Skill-based Career Path Modeling and Recommendation

Aritra Ghosh, Beverly Woolf, Shlomo Zilberstein, Andrew Lan
College of Information and Computer Sciences, University of Massachusetts Amherst
{arighosh, bev, shlomo, andrewlan}@cs.umass.edu

***Abstract*—The development of new technologies at an unprecedented rate is rapidly changing the landscape of the labor market. Therefore, for workers who want to build a successful career, acquiring new skills required by new jobs through lifelong learning is crucial. In this paper, we propose a novel and interpretable monotonic nonlinear state-space model to analyze online user professional profiles and provide actionable feedback and recommendations to users on how they can reach their career goals. Specifically, we use a series of binary-valued and non-decreasing latent states to represent the expanding skill set of each user throughout their career and propose an efficient inference method under our model. Using a series of experiments on two large real-world datasets, we show that our model (sometimes significantly) outperforms existing methods on the tasks of company, job title, and skill prediction. More importantly, our model is interpretable and can be used for other important tasks including skill gap identification and career path planning. Using a series of case studies, we show that our model can provide i) actionable feedback to users and guide them through their upskilling and reskilling processes and ii) recommendations of feasible paths for users to reach their career goals.**

I. Introduction

New skills and knowledge are needed for jobs in the future due in part to the rapid development of workplace technology such as artificial intelligence and internet of things. Jobs in the future will likely require skills that are not taught in schools nor in standard training programs. Instead, workers will have to either upskill as they move to new jobs within the same industry, or reskill themselves through the lifelong learning process to move to another industry. In a survey conducted by the Pew Research Center in 2016, 87% of the participants realize the importance of retraining and reskilling [1]. Therefore, studying how users acquire skills in their lifelong learning process and how those skills affect their future career is of crucial importance to the world economy. Fortunately, the big data revolution has created an opportunity for researchers to collect and analyze large-scale data to understand the evolving labor market landscape and the upskilling and reskilling processes. Examples of such data include job postings, e.g., those collected by Burning Glass [2], connections between users, companies, and skills in economic graphs [3], and user profiles/resumes on online professional networking sites such as LinkedIn [4] and CareerBuilder [5]. These datasets enable researchers to develop tools to help individual users to navigate possible future career paths and guide them through upskilling and reskilling to reach their career goal.

There are mainly two types of existing works on career path analysis. First, there are works at the *macroscopic* level that use graph embedding methods to analyze co-occurrence graphs of companies, jobs, skills, or a combination thereof to learn representations of these entities. In [6], the authors developed the Job2Vec method to learn the relationship between jobs and companies using graph embeddings and showed that these embeddings are effective at link prediction. In [5], the authors developed a representation learning method to analyze transitions between jobs and skill co-occurrences and showed that these representations are effective at the next job and skill prediction. These methods mostly only analyze one career "hop", i.e., the jump from the previous job to the next job and do not take each user's entire professional history into account. Therefore, these methods are not personalized and cannot help users explore long-term career paths.

Second, there are works on the *microscopic* level that analyze the sequence of career experiences in each user's professional profile. In [4], the authors developed a contextual long short-term memory (LSTM) model, NEMO, to predict a user's next job using all the previous experiences and skills listed in their LinkedIn profiles. In some earlier works [7], [8], the authors developed survival analysis-based methods to predict how long a user will work on a particular job [9]. These methods mostly resort to recurrent neural networks (RNNs) and their variants to analyze sequences of career experiences and showed good performance in the next job prediction. However, due to the uninterpretable nature of these neural network-based methods, they cannot be used to provide meaningful feedback to users on how they should upskill or reskill in order to reach their career goals.

Given the limitations of existing works, there is a need to develop methods that can not only accurately predict each user's future jobs but also provide *actionable* feedback and recommendations. These methods should satisfy two requirements. First, they should be interpretable so that users can understand the impact of each job on their skill set and skills they need to acquire so that they can qualify for certain jobs. Second, they should take user preferences, e.g., their career goals and real-life constraints into account in their recommendations to fully adapt to the needs of each user.

A. Contributions

In this paper, we propose a novel and interpretable model, the monotonic nonlinear state-space (`MNSS`) model, to analyze

online user professional profiles and provide i) actionable feedback to users on skills they need to acquire and ii) recommendations on their future career path. Our model is motivated by the observation that working on a job is not only proof that the user has the skills required for the job, but also a valuable opportunity for a user to *acquire new skills*. Our specific contributions are as follows:

1) We use a series of stochastic, *binary-valued* latent states to characterize whether or not a user masters a skill at each point in their career. We also restrict them to be *non-decreasing* over time to capture users' expanding skill sets during their careers. We show that `MNSS` (sometimes significantly) outperforms baseline methods on the tasks of company, job title, and skill prediction, using two large datasets collected from LinkedIn and Indeed.
2) We formulate the task of skill gap identification as an optimization problem that can be solved to find a *small* set of skills a user needs to improve the most to achieve their desired career goal. We use several case studies to demonstrate that the identified skill gap can be used to provide actionable feedback to users on their upskilling and reskilling processes.
3) We formulate the task of career path recommendation as a path planning problem that can be (approximately) solved to find feasible paths a user can follow towards their ultimate career goal. We use several case studies to demonstrate that the identified (approximately) optimal career paths can be used to provide practical career recommendations to users.

We also acknowledge a limitation of our work in that the data we analyze does not contain counterfactual information, i.e., job offers that users turned down or jobs they did not qualify for. Therefore, we can only analyze the observed career decisions made by each user and can neither take real-life constraints they faced into account nor study user qualifications. Moreover, our career path recommendations are generated from historical user career path data and may be biased. Therefore, we emphasize that our recommendations can augment, but not replace, the decision-making process of a user throughout their career. Throughout the paper, we use only professional/educational experiences of users for forecasting; we do not use any user (such as demographic) information for any of the tasks. [1]

B. Related Work

The latent state-space model is a generic framework for modeling sequential data. Recently, deterministic RNNs and their modern versions such as LSTMs and gated recurrent units (GRUs) have shown remarkable performance on most sequential data modeling tasks including speech modeling, language processing, and video understanding [10]. Many of the previous career modeling works use RNNs; for example, NEMO uses an LSTM with contextual inputs to predict a user's next career move [4].

[1]Our code is available at https://github.com/arghosh/MNSS.

A common alternative to RNNs is to use models with stochastic latent states such as hidden Markov models (HMMs) or linear state-space models (L-SSMs), which offer interpretable latent state transition and observed state emission functions [11]. Moreover, stochastic models such as HMM/L-SSM are more appropriate for capturing randomness in the dataset than deterministic models such as RNNs [12]. However, HMM and L-SSM have simple (discrete or linear) latent states and can not capture the nuance in large, noisy real-world datasets. Lately, many advances have been made on nonlinear stochastic state-space models using the variational principle [12], [13]. However, exact inference is intractable in nonlinear state-space models. Following prior work, we learn the parameters in `MNSS` by maximizing the so-called evidence lower bound (ELBO) [14].

II. Problem Formulation

We denote the career profile of user i as $\mathcal{T}^i = \left([\mathcal{E}^i_1, \ldots, \mathcal{E}^i_{T_i}],\ \mathcal{S}^i\right)$, with T_i experiences in their career trajectory and $\mathcal{E}^i_t$ denotes the tth experience (either professional or educational). $\mathcal{S}^i = \{\mathbf{s}^i_1, \ldots, \mathbf{s}^i_{M_i}\}$ denotes the user's listed skill set and $\mathbf{s}^i_j$ denotes the jth observed skill, with a total of M_i observed skills. Each educational experience contains information on e.g., School, Degree, Major, Start Time, and Duration, whereas each professional experience contains information on e.g., Company, Job Title, Start Time, and Duration. There is no ordering among the skills listed in the observed skill set. Moreover, in most user profiles on major online professional network websites, there is no information on when the user acquired each skill in their career. Instead, the listed skill set is only a snapshot of their (evolving) true skill set captured at the time the profile is accessed. Therefore, we do not use these observed skills to predict a user's professional experiences like the work in [4].

Our goal is to develop a model for user career paths that can not only i) predict the career path of each user but also ii) provide actionable feedback and career recommendations to users to help them make important career decisions. In the remainder of the paper, we omit the superscript i for user i for simplicity of exposition when we discuss one user. We start by defining two prediction tasks that will help us model user career paths.

1) **Company and Job Title Prediction**: Predict $\mathbf{c}_t$, the company, and $\mathbf{o}_t$, the job title of a user's next professional experience, given their previous experiences, i.e., $[\mathcal{E}_1, \ldots, \mathcal{E}_{t-1}]$.
2) **Skill Set Prediction**: Predict a user's listed skill set $\mathcal{S} = \{\mathbf{s}_1, \ldots, \mathbf{s}_M\}$, given their entire career history, i.e., $[\mathcal{E}_1, \ldots, \mathcal{E}_T]$.

We can adopt many black-box models for these prediction tasks. However, due to their uninterpretable nature, these models cannot be used to provide actionable feedback and meaningful career recommendations to users. Therefore, we define three auxiliary tasks that an interpretable model should be able to tackle:

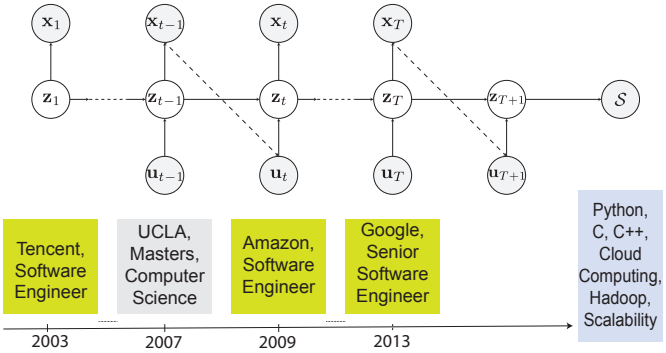

Fig. 1: The structure of the MNSS model and an example user profile with T experiences.

3) **Skill Acquisition Process Reconstruction**: Infer the (unobserved) skill sets, $\mathcal{S}_t \subseteq \mathcal{S}$, of each user after each past career experience. These predicted skill sets will help us to reconstruct the skill acquisition process throughout the user's career.
4) **Skill Gap Identification**: Identify the skill gap a user is facing, i.e., a list of skills that the user needs to acquire or improve on to reach their desired career goal (a company, job title pair).
5) **Career Path Recommendation**: Find feasible career paths that connect a user's current career state to their career goal. These career paths consist of professional experiences that are attainable and ultimately lead the user to their desired career goal.

We start by outlining a generic latent state-space model framework for tasks (1) and (2) defined above. In Section III, we propose the interpretable MNSS model that is capable of completing both these tasks and the auxiliary tasks (3), (4), and (5) defined above.

A. Latent State-space Models for Career Experience Prediction

As shown in Figure 1, we adopt a generic latent state-space model to analyze sequences of professional experiences in user career paths. The key component of this model is to use a set of latent states, $\mathbf{z}_t \in \mathbb{R}^D$, $t = 1, \ldots, T$, to characterize a set of *unobserved*, evolving variables that dictate the user's career experience at each time step. At each time step, there is an input to the latent state, denoted as $\mathbf{u}_t$, and an *observed* output from the latent state, denoted at $\mathbf{x}_t$. Under most real-world settings, the observed output from the last time step is used as the input to the next time step, i.e., $\mathbf{u}_t = \mathbf{x}_{t-1}$, except for the first time step where there is no input; we initialize $\mathbf{z}_1$ to an all-zero vector. [2] In our problem setting, the latent state variables $\mathbf{z}_t$ correspond to the *latent skill states* of each user at each point in time, and the observed output variables $\mathbf{x}_t$ correspond to the *observed career experiences* of the user at that time. Since our goal is to predict professional experiences, we omit educational experiences at the output, but they are still used as input into the next latent skill state. We do not predict the duration of professional experiences, although that can be modeled using point processes [15]. Instead, we use as part of the input states to learn how the duration of a professional experience impacts a user's skills.

This latent state-space model has two key components: a latent skill state *transition* model $p(\mathbf{z}_t|\ \mathbf{z}_{1:t-1}, \mathbf{u}_{1:t})$ and an observed career experience *emission* model $p(\mathbf{x}_t|\mathbf{z}_t)$. The transition model characterizes how each professional experience impacts the latent skill state of the user, i.e., *how does a user acquire new skills and improve existing skills in an on-job setting*. We note that there are other ways for user upskilling, such as going through training programs and taking online courses; however, since they are not often listed in online user professional profiles, we do not model them with our transition model. The emission model characterizes how latent skill states decide a user's next career experience. Therefore, we use the transition model to estimate a user's current latent skill states from their past experiences and then use the emission model to predict their next career experience. We also introduce an additional emission model $p(\mathcal{S}|\mathbf{z}_{T+1})$ to predict a user's listed skill set given their entire career path. Our goal is to learn these transition and emission models from real-world user career profiles.

III. Methodology

In this section, we detail the MNSS model for user career path modeling. Black-box neural network-based models are generally not interpretable, which means that they can excel at prediction while being unable to provide actionable feedback to users. For example, since the latent states in these models are usually not associated with a user's mastery level of any observed skills, they cannot be used to recommend a user which skills they should acquire in order to reach their career goal.

In order to enhance interpretability, we make a key assumption: *a user's skill mastery increases over time* as they have more career (either educational or professional) experiences. Therefore, we place a monotonic constraint on the latent skill states of each user. Moreover, typical state-space models use continuous latent states that are not easily interpretable. Therefore, we use *binary-valued* random variables as the latent skill states in MNSS that indicate whether a user masters a latent skill or not. In what follows, we first lay out the MNSS model and then detail a method for efficient approximate inference using a novel monotonic GRU module.

A. Monotonic Nonlinear State-space Model

Excluding the likelihood of the listed skill set (which we detail in Section III-B0c), the joint probability of all career experiences in a user's profile and their corresponding latent skill states is given by

$$p_\theta(\mathbf{x}_{1:T}, \mathbf{z}_{1:T}|\mathbf{u}_{1:T}) = p_{\theta_z}(\mathbf{z}_1) \prod_{t=2}^{T} p_{\theta_z}(\mathbf{z}_t|\mathbf{z}_{t-1}, \mathbf{u}_t) \prod_{t=1}^{T} p_{\theta_x}(\mathbf{x}_t|\mathbf{z}_t),$$

where $p_{\theta_z}(\cdot)$ is the latent skill state transition model and $p_{\theta_x}(\cdot)$ is the career experience emission model; θ_z and θ_x

[2] In our experiments, we found that setting $\mathbf{z}_1$ to a learnable parameter vector did not lead to improved prediction performance.

denote the set of parameters in these models, respectively, and $\theta = \{\theta_x, \theta_z\}$ denotes the set of all model parameters. We exclude the term $p_{\theta_z}(\mathbf{z}_1)$ from our analysis since it is the same for all users and does not impact our predictions. As mentioned above, we make two novel model choices: First, we use *binary-valued* discrete random variables as the latent skill states, i.e., $\mathbf{z}_t = [z_{t,1}, \ldots, z_{t,D}] \in \{0,1\}^D$, where D denotes the number of latent skills. In this setup, $z_{t,j'} = 1$ means that the user masters latent skill j' after the tth experience. Second, we constraint the latent skill states to be non-decreasing, i.e., $\mathbf{z}_{t-1} \leq \mathbf{z}_t$[3], where the inequality operates element-wise on vectors. Under these model choices, $z_{t,j'} = 1$ and $z_{t-1,j'} = 0$ means that the user *acquired* latent skill j' after the tth experience, i.e., by working a job at a company for some duration (in case of a professional experience).

B. Approximate Inference

Exact inference of the posterior distribution $p_\theta(\mathbf{z}_{1:T}|\mathbf{x}_{1:T}, \mathbf{u}_{1:T})$ in MNSS is computationally intractable. Therefore, following [12], [13], we use an auxiliary distribution $q_\phi(\mathbf{z}|\mathbf{x}, \mathbf{u})$ (referred to as an recognition network and usually parameterized by an RNN) with parameters ϕ, to approximate the true posterior. We perform approximate inference by maximizing the ELBO on the marginal observed data log-likelihood given by

$$\begin{aligned}\log p_\theta(\mathbf{x}_{1:T}|\mathbf{u}_{1:T}) &\geq \mathcal{L}(\theta, \phi) \\ &= \mathbb{E}_{q_\phi(\mathbf{z}_{1:T}|\mathbf{x}_{1:T}, \mathbf{u}_{1:T})} \log p_{\theta_x}(\mathbf{x}_{1:T}|\mathbf{z}_{1:T}) \\ &- \beta\ D_{\text{KL}}(q_\phi(\mathbf{z}_{1:T}|\mathbf{x}_{1:T}, \mathbf{u}_{1:T}) \,||\, p_{\theta_z}(\mathbf{z}_{1:T}|\mathbf{u}_{1:T})),\end{aligned}$$

with respect to the model parameters θ and the recognition network parameters ϕ [14]. Here, $q_\phi(\mathbf{z}_{1:T}|\mathbf{x}_{1:T}, \mathbf{u}_{1:T})$ is the approximate posterior distribution of the latent states and $p_{\theta_z}(\mathbf{z}_{1:T}|\mathbf{u}_{1:T})$ is its prior distribution. The first term is the reconstruction loss of the observed data sampled from the approximate posterior distribution, whereas the second term is the KL divergence between the approximate posterior distribution and the prior distribution. The parameter $\beta > 0$ controls the balance between the reconstruction loss and the KL divergence loss. Using the d-separation criterion in Figure 1, we can decompose the prior distribution as

$$p_{\theta_z}(\mathbf{z}_{1:T}|\mathbf{u}_{1:T}) = p_{\theta_z}(\mathbf{z}_1) \textstyle\prod_{t=2}^{T} p_{\theta_z}(\mathbf{z}_t|\mathbf{z}_{t-1}, \mathbf{u}_t),$$

and the approximate posterior distribution as

$$q_\phi(\mathbf{z}_{1:T}|\mathbf{x}_{1:T}, \mathbf{u}_{1:T}) = q_\phi(\mathbf{z}_1) \textstyle\prod_{t=2}^{T} q_\phi(\mathbf{z}_t|\mathbf{z}_{t-1}, \mathbf{x}_t, \mathbf{u}_t).$$

To approximate $q_\phi(\mathbf{z}_t|\mathbf{z}_{t-1}, \mathbf{x}_t, \mathbf{u}_t)$, we need to sample $\mathbf{z}_{t-1}$ first. One option is to use ancestral sampling to approximate q_ϕ in a way that is similar to the structured inference method for nonlinear state-space models [12], [13]. However, ancestral sampling is slow due to its sequential nature and has high variance in its estimates [16]. Therefore, we employ another factorization of the approximate posterior distribution where the distribution of $\mathbf{z}_t$ does not condition on $\mathbf{z}_{t-1}$, to avoid sampling. We observe that the variable $\mathbf{z}_t$ depends on $\mathbf{u}_{1:t}$ and $\mathbf{x}_t$ when $\mathbf{z}_{t-1}$ is unobserved. Using recurrence, we can equivalently write $q_\phi(\mathbf{z}_t|\mathbf{z}_{t-1}, \mathbf{x}_t, \mathbf{u}_t) = q_\phi(\mathbf{z}_t|\mathbf{x}_t, \mathbf{u}_{1:t})$. Since $\mathbf{x}_t = \mathbf{u}_{t+1}$, we have $q_\phi(\mathbf{z}_t|\mathbf{x}_t, \mathbf{u}_{1:t}) = q_\phi(\mathbf{z}_t|\mathbf{u}_{1:t+1})$. Similarly, we can write the prior as $p_{\theta_z}(\mathbf{z}_t|\mathbf{z}_{t-1}, \mathbf{u}_t) = p_{\theta_z}(\mathbf{z}_t|\mathbf{u}_{1:t})$. The final ELBO becomes:

$$\begin{aligned}\mathcal{L}(\theta, \phi) = &\textstyle\sum_{t=1}^{T} \mathbb{E}_{q_\phi(\mathbf{z}_t|\mathbf{u}_{1:t+1})} \log p_{\theta_x}(\mathbf{x}_t|\mathbf{z}_t) \\ &- \beta \textstyle\sum_{t=1}^{T} D_{\text{KL}}(q_\phi(\mathbf{z}_t|\mathbf{u}_{1:t+1}) \,||\, p_{\theta_z}(\mathbf{z}_t|\mathbf{u}_{1:t})). \quad (1)\end{aligned}$$

Thus, given the input states up until time step t and the current output state $\mathbf{x}_t$, the recognition network $q_\phi(\cdot)$ predicts the distribution of the latent state $\mathbf{z}_t$. On the other hand, given the input states up until the current time step, $\mathbf{u}_{1:t}$, the prior network p_θ predicts the distribution of current latent state $\mathbf{z}_t$. The KL divergence term regularizes the recognition network so that it is close to the prior distribution [14]. If we replace the approximate posterior $q_\phi(\mathbf{z}_t|\mathbf{u}_{1:t+1})$ with the prior, $p_\theta(\mathbf{z}_t|\mathbf{u}_{1:t})$ in Eq. 1, the ELBO reduces to the log-likelihood, which is the objective in maximum likelihood estimation (MLE):

$$\mathcal{L}_{MLE}(\theta) = \textstyle\sum_{t=0}^{T} \mathbb{E}_{p_{\theta_z}(\mathbf{z}_t|\mathbf{u}_{1:t})} \log p_{\theta_x}(\mathbf{x}_t|\mathbf{z}_t).$$

In our experiments, we found that adding this objective to the ELBO improves training stability [10]. Thus, the final objective that we maximize for the career experiences of all users is

$$\textstyle\sum_i \mathcal{L}^i(\theta, \phi) + \alpha \mathcal{L}^i_{MLE}(\theta),$$

where $\alpha > 0$ is a tunable parameter. We learn the recognition network parameters ϕ and the generative model parameters θ simultaneously using stochastic gradient descent [14]. After the model parameters are trained, we use the learned prior model $p_{\theta_z}(\mathbf{z}_t|\mathbf{u}_{1:t})$ to compute the distribution of the latent skill state $\mathbf{z}_t$, which is used to predict the user's next career experience, $\mathbf{x}_t$.

a) The monotonic gated recurrent unit: We now detail our choice for the recognition and prior networks for the approximated posterior distribution and the prior distribution. Since the latent state $\mathbf{z}_t$ is a binary-valued vector, we characterize it as Bernoulli random variables with success probabilities $p_{\theta_z}(\mathbf{z}_t|\mathbf{u}_{1:t}) = \boldsymbol{\gamma}_t \in [0,1]^D$ for the prior model. We use an RNN-type model, g_{θ_z}, to characterize the latent skill state transitions as $\boldsymbol{\gamma}_t = g_{\theta_z}(\boldsymbol{\gamma}_{t-1}, \mathbf{u}_t)$. Similarly, for the approximate posterior model, we have $q_\phi(\mathbf{z}_t|\mathbf{u}_{1:t+1}) = \boldsymbol{\kappa}_t \in [0,1]^D$ and another RNN-type model, g_ϕ, where $\boldsymbol{\kappa}_t = g_\phi(\boldsymbol{\kappa}_{t-1}, \mathbf{x}_t)$.

Common RNN variants such as LSTMs and GRUs do not have monotonic latent states. Therefore, we propose a new variant of GRU, which we dub the monotonic GRU (MGRU), to model the latent skill state transitions. In the context of the prior, the MGRU is defined as (the approximate posterior follows the same structure with a different set of parameters)

$$\boldsymbol{\gamma}_t = \boldsymbol{\gamma}_{t-1} + (\mathbf{1} - \boldsymbol{\gamma}_{t-1}) \odot \mathbf{o}_t, \quad (2)$$

[3]In terms of the latent skill state transition model, the monotonic constraint can be stated as $p_{\theta_z}(\mathbf{z}_t|\mathbf{z}_{t-1}, \mathbf{u}_t) = 0$ if $\exists j'$ such that $z_{t-1,j'} > z_{t,j'}$.

where $\odot$ denotes element-wise multiplication between vectors and

$$\begin{aligned}\mathbf{k}_t &= \sigma(\mathbf{W}_k\mathbf{u}_t + \mathbf{U}_k\boldsymbol{\gamma}_{t-1}),\\ \mathbf{r}_t &= \sigma(\mathbf{W}_r\mathbf{u}_t + \mathbf{U}_r\boldsymbol{\gamma}_{t-1}),\\ \mathbf{o}_t &= \sigma(\mathbf{W}_o\mathbf{u}_t + \mathbf{U}_o(\mathbf{r}_t \odot \boldsymbol{\gamma}_{t-1})) \odot \mathbf{k}_t,\end{aligned} \tag{3}$$

where $\sigma(x) = 1/(1+\exp(-x))$ is the sigmoid function and $\mathbf{W}, \mathbf{U}$ denote parameter matrices. The $\boldsymbol{\gamma}_{t-1}$ term in Eq. 2 corresponds to the expected value of the latent skill states after time step $t-1$; since $\boldsymbol{\gamma}_{t-1} \in [0,1]^D$, we can interpret $(\mathbf{1} - \boldsymbol{\gamma}_{t-1}) \in [0,1]^D$ as the *skill deficiency* of the user (relative to full mastery of all latent skills, i.e., $\boldsymbol{\gamma} = \mathbf{1}$). $\mathbf{o}_t \in [0,1]^D$ corresponds to the portion of the skill deficiency gap being filled by the professional experience at time step t. Therefore, it is easy to see that $\boldsymbol{\gamma}_t \geq \boldsymbol{\gamma}_{t-1}$. We use a gating structure similar to that in GRUs [10] for the recurrence in Eq. 3; $\mathbf{k}_t, \mathbf{r}_t, \mathbf{o}_t$ correspond to the update gate, reset gate and update value components of GRUs, respectively. We note that in addition to being non-decreasing, another advantage of MGRU is that the gradients do not vanish due to the additive recurrence relation in Eq. 2. The inference procedure for time step t is summarized in Figure 2.

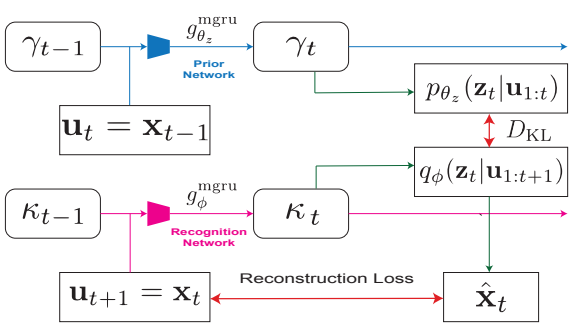

Fig. 2: Visualization of the approximate inference method.

b) Sampling: The KL divergence between the two Bernoulli distributions $q_\phi(\mathbf{z}_t|\mathbf{u}_{1:t+1})$ and $p_{\theta_z}(\mathbf{z}_t|\mathbf{u}_{1:t})$ can be computed in closed-form, so the gradient of the KL divergence term can back-propagate through the parameters ϕ and θ. To compute the expected reconstruction loss, however, we need to sample from $q_\phi(\mathbf{z}_t|\mathbf{u}_{1:t+1})$. Due to the discrete nature of the latent state $\mathbf{z}$, we can not use the reparameterization trick to sample from q_ϕ. Therefore, we can use the Gumbel-softmax trick, which allows us to obtained biased samples from q_ϕ [17]. In practice, we found that setting $\hat{\mathbf{z}} = \boldsymbol{\gamma}$, i.e., to its expected value and avoid sampling leads to comparable model fitting quality with lower empirical computational complexity.

c) Emission models: The output state at time t is $\mathbf{x}_t = \mathcal{E}_{t+1} = (\mathbf{c}_t, \mathbf{o}_t)$, where $\mathbf{c}_t, \mathbf{o}_t$ are one-hot encoded vectors representing the company and the job title for the user's professional experience at time step t. We use learnable embedding modules for all the fields including companies, job titles, and skills; the embedding modules $\mathbf{e}_c(k) \in \mathbb{R}^E$, $\mathbf{e}_o(l) \in \mathbb{R}^E$, $\mathbf{e}_s(j) \in \mathbb{R}^E$ project the kth company, l^{th} job title, and j^{th} observed skill to an embedding space of dimension E, respectively.

For the task of predicting the next professional experience of a user, we treat the company and job title fields separately. We use nonlinear embedding modules (parameterized by fully-connected neural networks) $\mathbf{f}_c(\cdot)$ and $\mathbf{f}_o(\cdot)$ to map the latent skill state $\mathbf{z}_t$ into the embedding space of companies and job titles. We then minimize the combined cross-entropy objective with the softmax emission function $p(\mathbf{x}_t|\mathbf{z}_t)$ for a (company, job title) pair as

$$-\log\Big(\frac{\exp(\mathbf{e}_c(k)^\intercal \mathbf{f}_c(\mathbf{z}_t))}{\sum_{k'=1}^{M_c}\exp(\mathbf{e}_c(k')^\intercal \mathbf{f}_c(\mathbf{z}_t))}\Big) - \log\Big(\frac{\exp(\mathbf{e}_o(l)^\intercal \mathbf{f}_o(\mathbf{z}_t))}{\sum_{l'=1}^{M_o}\exp(\mathbf{e}_o(l')^\intercal \mathbf{f}_o(\mathbf{z}_t))}\Big),$$

where M_c, M_o are the number of unique companies and job titles, respectively. However, since these numbers are likely very large, the denominators in the loss function cannot be computed efficiently. Therefore, we employ a negative sampling approach and convert the multi-class classification problem into a series of binary classification problems, using a small number of randomly sampled companies and job titles as negative examples [18], [19]. Concretely, we minimize the following objective for company prediction:

$$-\log(\sigma(\mathbf{e}_c(k)^\intercal \mathbf{f}_c(\mathbf{z}_t)) - \textstyle\sum_{k'=1}^{N_s}\log(\sigma(-\mathbf{e}_c(k')^\intercal \mathbf{f}_c(\mathbf{z}_t)),$$

where k' indexes a total of N_s negative samples from a uniform distribution among companies that do not correspond to the user's professional experience at time step t. The objective for job title prediction is defined similarly. We include this objective term for professional experiences but not educational experiences.

For the task of predicting the observed skill set $\mathcal{S}$, we use a two-step approach. First, we take the user's current professional experience $\mathbf{x}_T$ and use it to estimate the user's current latent skill state, $\mathbf{z}_{T+1}$, from $p_{\theta_z}(\mathbf{z}_{T+1}|\mathbf{x}_{1:T})$. Second, we use a (linear) embedding module $\mathbf{f}_s(\cdot)$ parameterized by the matrix $\mathbf{W}_s \in \mathbb{R}^{D\times E}$ to map $\mathbf{z}_{T+1} \in [0,1]^D$ into the embedding space of observed skills as

$$\mathbf{f}_s(\mathbf{z}_{T+1}) = \textstyle\sum_{j'=1}^{D} z_{T+1,j'}[\mathbf{W}_s]_{j'} = \mathbf{W}_s^\intercal \mathbf{z}_{T+1} \in \mathbb{R}^E,$$

where $[\mathbf{W}_s]_{j'}$ denotes the vector in the j'^{th} row of $\mathbf{W}_s$, i.e., the embedding of latent skill j'. Since a user's observed skill set contains multiple skills, we predict these observed skills by minimizing the following objective:

$$-\sum_{j\in\mathcal{S}}\log(\sigma(\mathbf{e}_s(j)^\intercal \mathbf{f}_s(\mathbf{z}_{T+1})) + \sum_{j\notin\mathcal{S}}\log(\sigma(-\mathbf{e}_s(j)^\intercal \mathbf{f}_s(\mathbf{z}_{T+1})).$$

Since the total number of unique skills is significantly larger than the number of observed skills for each user, we employ negative sampling again to randomly select a small subset of unobserved skills in the second term of the loss function above for each user.

C. Career Path Recommendation

Using `MNSS`, we can trace a user's latent skill states $\mathbf{z}_t$ over time and compute their probability of attaining any job at any company at any point in time. These capabilities enable us to study career paths that connect a user to their desired job. We define the *career goal* of a user as a (company, job title)

Dataset	# Users	# Experiences	# Companies	# Job Titles	# Skills
LinkedIn	$1,136,231$	$6,281,572$	$89,126$	$56,773$	$16,976$
Indeed	$3,945,040$	$26,009,711$	$85,427$	$50,219$	$17,836$

TABLE I: Dataset statistics.

pair $(\mathbf{c}^*, \mathbf{o}^*)$; given their experiences until time step t, our task is to plan a career path, i.e., intermediate (company, job title) pairs, for them to eventually reach their career goal. For example, after completing a Bachelor's degree in Economics, a user is working in Deloitte Ltd. as an Analyst; our goal is to recommend a path consisting of a series of intermediate career "hops" that has the best chance of leading them to their career goal, which is to become a partner at Goldman Sachs.

Formally, for each user, given their experiences (both educational and professional) for the first t time steps as $[\mathcal{E}_1, \ldots, \mathcal{E}_t]$, our goal is to find a path $[\hat{\mathcal{E}}_{t+1}, \ldots, \hat{\mathcal{E}}_G]$, where $\hat{\mathcal{E}}_G = (\mathbf{c}^*, \mathbf{o}^*)$ is their career goal, with an arbitrary number of intermediate hops $[\hat{\mathcal{E}}_{t+1}, \ldots, \hat{\mathcal{E}}_{G-1}]$. For the path to be feasible to the user, we maximize the likelihood of the entire career path, including each intermediate hop, as

$$\max_{[\hat{\mathcal{E}}_{t+1},\ldots,\hat{\mathcal{E}}_{G-1}]} \prod_{t'=t+1}^{G} p(\hat{\mathcal{E}}_{t'} | \mathcal{E}_{1:t}, \hat{\mathcal{E}}_{t+1:t'-1}), \tag{4}$$

where we maximize over all paths consisting of valid combinations of companies and job titles.

a) Conditional optimal path: An additional relevant real-world scenario is that a user has several options for their next career hop and need to choose one. For example, after completing a Bachelor's degree in Computer Science, a user has offers to join a Ph.D. program in Economics, an MBA program, or a banking company as a developer. They must make a decision on which offer to accept in order to maximize the chance of reaching their final career goal, which is to become a partner at McKinsey & Co. We denote these K choices (after time step t) as $\{\hat{\mathcal{E}}^1_{t+1}, \cdots, \hat{\mathcal{E}}^K_{t+1}\}$ and recommend the user to choose the one that maximizes the likelihood of their subsequent career path to the career goal as

$$\max_{\hat{\mathcal{E}}^k_{t+1} \in \{\hat{\mathcal{E}}^1_{t+1},\ldots,\hat{\mathcal{E}}^K_{t+1}\}} \sup_{\hat{\mathcal{E}}_{t+2},\ldots,\hat{\mathcal{E}}_{G-1}} \prod_{t'=t+2}^{G} p(\hat{\mathcal{E}}_{t'} | \mathcal{E}_{1:t}, \hat{\mathcal{E}}^k_{t+1}, \hat{\mathcal{E}}_{t+2:t'-1}). \tag{5}$$

This problem setting is also inspired by do-calculus in causal inference, where one is interested in modeling the effect of setting a random variable ($\mathcal{E}_{t+1}$) to a particular value and study its effect on other random variables [20].

IV. Experimental Results

We start by conducting quantitative experiments on the standard company, job title, and skill prediction tasks and show that `MNSS` outperforms baselines. We then conduct several qualitative experiments to show that our model i) extracts insights on career paths from real data and ii) provides actionable feedback and career path recommendations to users.

A. Dataset

We use two publicly available datasets containing user career profiles, extracted from LinkedIn[4] and Indeed[5]. A user profile consists of educational and professional experiences along with their skill set. Table I lists the number of user profiles, experiences, and unique skills, companies, job titles for both datasets. In each dataset, the numbers of unique entities, including skills, companies, job titles, schools, degrees, and majors are very large and most of them only occur in very few profiles. Therefore, we use a threshold to filter out infrequent entities and denote the filtered out ones as "Unknown" and do not include them in both training and testing. To remove outliers, we filter user profiles (around $\sim 0.5\%$ users) with more than $10/20$ professional experiences in the Indeed/Linkedin datasets. For all datasets, we sort user experiences based on their start times in chronological order.

B. Experimental Setup

Quantitatively, we measure our model's ability to predict professional experiences and skills listed in each user's profile. In each time step, we predict a user's next professional experience, i.e., the company and title of their next job; After the final time step, we use all past experiences to predict the user's listed skill set. We observe that around 20% of the time, a user's next job is either within the same company or has the same job title as the previous one. Since these transitions are easy to predict, we only evaluate job transitions where the user's next company and job title are different than the previous ones.

a) Evaluation measures and baselines: For each entity that we are predicting (company, job title, skill), we rank all unique entities in the dataset by sorting their predicted likelihood in descending order. We use mean percentile rank (MPR) and precision@K (P@K) for $K \in \{10, 100\}$ as the evaluation metrics of predictive quality. Using companies as an example, we define MPR as

$$\text{MPR} = \textstyle\sum_i \sum_{t=1}^{T_i} \mathbb{1}_{\text{obs}} \text{rank}(\mathbf{c}^i_t)/ZC,$$

where $\mathbf{c}^i_t$ denotes the actual company user i worked on at time step t and C denotes the total number of unique companies. $\mathbb{1}_{\text{obs}}$ is the indicator function with value 1 when the event is professional (we do not predict educational experiences) and the entity is not "Unknown", and 0 otherwise. $Z = \sum_i \sum_t \mathbb{1}_{\text{obs}}$ is the normalizing factor that counts all the observed occurrences. In other words, MPR corresponds to the average percentile rank for the observed entity in our prediction; smaller MPR values correspond to higher predictive quality. Similarly, we define P@K as

$$\text{P@K} = \textstyle\sum_i \sum_{t=1}^{T_i} \mathbb{1}_{\text{obs}} \mathbb{1}\big(\text{rank}(\mathbf{c}^i_t) \leq K\big))/Z,$$

where $\mathbb{1}(\text{rank} \leq K)$ is another indicator function that has value 1 if the actual observed company is ranked at or above

[4]The dataset was retrieved from https://www.kaggle.com/linkedindata/linkedin-crawled-profiles-dataset, but no longer available.

[5]https://datastock.shop/download-indeed-job-resume-dataset/

Performances		Skills			Companies			Job Titles		
Dataset	Methods	MPR ↓	P@10 ↑	P@100 ↑	MPR ↓	P@10 ↑	P@100 ↑	MPR ↓	P@10 ↑	P@100 ↑
LinkedIn	Job2Vec	7.16 ± 0.03	15.99 ± 0.03	47.69 ± 0.05	29.14 ± 0.05	10.09 ± 0.06	13.9 ± 0.2	25.74 ± 0.02	11.87 ± 0.09	20.93 ± 0.07
	HRM	3.82 ± 0.02	26.18 ± 0.09	65.11 ± 0.05	5.38 ± 0.02	13.98 ± 0.07	37.4 ± 0.09	4.94 ± 0.01	17.9 ± 0.1	42.13 ± 0.06
	NEMO	3.69 ± 0.02	26.21 ± 0.06	66.01 ± 0.05	5.15 ± 0.03	13.99 ± 0.1	38.0 ± 0.1	4.58 ± 0.01	19.24 ± 0.05	44.15 ± 0.07
	NSS	2.94 ± 0.02	27.66 ± 0.07	67.62 ± 0.06	$\mathbf{4.72 \pm 0.01}$	$\mathbf{15.63 \pm 0.08}$	$\mathbf{40.85 \pm 0.1}$	4.17 ± 0.01	$\mathbf{20.62 \pm 0.03}$	46.07 ± 0.04
	MNSS	$\mathbf{2.81 \pm 0.01}$	$\mathbf{27.79 \pm 0.09}$	$\mathbf{67.69 \pm 0.04}$	4.75 ± 0.01	15.58 ± 0.06	40.65 ± 0.08	$\mathbf{4.13 \pm 0.01}$	20.52 ± 0.08	$\mathbf{46.15 \pm 0.08}$
Indeed	Job2Vec	8.89 ± 0.07	22.59 ± 0.05	50.42 ± 0.08	28.15 ± 0.03	$\mathbf{12.85 \pm 0.08}$	18.5 ± 0.07	21.31 ± 0.03	15.07 ± 0.06	25.82 ± 0.08
	HRM	7.23 ± 0.09	26.32 ± 0.04	57.93 ± 0.08	12.18 ± 0.04	7.49 ± 0.02	22.09 ± 0.05	5.75 ± 0.01	18.37 ± 0.02	42.85 ± 0.02
	NEMO	7.99 ± 0.17	25.77 ± 0.05	57.18 ± 0.12	9.85 ± 0.04	8.17 ± 0.03	24.24 ± 0.03	4.99 ± 0.01	19.85 ± 0.04	45.11 ± 0.03
	NSS	6.52 ± 0.11	26.21 ± 0.03	58.45 ± 0.07	9.71 ± 0.04	8.25 ± 0.02	24.33 ± 0.06	4.96 ± 0.02	19.96 ± 0.04	45.49 ± 0.04
	MNSS	$\mathbf{4.61 \pm 0.05}$	$\mathbf{26.6 \pm 0.03}$	$\mathbf{59.16 \pm 0.08}$	$\mathbf{8.97 \pm 0.04}$	8.71 ± 0.05	$\mathbf{25.24 \pm 0.12}$	$\mathbf{4.81 \pm 0.02}$	$\mathbf{20.29 \pm 0.03}$	$\mathbf{46.15 \pm 0.08}$

TABLE II: Predictive performance on companies, job titles, and skills on both datasets and all three metrics for all methods. Best performances across all methods are in bold.

K in our prediction. In other words, P@K corresponds to the frequency that the observed entity is within the top K predicted entities; larger P@K values correspond to higher predictive quality. The corresponding metrics for job title and skill prediction tasks are defined similarly, with the exception that we do not sum over all time steps for skill prediction.

We compare MNSS against three baselines. We do not discuss simple baselines such as recommending the most frequent skills/job titles/companies and bigrams since they do not perform well. The first baseline, which we dub the Job2Vec method, learns embeddings of job titles, companies, and skills from directed transition graphs of users between jobs [6]. For each user, we only use the job title/company embedding of their previous professional experience to predict the job title/company of their next career experience. We also use the job title embedding of a user's last professional experience to predict their skill set. The second baseline, the hierarchical deep representation learning (HRM) method [21], does not introduce recurrence in the latent skill states. Instead, HRM aggregates the embeddings of all past experiences using max-pooling and uses it as the latent skill state, which is used to predict the next professional experience. The third baseline, the NEMO method [4], is the current state-of-the-art in next job title and company prediction to our problem setting. We slightly modify NEMO to predict a user's skill set instead of using it as input to the LSTM model to predict future job titles and companies since we do not know when a user acquires their listed skills. We also experimented with a version of MNSS without the monotonic constraints, which we dub NSS.

b) Training and testing: For evaluation purposes, we perform standard k-fold cross-validation (with $k = 5$) for all models and all datasets. Thus, for each fold, 20% user profiles are used as the test set, 20% are used as the validation set, and 60% are used as the training set. The validation set is used for parameter tuning and the test set is used for evaluation of the tuned methods.

c) Implementation details: At each time step t, we compute the input state for all deep models (except NEMO) as $\mathbf{u}_t = [\text{Education_Vector}_t \oplus \text{Profession_Vector}_t]$, where $\oplus$ denotes vector concatenation. We concatenate embeddings for School, Major, Degree, Start Time, Duration to represent the contextual $\text{Education_Vector}_t$ and we concatenate embeddings for Company, Job Title, Start Time, Duration to represent the contextual $\text{Profession_Vector}_t$. For NEMO, we use education vectors as input to the initial latent skill states and use the profession vectors as input in later time steps to predict the next job title and company [4]. Using inputs $\mathbf{u}_{1:t}$, HRM uses max-pooling operation, NEMO and NSS use LSTM/GRU, and MNSS uses our monotonic state-space model to update the latent skill state $\mathbf{z}_t$, respectively.

Since companies, job titles, and skills are likely to be closely associated with the semantic meanings of the words in their names, we initialize the embedding vectors for these entities using the average Glove embedding vector (300 dimensional) [22]. For other fields, we use smaller embedding dimensions as follows: $\text{Major}_t, \text{Degree}_t, \text{School}_t \in \mathbb{R}^{64}$ and $\text{Start_Time}_t, \text{Duration}_t \in \mathbb{R}^{32}$. Additionally, the Linkedin dataset contains industry information. On the Linkedin dataset, for all methods, we concatenate the industry embeddings $\in \mathbb{R}^{64}$ with the latent states $\mathbf{z}_t$ for the prediction tasks. Since industry, school, major, and degrees are less likely to have semantically close associations, we initialize them from a standard normal distribution. For all methods, we fix the embedding dimensions; we do not see any improvement using larger embedding dimensions. For all the other network parameters, we use the Xavier initializer [23]. For our model MNSS, we employ the annealing trick for stabilizing the training procedure similar to other stochastic recurrent neural network models [12]; we anneal the value of β from 0 to 0.1 from iteration 1 to iteration 20 in our MNSS model.

We use the Adam optimizer [24] to train our model and use dropout as the regularization method [25]. For all methods, we use $\{10^{-3}, 10^{-4}\}$ as values of the learning rate in the Adam optimizer. For all methods, we use a fixed batch size of 64 and we fix the dropout parameter to 0.2; as the datasets are large, we observe minimal differences with regularization. For all methods, we use $\{256, 512, 1024\}$ as values of the latent state dimension; we do not observe any noticeable differences in performances for different choices of the latent state dimension. For all trainable models, we do early stopping in our training process using the validation set. We implement all the methods in PyTorch [26] and train all models on a single NVIDIA Titan X GPU.

C. Results and Discussion

Table II lists means and standard deviations of the predictive performances on companies, job titles, and skills for all models in the MPR, P@10, and P@100 metrics, on both datasets. We see that MNSS and NSS outperforms all other baselines in

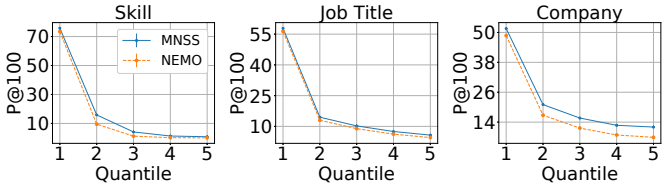

Fig. 3: Performance across quantiles of skill, job title, and company prediction according to how often they occur.

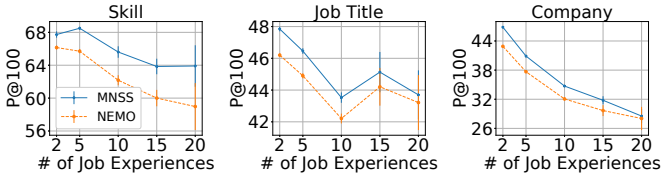

Fig. 4: Performance across user groups according to the number of career experiences listed in their profiles.

most cases. Specifically, MNSS outperforms the state-of-the-art baseline NEMO by $5-20\%$ on the company, job title, skill prediction tasks in MPR on both datasets. Similarly, in P@10 and P@100, we observe that MNSS significantly outperforms NEMO (by $5-20\%$ on the LinkedIn dataset and $1-5\%$ on the Indeed dataset). We also observe that MNSS slightly outperforms NSS on the skill prediction task. On the company and job title prediction tasks, MNSS performs on par with NSS in all metrics on both datasets. This result suggests that the interpretable monotonic latent skill states in MNSS are well-suited to model user upskilling and reskilling processes, as evident by its superior performance on the skill prediction task; on the other hand, for the standard job title and company prediction tasks, uninterpretable black-box models such as NEMO are also applicable. The graph embedding-based method, Job2Vec, does not perform well in our experiments (except for P@10 on the Indeed dataset); a possible reason is that it does not take a user's entire past career history into account and is not personalized. It performs relatively better in P@10 compared to MPR since graph embeddings are effective at finding closely related companies and job titles. On the contrary, RNN-based methods like NEMO, NSS, and MNSS significantly outperform other baselines since the recurrence in their latent states can capture users' evolving skill sets and extract information from their past carer experiences.

To further understand where the performance gain of MNSS over NEMO comes from, we compare their performance on the different entities and user groups. In Figure 3, we divide skills, job titles, and companies according to how often these entities occur and visualize the predictive performance in P@100 for entities in different quantiles. We observe that MNSS consistently outperforms NEMO across all quantiles, especially in the middle quantiles. In Figure 4, we divide users into groups according to the number of career experiences listed in their profiles and visualize the predictive performance across groups; the width of error bars increase as the number of career experiences increase since there are fewer users in these groups. MNSS consistently outperforms NEMO and especially in skill prediction for users with long career histories, suggesting that it is effective in monitoring users' upskilling processes.

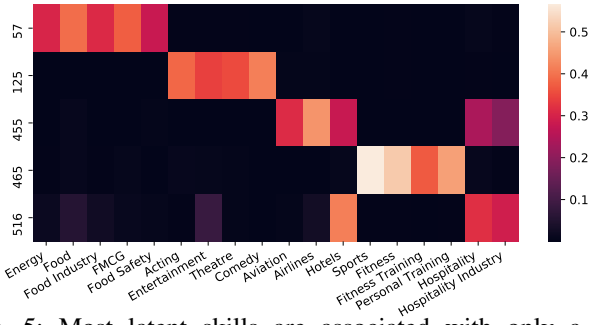

Fig. 5: Most latent skills are associated with only a few observed skills.

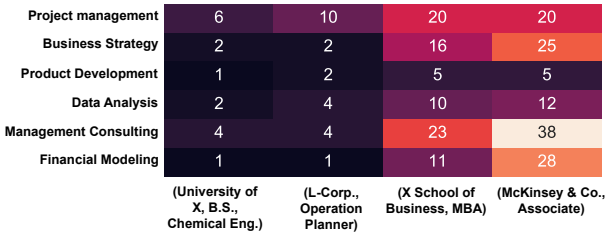

Fig. 6: The reconstructed skill acquisition process with each career experience for a user using MNSS.

In what follows, we conduct a series of qualitative experiments and present insights we obtained from the datasets and show the excellent interpretability of MNSS. We cannot obtain these insights from uninterpretable baselines such as NEMO; however, we have to omit qualitative comparisons due to spatial constraints.

a) Mapping latent skills to observed skills: Since the number of latent skills (D) is significantly less than the number of observed skills (M), we can interpret the meaning of each latent skill by examining the subset of observed skills that are the most closely associated with it. Let $\mathbf{E}_s \in \mathbb{R}^{E\times M}$ denote the matrix containing all observed skill embeddings $\mathbf{e}_s(j)$, the entries of the product matrix $\mathbf{W}_s\mathbf{E}_s \in \mathbb{R}^{D\times M}$ quantify the similarity between each latent skill and each observed skill. We visualize a re-arranged subset of $\mathbf{W}_s\mathbf{E}_s$ in Figure 5, where rows correspond to latent skills and columns correspond to observed skills; brighter colors represent higher similarities. We see that, as expected, each latent skill is closely associated with only a few observed skills; for example, latent skill 465 is closely associated with observed skills "Sports", "Fitness", "Fitness Training", and "Personal Training", while latent skill 125 is closely associated with observed skills "Acting", "Entertainment", "Theatre", and "Comedy". We found that without the monotonic property of MNSS, other baselines do not learn clear associations between latent skills and observed skills.

b) Skill acquisition process: The monotonic latent skill states in MNSS enable us to reconstruct a user's skill acquisition process through different professional experiences in their career. Specifically, we visualize the probability of a user mastering each observed skill in Figure 6. The number in cell (j,t) corresponds to the probability ($\times 100$) that the user masters observed skill j after the professional experience

Past Experiences	Skills to be Improved
Goal: (Google, Engineering Manager)	
(Duke, Computer Science, B.S.) → (CGI Inc., Senior Consultant) → (Cake Financial, Director)	C++, Product Development, C, Java, Python
(MIT, Computer Science, B.S.) → (CMU, Computer Science, M.S.)	Project Management, Product Development, Cross-functional Team Leadership

TABLE III: Top skills that need to be improved to reach the career goal.

Past Experiences	Optimal Path	Log-likelihood
Goal: (Goldman Sachs, Managing Director)		
(PKU, Physics, B.S.) → (UIUC, Physics, Ph.D.)	(Morgan Stanley, Quantitative Analyst) → (Knight Capital, Analyst) → (Knight Capital, Algorithmic Trading Developer)	-0.985

TABLE IV: Optimal career path towards a user's career goal.

at time step t. We see that MNSS is able to discover how professional experiences improve a user's skill. For example, this user acquired most of the skills listed, especially Project Management, Business Strategy, and Data Analysis after getting an MBA degree from a business school, and significantly improved on Management Consulting and Financial Modeling after becoming an Associate at McKinsey & Co.

c) Skill gap analysis: For each user, we can analyze the *skill gap* they are facing between their current estimated latent skill state and the desired latent skill state to reach their career goal. In order to analyze the skill gap, we take the trained model and solve the following optimization problem

$$\begin{aligned} \text{minimize}_{\tilde{\mathbf{z}}} \quad & -\log p\big((\mathbf{c},\mathbf{o})|\tilde{\mathbf{z}}\big) + \lambda\|\mathbf{z}_t - \tilde{\mathbf{z}}\|_1 \\ \text{subject to} \quad & \tilde{\mathbf{z}} \in [0,1]^D, \tilde{z}_j \geq z_{t,j}, \forall j \in \{1,\cdots,D\}, \end{aligned}$$

where $\lambda > 0$ is a regularization parameter. In other words, we search for a new latent skill state $\tilde{\mathbf{z}}$ by minimizing the negative log-likelihood of reaching the career goal $(\mathbf{c},\mathbf{o})$; Moreover, since it is not realistic to ask a user to improve all skills, we use an ℓ_1-norm penalty to promote new latent skill states that requires improvements on only a *sparse* set of latent skills. We solve this problem using the projected gradient descent algorithm [27]. We then map this latent skill gap to the space of observed skills by calculating the required increase in the probability of mastering each observed skill as $\sigma(\mathbf{e}_s(j)^\intercal \mathbf{f}_s(\mathbf{z}^*)) - \sigma(\mathbf{e}_s(j)^\intercal \mathbf{f}_s(\mathbf{z}_t))$, where $\mathbf{z}^*$ denotes the solution to the optimization problem above.

We list the observed skills that require the most improvement for two users to reach the same career goal, Engineering Manager at Google, in Table III. In this case, although both users started with a Bachelors's degree in Computer Science, they need to improve different skills. One user worked in consulting and management after graduation and needs to improve technical skills like C++, Product Development, and Java, while the other user who pursued graduate studies needs to improve management skills like Project Management and Cross-functional Team Leadership. These results show that MNSS has the potential to provide personalized, actionable feedback to help users plan their personal upskilling process and reach their career goals.

Career Goals	Previous Experience	**(U. Toronto, Ph.D., Computer Science)**	**(Microsoft Inc., Software Engineer)**	**(Harvard, MBA)**
(Georgia Tech, Assistant Professor)	(Stanford, B.S., Computer Science)	-1.722	-4.217	-6.121
(Google, Engineering Manager)	(MIT, B.S., Computer Science)	-0.5863	-0.4390	-0.9391
(McKinsey & Co., Associate)	(Harvard, B.S., Economics)	-2.3868	-2.018	-0.4456

TABLE V: Options each user faces and the log-likelihood of reaching their career goals by taking each option.

D. Career Path Recommendation

As detailed in Section III-C, given a user's past career experiences $\mathcal{E}_{1:t}$ and a future career goal $\hat{\mathcal{E}}_G$, we can plan a path of career hops $\hat{\mathcal{E}}_{t+1:G}$ to connect them to their career goal by maximizing the overall likelihood of the entire path. However, computing the likelihood of all possible paths between a certain start state and a certain goal state is computationally intractable. Therefore, we use beam search [28] to identify the most feasible career paths by using the latent skill state distributions $p(\mathbf{z}_{t+1}|\mathcal{E}_{1:t})$ to compute the probabilities in Eq. 4 and Eq. 5. In our experiments, we found that with a modest number of paths, beam search returns high-quality career paths. We set the maximum number of intermediate hops to 5 and keep the best 10 paths during beam search, with a small modification: we select 10 most likely companies and 10 most likely job titles at each intermediate time step, and randomly sample 10 companies and job titles pairs from the 100 total combinations. This modification brings some balance between the two fields, especially when one company or job title has a significantly higher likelihood than others, which will result in that company or job title being the only one included in the beam search process.

We show the sampled optimal career path for one user with the career goal of Managing Director at Goldman Sachs and the corresponding log-likelihood of reaching that goal, given their past career experiences, in Table IV. In this case, the path consists of the intermediate hops of Quantitative Analyst, Analyst, and Algorithmic Trading Developer at different banking companies. The path takes the user's Physics background into account and suggests that quantitative work is a good first hop for the user to go into banking. We also show multiple options for next career experience and the corresponding log-likelihoods of reaching their career goal for three users in Table V. For example, for a user graduating with a Bachelors' Degree in Computer Science from Stanford to reach their career goal of becoming an Assistant Professor at Georgia Tech, the option of obtaining a Ph.D. Degree in Computer Science is their best option among other options including becoming a Software Engineer and obtaining an MBA degree. These results show that MNSS can help users plan career goals and decide on the next professional experience when facing multiple options, in order to maximize their chances of reaching their career goals.

V. Conclusions and Future Work

We have proposed an interpretable monotonic nonlinear state-space model for career path modeling and analyzed two large-scale online user professional profile datasets. Experimental results show that our model achieves excellent predictive performance on the tasks of the company, job title, and skill prediction. Moreover, we used a series of case studies to show that our model is interpretable and can be used to provide actionable feedback to users on the skills they need to acquire and recommendations on feasible career paths they can take to achieve their desired career goal.

There are numerous avenues for future work to address the limitations of our work. First, since real-life constraints including geographical, transportation, and family constraints are significant factors in a user's career decision, they should be taken into account by the career path planning algorithm. Second, since a detailed career path also contains not only jobs but also the duration of these jobs, there is a need to use path search algorithms that take duration into account, given a user's desire to reach a career goal within a certain amount of time. Third, since different users have different ideologies, we need to explore other formulations of the career path planning problem, e.g., maximizing total career earnings. Fourth, since the numbers of companies, job titles, and skills are large, there is a need to use taxonomies such as O*NET [29] or language models [30] to standardize the data, resulting in more compact entity representations. Fifth, since the sample of online user professional profiles is inevitably biased toward high-skill users who have enough digital skills to craft these profiles, there is a need to include more diverse data sources to promote fairness and benefit every user equally.

Acknowledgement

This work is supported by the NSF under grant no. CA-FW-HTF-1936915.

References

[1] Pew Research Center, "The future of jobs and job training," online: https://www.pewinternet.org/2017/05/03/the-future-of-jobs-and-jobs-training/, 2017.

[2] D. Restuccia and B. Taska, "Different skills, different gaps: Measuring and closing the skills gap," *Developing Skills in a Changing World of Work: Concepts, Measurement and Data Applied in Regional and Local Labour Market Monitoring Across Europe*, pp. 207–225, 2018.

[3] LinkedIn Corp., "The LinkedIn economic graph," Online: https://economicgraph.linkedin.com/, 2020.

[4] L. Li, H. Jing, H. Tong, J. Yang, Q. He, and B.-C. Chen, "Nemo: Next career move prediction with contextual embedding," in *Proc. International Conference on World Wide Web Companion*, 2017, pp. 505–513.

[5] V. Dave, B. Zhang, M. Al Hasan, K. AlJadda, and M. Korayem, "A combined representation learning approach for better job and skill recommendation," in *Proc. International Conference on Information and Knowledge Management*, 2018, pp. 1997–2005.

[6] D. Zhang, J. Liu, H. Zhu, Y. Liu, L. Wang, P. Wang, and H. Xiong, "Job2vec: Job title benchmarking with collective multi-view representation learning," in *Proc. ACM International Conference on Information and Knowledge Management*, 2019, pp. 2763–2771.

[7] W. Wu, J. Yan, X. Yang, and H. Zha, "Decoupled learning for factorial marked temporal point processes," in *Proc. ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 2018, pp. 2516–2525.

[8] Q. Meng, H. Zhu, K. Xiao, L. Zhang, and H. Xiong, "A hierarchical career-path-aware neural network for job mobility prediction," in *Proc. ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 2019, pp. 14–24.

[9] H. Li, Y. Ge, H. Zhu, H. Xiong, and H. Zhao, "Prospecting the career development of talents: A survival analysis perspective," in *Proc. ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2017, pp. 917–925.

[10] J. Chung, C. Gulcehre, K. Cho, and Y. Bengio, "Empirical evaluation of gated recurrent neural networks on sequence modeling," *arXiv preprint arXiv:1412.3555*, 2014.

[11] S. Roweis and Z. Ghahramani, "A unifying review of linear gaussian models," *Neural Computation*, vol. 11, no. 2, pp. 305–345, 1999.

[12] M. Fraccaro, S. K. Sønderby, U. Paquet, and O. Winther, "Sequential neural models with stochastic layers," in *Proc. Advances in Neural Information Processing Systems*, 2016, pp. 2199–2207.

[13] R. Krishnan, U. Shalit, and D. Sontag, "Structured inference networks for nonlinear state space models," in *Proc. AAAI Conference on Artificial Intelligence*, 2017.

[14] D. Kingma and M. Welling, "Auto-encoding variational Bayes," in *Proc. International Conference on Learning Representations*, 2014.

[15] N. Du, H. Dai, R. Trivedi, U. Upadhyay, M. Gomez-Rodriguez, and L. Song, "Recurrent marked temporal point processes: Embedding event history to vector," in *Proc. ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2016, pp. 1555–1564.

[16] E. Denton and R. Fergus, "Stochastic video generation with a learned prior," in *Proc. International Conference on Machine Learning*, 2018, pp. 1182–1191.

[17] E. Jang, S. Gu, and B. Poole, "Categorical reparametrization with gumbel-softmax," in *Proc. International Conference on Learning Representations*, 2017.

[18] T. Mikolov, I. Sutskever, K. Chen, G. Corrado, and J. Dean, "Distributed representations of words and phrases and their compositionality," in *Proc. Conference on Advances in Neural Information Processing Systems*, 2013, pp. 3111–3119.

[19] M. Gutmann and A. Hyvärinen, "Noise-contrastive estimation: A new estimation principle for unnormalized statistical models," in *Proc. International Conference on Artificial Intelligence and Statistics*, 2010, pp. 297–304.

[20] J. Pearl *et al.*, "Causal inference in statistics: An overview," *Statistics Surveys*, vol. 3, pp. 96–146, 2009.

[21] P. Wang, J. Guo, Y. Lan, J. Xu, S. Wan, and X. Cheng, "Learning hierarchical representation model for nextbasket recommendation," in *Proc. International ACM SIGIR conference on Research and Development in Information Retrieval*, 2015, pp. 403–412.

[22] J. Pennington, R. Socher, and C. Manning, "Glove: Global vectors for word representation," in *Proc. Conference on Empirical Methods in Natural Language Processing*, 2014, pp. 1532–1543.

[23] X. Glorot and Y. Bengio, "Understanding the difficulty of training deep feedforward neural networks," in *Proc. International Conference on Artificial Intelligence and Statistics*, 2010, pp. 249–256.

[24] D. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *Proc. International Conference on Learning Representations*, 2015.

[25] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: A simple way to prevent neural networks from overfitting," *The Journal of Machine Learning Research*, vol. 15, no. 1, pp. 1929–1958, 2014.

[26] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga *et al.*, "Pytorch: An imperative style, high-performance deep learning library," in *Proc. Conference on Advances in Neural Information Processing Systems*, 2019, pp. 8024–8035.

[27] A. Madry, A. Makelov, L. Schmidt, D. Tsipras, and A. Vladu, "Towards deep learning models resistant to adversarial attacks," in *Proc. International Conference on Learning Representations*, 2018.

[28] I. Sutskever, O. Vinyals, and Q. Le, "Sequence to sequence learning with neural networks," in *Proc. Advances in Neural Information Processing Systems*, 2014, pp. 3104–3112.

[29] National Center for O*NET Development, "O*NET online," Online: https://www.onetonline.org/, 2020.

[30] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of deep bidirectional transformers for language understanding," in *Proc. Conference of the North American Chapter of the Association for Computational Linguistics*, 2019, pp. 4171–4186.