in \Delta$ for all users $u \in [U]$ and if $(\vec{E}_{iz})_z \in \mathcal{M}$ for all items $i \in [I]$.

4 Hierarchical structure of tags

In conventional movie recommendation, we only have access to some ratings users provide for movies. Oftentimes, movies (or more general observations respectively policies) come with side information. For instance, the famous MovieLens 1M dataset specifies the genres (like *Action, Adventure, Animation,* etc) of movies. Or on webpages like StackExchange, users tag questions with tags like *c++*, *sql-server*, etc. We propose modeling properties like genre tags through the following game which is motivated by the question whether a particular user likes a randomly drawn movie from a particular genre. The game involves a referee and a player named Alice. It proceeds as follows.

1. The referee chooses at random a genre g and a user u.
2. Alice is given access to $\vec{p}_u$, to g, to all genre-labels $(g_1^i, ..., g_{n_i}^i)_{i \in [I]}$ of all movies and to all movie vectors $(\vec{E}_{iz})_{iz}$ ($Z = 2$; $z = 1$ means 'dislike', $z = 2$ means 'like').
3. The referee chooses uniformly at random a movie i^* from the set $\{i_1^g, ..., i_{m_g}^g\}$ of all movies that were tagged with g.
4. By looking up the user-item matrix, the referee checks whether user u likes or dislikes i^*. We denote u's opinion by $z^* \in \{\text{dislike,like}\}$.
5. Alice guesses z^*. She wins the game if she guesses correctly and loses otherwise.

What is Alice's winning probability? Conditioned on the event *Referee draws* i, $\mathbb{P}[z^* = 2|i] = \vec{p}_u^T \vec{E}_{i2}$. The probability for the event *Referee draws* i is $1/m_g$ because there are m_g movies tagged with g. It follows that $\mathbb{P}[z^* = 2] = \frac{1}{m_g} \sum_{i \in \{i_1^t, ..., i_{m_g}^t\}} \mathbb{P}[z^* = 2|i] = \vec{p}_u^T \vec{E}_g$ with

$$\vec{E}_g := \frac{1}{m_g} \sum_{i \in \{i_1^g, ..., i_{m_g}^g\}} \vec{E}_{i2}. \tag{3}$$

We think that $\vec{E}_g$ serves as useful characterization of the genre g because $\vec{E}_g$ determines for each user u the probability that u likes a random movie with property g.

As we are going to see next, (3) enables us to order tags in a hierarchical manner [1]. For that purpose, we first observe that if the genre vectors $\vec{E}_g$ were binary vectors (i.e., $\in \{0,1\}^D$), then we would say $g \subseteq g'$ whenever the support of $\vec{E}_g$ is contained in the support of $\vec{E}_{g'}$. This is a meaningful definition of '$\subseteq$' because if $g \subseteq g'$ then $\vec{p}_u$ *likes* g implies $\vec{\delta}_\omega$ *likes* g'.

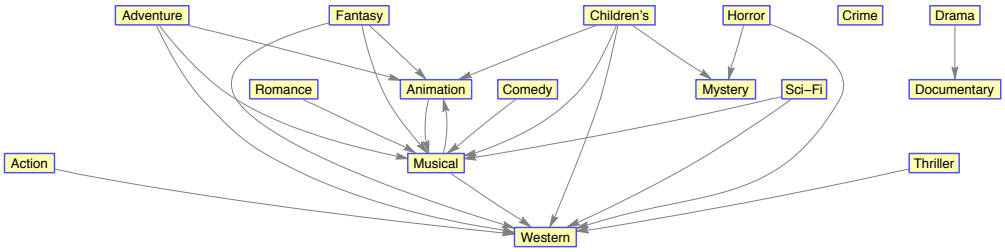

Figure 2: Hierarchy graph G extracted from MovieLens 1M dataset. Here, $\varepsilon = 1/3$ and $D = 8$. $D = 8$ was selected by cross validation: $D = 8$ results in best mean average error in 5-fold cross-validation.

Generally, in NNMs, genre vectors $\vec{E}_g$ are not exactly binary and therefore, the definition of $g \subseteq g'$ needs to make sense for non-binary vectors. To find a new definition of '$\subseteq$' we note that in the previous binary setting, $g \subseteq g'$ if and only if $\vec{E}_{g'}^T \vec{E}_g = \sum_{\omega \in \Omega} (\vec{E}_g)_\omega =: \|\vec{E}_g\|_1$. This last condition may not be satisfied if the components of $\vec{E}_{g'}$ are < 1. This can happen in NNMs. However, the operational meaning of $\vec{E}_{g'}^T \vec{E}_g = \|\vec{E}_g\|_1$ is preserved under the relaxation

$$\vec{E}_{g'}^T \vec{E}_g \geq (1 - \varepsilon) \|\vec{E}_g\|_1 \tag{4}$$

if $\varepsilon > 0$ is small. That is because if the relaxed condition (4) is satisfied then we still have that most of the weight of $\vec{E}_g$ is contained in the approximate support of $\vec{E}_{g'}$. Therefore, we say $g \subseteq_\varepsilon g'$ if condition (4) is satisfied.

The collection of all relations '$\subseteq_\varepsilon$' between genres can be represented graphically in terms of a graph. For that purpose we interpret the bag of all genres $\{g\}_{g \in [G]}$ as the vertex set V of a directed graph $G = (V, E)$. We define the edge set E of G in terms of the following rule: $(g \to g')$ if $g' \subseteq_\varepsilon g$. For every choice of $\varepsilon \in [0, 1]$, the graph G induces an approximate hierarchical ordering of genres; see figure 2 for an example.

5 Amortized top-down inference

Assume user u has not rated any movie. How can we estimate $\vec{p}_u$? This is the so called *cold-start problem*. Here we propose to make use of hierarchy graphs (see for example figure 2) to very quickly get non-trivial estimates for $\vec{p}_u$. We assume that $\vec{p}_u$ is approximately sparse because otherwise, it is anyways impossible to make meaningful predictions (e.g., when $\vec{p}_u$ is the uniform distribution). Thus, we need to determine the small support of $\vec{p}_u$. We proceed by traveling from top to bottom in the hierarchy graph G. This is best explained in terms of an example. Consider the hierarchy graph in figure 2, and imagine the following interaction between u and 'RecSys' (short for 'recommender system'):

1. *RecSys:* Which of the genres *Action, Adventure, Romance, Fantasy, Comedy, Children's, Horror, Sci-Fi, Crime, Drama, Thriller* do you like most?
2. *User: Children's.*
3. *RecSys:* Which of the genres *Animation, Musical, Mystery* do you like most?
4. *User: Musical.*
5. *RecSys:* Do you like *Westerns*?
6. *User:* No.

Based on this interaction we set our initial estimate for $\vec{p}_u$ equal to the uniform distribution on the approximate support of $\vec{E}_{\text{Musical}}$ minus the approximate support of $\vec{E}_{\text{Western}}$. We conclude that 3 simple questions ($\approx$ depth of G) suffice to determine a nontrivial estimate for $\vec{p}_u$. This type of inference is *amortized* because we make use of the computed NNM and it is *top-down* because we travel from (coarse grained) genres with large support to (fine grained) genres with small support.

References

[1] Cyril J Stark. Normalized nonnegative models for expressive recommender systems. *accepted to AAAI conference 2016; preprint arXiv:1511.04775*, 2015.

[2] Andreas Stuhlmüller, Jacob Taylor, and Noah Goodman. Learning stochastic inverses. In *Advances in neural information processing systems*, pages 3048–3056, 2013.

[3] Samuel J Gershman and Noah D Goodman. Amortized inference in probabilistic reasoning. In *Proceedings of the 36th Annual Conference of the Cognitive Science Society*, 2014.

[4] Andreas Stuhlmüller. *Modeling Cognition with Probabilistic Programs: Representations and Algorithms*. PhD thesis, Stanford University, 2015.

[5] Yehuda Koren, Robert Bell, and Chris Volinsky. Matrix factorization techniques for recommender systems. *Computer*, (8):30–37, 2009.

[6] Cyril J Stark. Top-n recommendations from expressive recommender systems. *preprint arXiv:1511.04775*, 2015.

[7] Paolo Cremonesi, Yehuda Koren, and Roberto Turrin. Performance of recommender algorithms on top-N recommendation tasks. In *Proceedings of the fourth ACM conference on Recommender systems*, pages 39–46. ACM, 2010.

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

[9] David M Blei, Thomas L Griffiths, Michael I Jordan, and Joshua B Tenenbaum. Hierarchical topic models and the nested chinese restaurant process. *Advances in neural information processing systems*, 16:17, 2004.

[10] David M Blei, Thomas L Griffiths, and Michael I Jordan. The nested chinese restaurant process and bayesian nonparametric inference of topic hierarchies. *Journal of the ACM (JACM)*, 57(2):7, 2010.

[11] John Paisley, Chingyue Wang, David M Blei, Michael Jordan, et al. Nested hierarchical dirichlet processes. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 37(2):256–270, 2015.