Early Inference of Nuclear Technology-Directed Research Activities of Authors from Scientific Publications

Dennis G. Thomas [a,*], Zachary J. Weems [a], Richard E. Overstreet [a], Benjamin A. Wilson [a]

[a] Pacific Northwest National Laboratory, 902 Battelle Blvd, Richland, WA, USA, 99354
[*] E-mail: dennis.thomas@pnnl.gov

Abstract:

Nuclear research articles can provide information about early nuclear proliferation indicators such as influential research entities and technology capability levels of a country, but detection of nuclear activities typically occurs after they have started. We investigate the extent to which nuclear research articles can be used to infer whether a research entity will acquire or develop a nuclear technology before it happens. Early detection of nuclear proliferation or technology development indicators from data is challenging due to partial observability, sparse and unlabeled information, and confounding signals from multiple concurrent activities. This paper presents the early detection problem as a sequential decision-making, goal inference problem, where the objective is to characterize and predict an individual's, organization's, or a country's intent (unobserved goal-directed behavior) towards developing a nuclear capability from partially observed sequences of their research publications, using inverse reinforcement learning and Bayesian goal inference methods. A computational framework is presented, and its application demonstrated using 29,196 Scopus records for a case study related to a civil nuclear capability. The case study results serve as a proof-of-concept demonstration for inference of technology-directed research activity of authors who publish in the nuclear domain. The inference method, combined with advanced computing, may be used to assess and monitor activities pertaining to early developmental stages of a nuclear technology or capability, which in turn can help to identify and prioritize activities with nuclear proliferation potential for further investigation.

Keywords: Nuclear, Reinforcement, Learning, Bayesian, Inference, AI

1. Introduction

The goal of nuclear proliferation detection is to deter state and non-state actors from pursuing the development and acquisition of nuclear weapons. Traditional methods for nuclear proliferation detection focus on detecting proliferation indicators such as chemical signatures and activities associated with the acquisition and production of special nuclear materials (Sheffield, 2020). Advanced data-driven methods are required to enable detection of proliferation indicators that are not only associated with special nuclear materials but also with the research, development and acquisition of specialized equipment, and technical expertise needed for building a nuclear weapon (Sheffield, 2020; Alexander et al., 2020). Such methods may also enable the detection of undeclared nuclear materials and activities from technical sources of information that are relevant for IAEA safeguards (Barletta et al., 2014; Carlson et al., 2006; Cojazzi et al., 2013; Ferguson and Norman, 2010; Pabian et al., 2014).

Recent advancements in computing, data science, and artificial intelligence / machine learning (AI/ML) technologies might offer opportunities for enhancing current data-driven detection methods to characterize and detect nuclear proliferation indicators at earlier stages of material production or nuclear weapon development (Sheffield, 2020). For example, scientific publications and networks (e.g., coauthorship, citation networks) have been analyzed using machine learning and natural language processing (NLP) techniques to identify early potential proliferation indicators such as influential entities in a research topic (Chatterjee et al., 2023), or the level of an entity's nuclear expertise and technology capability (Kas et al., 2012). Here, an entity can be a person, organization, city, state, or a country. Early detection of nuclear proliferation indicators from data is however challenging due to partial observability, sparse and unlabeled information, confounding signals from multiple concurrent activities, and difficulty in differentiating between peaceful and detrimental nuclear activities.

Nuclear research articles may provide valuable insights into an entity's intent (unobserved goal-directed behavior) to develop or acquire nuclear expertise and technology. Technical documents

and publications contain information to characterize and detect potential early proliferation indicators such as influential researchers, author collaborative patterns, nuclear expertise levels, and technology capabilities (Chatterjee et al., 2023; Kas et al., 2012). Recently, Chatterjee et al. (2023) performed a case study with author collaboration networks that were constructed using text (titles, abstract) and metadata of 33,517 Scopus records of nuclear research articles published from 2000 to 2019. They applied topic modeling and topic-aware influence maximization algorithms to identify authors who are the most influential in diffusing information about a selected topic mixture through collaboration networks. They also analyzed the collaboration dynamics of the influential authors over time, such as their ability to maintain old and to start new collaborations. Information about author influence and their collaborative patterns or behavior may be used to assess advancements in technological capabilities and expertise by key players (at the individual, organization, city, state, or country level) in a nuclear research area. Kas et al. (2012) used author affiliation information from 20,000+ nuclear physics articles in arXiv to extract coauthorship networks and to identify key players (authors, countries) based on network centrality measures. They also used text mining tools to extract information about nuclear processes from full-text contents of the articles for assessing the nuclear expertise level of countries. These studies indicate that scientific publications are useful for identifying key players and for assessing their nuclear expertise and technological capability levels, which in turn can help with early proliferation detection.

Instead of assessing the nuclear expertise or technology capability levels of an entity, we take the problem of early detection a step further to predict whether a research entity will attain a nuclear expertise or technological capability by observing their temporal sequences of past activities. Drawing inspiration from the literature on inference of driver route behavior and destinations from partial trip trajectories (Krumm and Horvitz, 2006; Snoswell et al., 2020; Xue et al., 2015; Ziebart et al., 2008), we develop a novel computational framework based on topic modeling, inverse reinforcement learning (Adams et al., 2022; Arora and Doshi, 2021; Ng and Russell, 2000) and Bayesian goal inference methods to predict the technology-directed publication behavior (intent) of authors from partial trajectories of their publication sequences in time. To our knowledge, this work represents the first attempt at using research articles and reinforcement learning framework to formulate and solve a sequential decision-making problem for modeling and predicting author behavior towards developing a technology or performing a research activity in a nuclear technology area.

The remainder of this paper is organized as follows. First, we briefly describe the reinforcement learning (RL) and inverse RL frameworks, followed by the problem of Bayesian goal inference from partial trajectories (sequences) of state transitions. Next, we formulate the sequential decision-making problem for technology-directed goal inference and present a case study for the approach. We then describe the approach and present the results of the case study. Finally, we discuss the performance aspects of the method and future research directions for technology-directed goal activity inference.

2. Background

2.1. Reinforcement learning

In the basic RL framework, a goal-seeking and domain-aware agent interacts with an external environment by performing various actions and learning what actions to take at each step to achieve a goal (Sutton and Barto 2018). An action performed by the agent causes a change in the internal and external environment (state) of the agent, and the environment in turn responds to the change by giving the agent a reward/penalty for taking the action. The agent learns to choose actions in such a way to maximize the total rewards accumulated along a sequence of state (s)-action(a)-state(s') transitions that lead to the goal state from any starting state. For example, let's consider a simple RL problem with a 5 x 5 grid world environment, as shown in Fig. 1. Each grid cell can be considered a state and an agent can move from one state to a neighboring state by choosing one of four actions: *left*, *right*, *up*, *down*. Given a destination cell (goal state), the RL agent must learn to take the most optimal sequence of transitions from any starting cell location (starting state) to the destination cell. Fig. 1 shows one such sequence from state S_1 to state S_{25}.

When the agent is in a state s, it must decide which action a to choose and to which state s' it should move to. This decision-making process depends on a Monte Carlo probabilistic criterion that uses two types of information: a transition probability and a reward for the (s, a, s') transition. In most RL problems, the sequence of (s, a, s') transitions is modeled as a first order

Markov decision process (MDP). That is, to choose the next action and the next state in a first order MDP, the agent in state s, does not need to have information about the path that it took to state s. A MDP can be finite or infinite, deterministic or stochastic. A finite MDP has a finite number of states like the grid world example in Fig. 1, which has 25 states. An infinite MDP has infinite number of states, i.e. the state space is continuous.

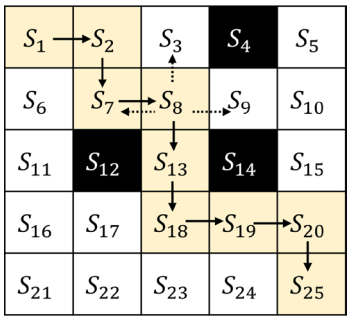

Figure 1. A 5 x 5 grid world environment with 25 states and 4 actions (left, right, up, down), illustrating a full sequence of state-action-state transitions (yellow-shaded cells) taken by a RL agent from state S_1 to goal state S_{25}. The dotted arrows from state S_8 indicate the actions that were not chosen by the agent along the path from S_1 to S_{25}. The black-shaded cells indicate inaccessible states.

To solve a finite MDP problem in the RL framework, we must specify five inputs: state space, action space, transition probabilities, reward functions, and a discount factor for calculating the discounted sum of rewards along a sequence. Thus, a finite MDP is typically denoted as a tuple (S, A, T, R, γ), where the state space, denoted as $S = \{S_i | i = 1, \dots, N\}$, is a finite set of N states; the action space, denoted as $A = \{A_j | j = 1, \dots, M\}$, is a finite set of M actions; T is the state-action-state transition probability matrix of size $N \times M \times N$, R is the reward function, and $\gamma \in (0,1)$ is the discount factor. Given these inputs, the RL agent computes a reward-based policy by which it determines an optimal sequence of state-action-state transitions from any starting state to the goal state. The policy is the probability of choosing an action a and the next state s', given the agent is in state s; which, in turn is a function of the state-action-state transition probabilities and the rewards. The optimal sequence is the sequence that would give the highest discounted sum of rewards from the starting state to the goal state. Various methods such as value iteration, policy iteration, Q-learning, Sarsa, have been developed to solve an RL problem (Sutton and Barto, 2018).

The reward function captures the goal-directed behavior of the agent. Reward functions must be designed specifically for the goal state of interest and in many cases have non-trivial structures. The reward along each transition is defined either independently or as functions (linear or non-linear) of state, state-action, and/or state-action-state features. The state-based reward, denoted as $R_1(s)$, is the reward for moving to state $s \in S$ from any state. The state-action reward, denoted as $R_2(s, a)$, is the reward for taking action $a \in A$ in state s. The state-action-state reward, denoted as $R_3(s, a, s')$, is the reward for moving to state $s' \in S$ from state s through action a. For example, let's consider a sequence, τ, of L state-action-state transitions, as ((s_1, a_1, s_2), (s_2, a_2, s_3), …, (s_t, a_t, s_{t+1}), …, $(s_L, \text{none}, \text{none})$), where s_1 is the starting state and s_L is the terminal (goal) state. The general formula for calculating the discounted sum of rewards for the sequence, $R(\tau)$, based on any combination of $R_1(s)$, $R_2(s, a)$, and $R_3(s, a, s')$ rewards, can be written as (Snoswell et al., 2020):

$$R(\tau) = \sum_{t=1}^{L} \gamma^{t-1} R_1(s_t) + \sum_{t=1}^{L-1} \gamma^{t-1} [R_2(s_t, a_t) + R_3(s_t, a_t, s_{t+1})]. \quad (1)$$

When γ is close to zero, the agent makes decisions by giving more importance to acquiring *immediate* rewards than *future* rewards. When γ is close to 1, the agent makes decisions by giving more importance to acquiring *future* rewards than *immediate* rewards.

A deterministic MDP is one where only one state is accessible when an action is performed, whereas in a stochastic MDP, two or more states may be accessible. The deterministic and stochastic MDP dynamics are modeled using the transition probability. If $T(i, j, k)$ represent the transition probability from state S_i to state S_k through action A_j, then by definition, $T(i, j, k) \in [0,1]$ and $\sum_{k=1}^{N} T(i, j, k) = 1$. For deterministic dynamics, $T(i, j, k) = 1$ for $k = k^*$, where $k^* \in \{1, \dots, N\}$ and $T(i, j, k) = 0$ for $k \neq k^*$. For example, the grid world environment in Fig. 1 is a finite MDP with 25 states and 4 actions: $S = \{S_i | i = 1, \dots, 25\}$ and $A = \{A_j | j = 1, .., 4\}$, where, we denote the actions, *left*, *right*, *up*, *down* as A_1, A_2, A_3, A_4, respectively In the grid world example (Fig. 1), when the agent in S_8 moves right, it can access only state S_9 if $T(8,2,9) = 1$ and $T(8,2,k) = 0$ for $k \neq 9$. For stochastic dynamics, $T(i, j, k) < 1 \; \forall \; k \in \{1, \dots, N\}$. For example, if $T(8,2,3) = 0.1$, $T(8,2,7) = 0.2$,

$T(8,2,9) = 0.6$, and $T(8,2,13) = 0.1$, the agent can move to any of the four neighboring states, S_3, S_7, S_9 or S_{13}, in a Monte Carlo step. For deterministic MDPs, the RL agent will follow the same optimal sequence or path from a starting state to the goal state. In a stochastic MDP, the agent is allowed to take sub-optimal paths to the goal state.

2.2. Inverse reinforcement learning

In the inverse RL (IRL) problem, one can observe a partial sequence of state-action-state transitions of an agent, not knowing the agent's intended goal state and the structure of the reward function. The objective of an IRL algorithm is therefore to compute the rewards for a selected goal state from historically observed expert trajectories (Adams et al., 2022; Arora and Doshi, 2021; Ng and Russell, 2000; Ramachandran and Amir, 2007), and then to use these rewards to infer the probability that a future agent will pursue the same goal state given its partially observed path. Therefore, the inputs to the IRL problem are the state space, action space, transition probabilities, and the goal-specified expert trajectories. An IRL algorithm computes the rewards by predicting and matching the state, state-action, or state-action-state feature expectations observed in the expert trajectories. If the reward function is known, the intended goal state can be inferred from a given partial trajectory using Bayesian formulations (Snoswell et al., 2020, Ziebart et al., 2008).

2.3. Bayesian goal inference from a partial trajectory of states

The problem of Bayesian goal inference is to predict the probability that an agent will reach a destination state in a MDP environment, given a partial trajectory of states. Given a partial path from state A to state B, denoted as $\tau_{A:B}$, the probability of reaching a goal state G is given by the Bayes formula:

$$P(G|\tau_{A:B}) = \frac{P(\tau_{A:B}|G) \cdot P(G)}{P(\tau_{A:B})} \propto \frac{P(B \to G)}{P(A \to G)} \cdot P(G|A), \qquad (2)$$

Here, $P(G|A)$ is the prior probability of reaching goal state G along all observed paths from A in the set of expert trajectories; $P(A \to G)$ and $P(B \to G)$ are the probabilities of reaching goal state G along all possible paths from states A and B, respectively.

Most approaches for route preference and driver destination prediction use historical trip trajectories and Bayes rule to compute the probability that a location is the destination while a trip is in progression (Krumm and Horvitz, 2006, Snoswell et al., 2020, Xue et al., 2015, Ziebart et al., 2008). The first step in all these approaches is to segment the trips and map them on a 2-dimensional grid representation of the geographical area where the trips were observed. These methods differ in the way the likelihood and prior probabilities are computed.

Ziebart et al. (2008) developed an IRL algorithm based on maximum entropy (MaxEnt) principles to model driver behavior for route preference and destination prediction. The inverse RL algorithm solves for the unknown rewards to match the observed feature expectations of states, state-action pairs, or state-action-state triples in the expert trajectories. The prior destination and probabilities of the likelihood function are then calculated based on the state-action-state transition probabilities and IRL rewards. Specifically, the probability from a state, e.g., state A, to a goal state G is computed using the product of transition probability and an exponential function of the reward along each state-action-state transition and summing the products over the transitions in all possible paths from state A to G. The prior probabilities are computed in the same way but only over all previously observed paths of drivers from state A to G.

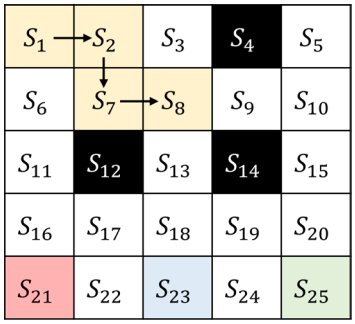

Figure 2. A 5 x 5 grid world environment illustrating a partial sequence of transitions (yellow-shaded cells) taken by a RL agent from state S_1 to state S_8 whose intended goal state must be determined from among three possible goal states, $\{S_{21}, S_{23}, S_{25}\}$, using IRL-based Bayesian formulation. Black-shaded cells indicate inaccessible states.

For example, consider the partial sequence of transitions taken by a RL agent from state S_1 to state S_8 in the 5 x 5 grid world environment, shown in Fig. 2. We want to determine which of the three goal states, S_{21}, S_{23}, and S_{24}, is the agent pursuing. To solve this in the IRL-Bayesian

formulation, we need to compute the IRL reward functions and the corresponding reward-based policies π_i for each goal state, G_i, using expert trajectories that terminate at each goal state. The posterior probability is computed using the formula:

$$P_{\pi_i}(G_i|\tau_{A:B}) = \frac{P_{\pi_i}(\tau_{A:B}|G_i)P_{\pi_i}(G_i|A)}{\sum_j P_{\pi_j}(\tau_{A:B}|G_j)P_{\pi_i}(G_j|A)} \quad (3)$$

where A and B correspond to states S_1 and S_8, respectively; G_i and $G_j \in \{S_{21}, S_{23}, S_{25}\}$. The most likely goal state is then the state with the maximum posterior probability.

In this work, we used the exact maximum entropy (ExactMaxEnt) IRL algorithm developed by Snoswell et al. (2020) to learn the rewards, instead of the original MaxEnt IRL algorithm developed by Ziebart et al. (2008) on which it was based on, because the ExactMaxEnt algorithm can be applied with variable length trajectories and various combinations of reward types ($R_1(s)$, $R_2(s,a)$, and $R_3(s,a,s')$). It also improves the reward learning by computing marginal probabilities exactly.

3. Problem formulation and computational approach

Here, we present the problem of predicting the destination of authors from partial trajectories of their state transitions in their research topic vector space. The problem is formulated starting with a set of research publications of authors working in a nuclear domain. We convert each author's sequence of publications in time into a sequence of state transitions on a rectangular grid that is defined by the research topic vector space of the publications. We then model the sequence of state transitions as a first-order Markov decision process (MDP) and use IRL to compute rewards that capture technology (goal state)-directed behavior of a group of authors. Finally, we develop and use a Bayesian formulation to compute the probability that a state in the topic space is the intended goal state, given a partial sequence of state transitions of an author.

3.1 Case study

To develop and test our approach, we selected papers based on a well-documented civil nuclear activity. In this work, we considered the construction of the Open Pool Australian Lightwater (OPAL) reactor in Australia (Olsen et al., 2008), as our case study application for the approach. The OPAL reactor is a 20-MW multi-purpose reactor, used for producing radioisotopes for cancer detection and treatment, and neutron beams for fundamental materials research (ansto.gov). It went critical in August 2006 and was officially opened in 2007. The goal inference problem is to infer the development of OPAL reactor activity from temporal sequences of publications of authors conducting nuclear research.

3.2. Approach

The computational approach involves the use of topic modeling, reward learning, and Bayesian inference methods to solve the problem of predicting technology-directed publication behavior of authors from partial trajectories of their publication sequences. The approach consists of the following ten steps:

Step 1. Find a paper or an initial set of papers associated with a technology or research activity. We call these papers the "coin" papers.

Step 2. Identify the authors of these coin papers and create a primary set of all papers written by these authors, which also includes the coin papers. We refer to these authors as the coin authors.

Step 3. Create a secondary set of papers published by co-authors of all the papers in the primary set.

Step 4. Combine the primary and secondary set of papers into one dataset.

Step 5. Extract the title, abstract, author information, and publication date of all the papers in the dataset.

Step 6. Perform topic modeling using the titles and abstracts of all the papers to define the research topic weight vector space.

Step 7. Construct a state-action-state transition graph to represent the state transitions of all author trajectories.

Step 8. Select the goal state associated with the technology of interest and the corresponding trajectory set for IRL reward learning.

Step 9. Compute state and state-action rewards based on the trajectory set, using the IRL algorithm.

Step 10. Calculate goal probability given a partial trajectory with a Bayesian formulation.

3.2.1. Creating the dataset (Steps 1 to 4)

The selection of the papers depends on the case study in hand. For our case study, we first

identified a flagship publication associated with OPAL reactor development or application, by searching Scopus for papers with keywords "Opal" and "reactor", found in the title, abstract, and/or keyword fields. Since the OPAL reactor went critical in August 2006, search results from 2004 through 2008 were considered. We considered a two-year time lag between the inception of a research activity and its publication. Hence a two-year buffer period for the search was applied after the year the OPAL reactor become operational. The chosen paper was written by Olsen et al. (2008). Here onwards, we shall call this flagship paper as the "coin" paper and the authors of this paper as coin authors (Step 1). The paper was written by nine authors, of whom five had Scopus ID's associated with a previous publication history.

After identifying the coin paper, we searched Scopus for papers written by the coin authors by their Scopus IDs (Step 2). Based on the search, we created a primary set of 278 coin-authored papers, including the coin paper. We then created a secondary set of papers that were written by all the non-coin authors of papers in the primary set, and this set contained 28,918 Scopus records (Step 3). Thus, a total of 29,196 Scopus records (spanning over the years from 1950 to 2008) were used to define the topic weight vector space of the OPAL MDP environment.

3.2.2. Defining the topic weight vector space (Steps 5 and 6)

We defined the topic weight vector space by first identifying an optimal number of *K* research topics to characterize the information extracted from the titles and abstracts of the 29,196 Scopus records. We used the Non-negative Matrix Factorization (NMF) algorithm, as implemented in the Scikit-learn Python package (Pedregosa et al., 2011), to obtain a list of topics and weights associated with the K topics for each paper.

To extract the features (relevant words) from the abstracts and titles, we used the *TfidfVectorizer* function provided by Scikit-learn Python package. English stop words, words that occurred only in one record, and words that occurred in over 95% of the records were removed during feature extraction. To fit the NMF model and to compute the weights of the *K* topics per record, we used the NMF function from *sklearn.decomposition*.

In addition to finding the optimal number of topics, we also determined the optimal combination of settings for three hyperparameters in the NMF analysis: *alpha*, *solver*, and *initialization*. This required running the NMF analyses for all possible hyperparameter combinations across a range of feature counts and number of components (topics) per feature count. The possible options for each hyperparameter were {'0.02', '0.1', '0.5'} for *alpha*, {'nndsvd', 'random,' 'nndsvda'} for *initialization*, and {'cd', 'mu'} for *solver*. It is noted that the 'mu' solver does not use 'nndsvd' initialization. Therefore, a total of 15 combination of these hyperparameter values were explored. Due to the longer run times associated with the full paper set, we assumed the optimal hyperparameters for the primary set to be optimal or near optimal for the full paper set. Eight papers that were written in 2008 (about the same time as the coin paper) were omitted from the initial NMF training set to reduce the likelihood of papers with similar topics as that of the coin paper. Thus, the hyperparameter exploration was limited to 270 papers in the primary set. Specifically, for each feature count and hyperparameter combination, we ran the NMF analysis and calculated the residual errors for all number of components (topics) ranging from 1 to the number of features. Using the Kneedle algorithm (Satopaa et al., 2011) from the *kneed* python package, we then determined the optimal number of components, which corresponded to the point of maximum curvature ("knee") in the error curve described by the residual error versus the number of components. For each hyperparameter combination, we then took the geometric mean of the optimal number of components across the sample features counts and compared these means to select the hyperparameter combination of *solver*, *initialization*, and *alpha* parameters that resulted in the lowest geometric mean value. The geometric mean value is the nth root of the product of number of component values across all sample feature counts per hyperparameter combination. The resulting hyperparameter values were '0.02' for *alpha*, 'cd' for *solver*, and 'random' for *initialization*. To avoid model overfitting, we applied the NMF function with *Frobenius* norm minimization and regularization, where the *L1 to L2* ratio was set to 0, and *alpha*, the constant multiplying the regularization term was set to 0.02.

Keeping the above hyperparameter settings, we then examined a more limited range of feature counts (up to 140) for the full set of 29,196 papers and determined the optimal number of components (*K* topics) following the same steps above. Our analysis indicated that seven topics ($K = 7$) with a feature count of 50 was optimal. In

all the NMF calculations, the maximum number of iterations was set to 20,000 and the stop condition tolerance was 0.0001.

3.2.3. Constructing the state-action-state transition graph (Step 7)

Based on the publication dates of all the 29,196 Scopus records, we created a temporal sequence of publications for each author, and then converted it to a sequence of state-action-state transitions, which were combined afterwards to form a state-action-state transition graph. The construction of a state-action-state transition graph involves defining the state space, action space, and state-action-state transition probabilities of the author topic grid MDP environment. The nodes and edges of the graph represent the states and actions of the author topic grid MDP, respectively. The transition probabilities were specified along each edge based on the number of authors who traversed the edge as per the data. The resulting state-action-state transition graph is a directed multi-edge graph, since more than one action (edge) is possible from one state to the other.

State space: We defined the states as the cells of a *K*-dimensional rectangular grid that represented the topic vector space of all the publications in the dataset. As described in the previous section, the topic weight vectors represent the topic vector space of the papers. Let $\boldsymbol{w} = (w_1, w_2, \cdots, w_K)$ denote a topic weight vector of a publication record in the K-dimensional topic vector space, such that $\sum_{i=1}^{K} w_i = 1$. We represent the topic vector space as a *K*-dimensional rectangular grid, where each axis represents the range of weight values from 0 to 1 for each research topic. The grid space may be discretized using uniform or non-uniform grid intervals. We used a partially uniform grid where each grid dimension is divided into m intervals, such that the width of the first and the last interval is half the width of the intermediate intervals whose widths were $1/(m-1)$. The half-width interval was used to separately group publications with topic weights close to zero and one. Each publication can be mapped to a cell on the topic grid based on their topic weight values. For example, Fig. 3 illustrates a map of 20 publications on a 2-dimensional topic grid ($K = 2$), characterized by topic vector weights w_1 and w_2. Each axis is divided into 5 intervals ($m = 5$), where the half-width interval is 1/8. This grid contains 25 grid cells (states), which can be numbered from 1 to 25 and these numbers are used to identify the states. The grid cell in which a publication is located is considered as the state (active research state) of an author of the publication. The number of grid cells (states) in the author topic MDP environment will depend on the choice of the grid cell spacing along each grid dimension. But not all grid cells will be occupied with a publication. For IRL reward learning, we considered only those grid cells with a minimum record occupancy of one to constitute the state space of the author topic grid MDP environment. Thus, the number of states in the state space is the number of grid cells with a minimum record occupancy of one. If N is the number of states, the state space of the MDP is represented as $S = \{S_i | i = 1,2, \ldots, N\}$.

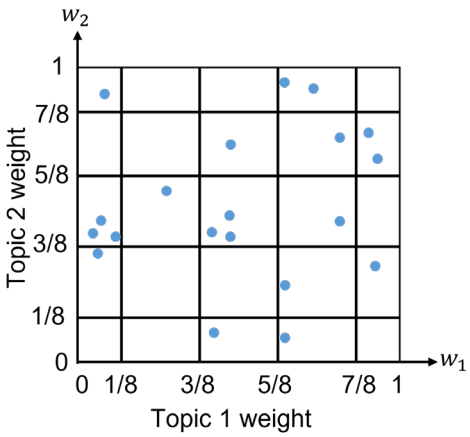

Figure 3. A 5 x 5 topic grid world environment illustrating the mapping of 20 research publications on a two-dimensional grid with 1/8 grid spacing for weight values near zero and one, and 2/8 spacing for the others.

Action space: We defined the actions as the difference in the number of years it took for authors to move from one state (grid cell) to another in the topic MDP grid environment. For example, if an author has a paper published in year t_1 and is in state S_1, and the author's next publication is in year t_2 and in state S_2, the action taken by the author to move from state S_1 to S_2 is calculated as $t = t_2 - t_1$. The graph will have a directed edge for the action t drawn from state S_1 to S_2. If the year difference was 0, we used the month and day information from the publication dates to determine the direction of the edge along each state transition. Actions that do not cause a change in the state of the author will result in self-loops, which are ignored in the state-action-state transition graph. If M is the number of states, the action space of the MDP is represented as $A = \{A_j | j = 1,2, \ldots, M\}$.

Differences in state transitions between a standard grid world walk and the author topic grid world walk: In the RL framework, it is important to understand how the RL agent will move (walk) from one state to the other.

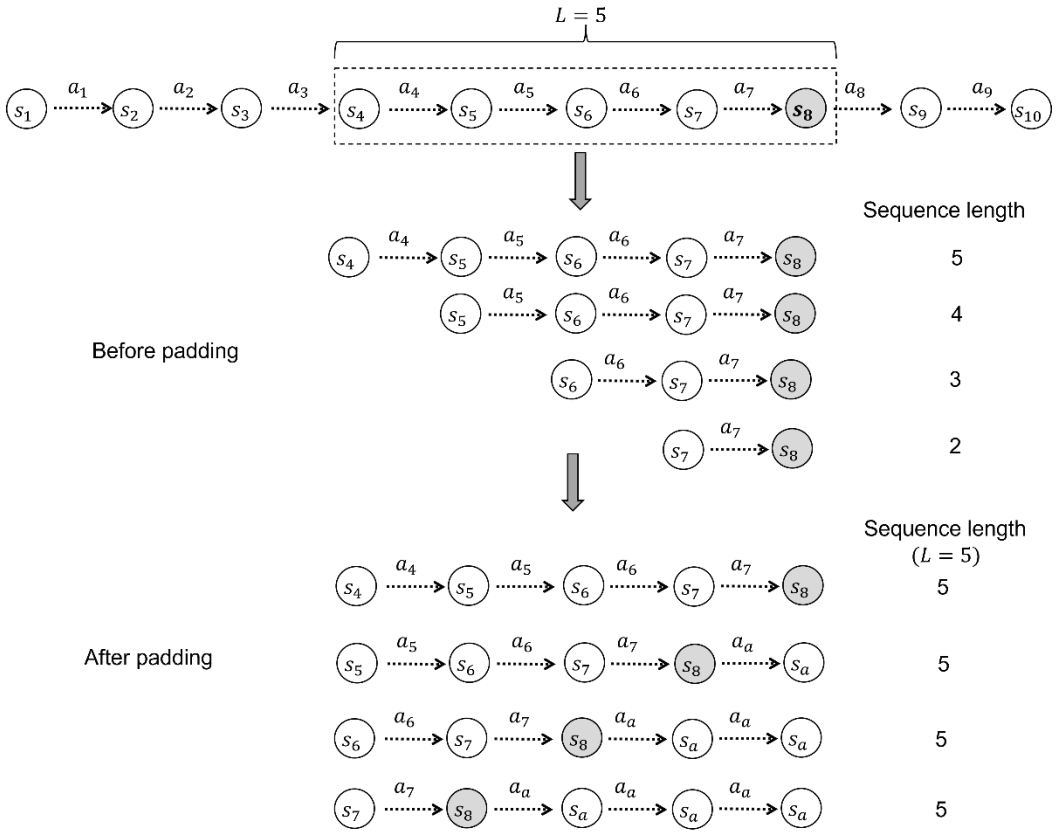

Figure 4. An illustration of how sub-trajectories of a specified length $L = 5$, with goal state s_8, are derived from an author trajectory with 10 states, before and after padding with auxiliary state s_a and action a_a.

Compared to a geo-spatially constrained grid world MDP walk, such as the 5 x 5 grid MDP walk shown in Fig. 1, the author MDP grid walk is not limited to adjacent state transitions in the topic grid space. There are no self-loops in the grid world walk because the agent needs to move out of a state to come back to that state. On the other hand, self-loops can exist in the author topic grid walk because an author can publish an article that falls in the same state as the previous one in time. In the grid world walk, an agent cannot jump over states in its path. In the author topic grid walk, authors can jump from one state to every other state. In the grid world walk, all the accessible states from each state are known. In the author topic grid walk, however, not all accessible states from each state are known because new state transitions can occur in the future, although they do not exist at the current point in time. Although, in theory, an RL agent can move from one state to every other state in the author topic grid walk, in our work, we limit the RL agent movements only to state transitions known from all the author trajectories in the dataset.

Transition probability: The transition probability, denoted as $T(s'|s,a)$, along each (s,a,s') edge was computed by dividing the number of authors who moved from state s to state s' through action a by the total number of authors who moved to all accessible states from s through a.

3.2.4. Trajectory set for IRL reward learning (Step 8)

The IRL reward learning begins by selecting a goal state, followed by a set of expert trajectories $\mathcal{T}$ that terminate at the goal state. In the author topic MDP grid space, authors can take different paths to a goal state, from different starting states. This will result in trajectories of unequal lengths in the set $\mathcal{T}$, where the length is defined as the number of states from a starting state to the goal state in a trajectory. Additionally, sub-trajectories can be realized as starting from states away from the goal state in varied number of steps, in an author trajectory. To form the trajectory set $\mathcal{T}$ for reward learning, we first select a sub-trajectory of a specified length L, which terminates at the goal state in each author trajectory. Each sub-trajectory and their subsequent ones are reduced in length by one, after removing the first state until the length is two. This process results in a set of trajectories of length varying from 2 to L. To illustrate this process, let's consider an author trajectory with 10 states as shown in Fig. 4, where s_8 is

indicated as the goal state. If $L = 5$, then four sub-trajectories can be derived from this trajectory, as shown in Fig. 4. These trajectories vary in length from 2 to 5 states. Thus, the IRL training set used for the reward calculations will contain trajectories of variable lengths from 2 to L, which are obtained from all author trajectories that contain the goal state. To make the reward learning efficient, we make all the trajectories in the IRL training set to be of the same length, L, using the padding trick described by Snoswell et al. (2020). Specifically, we pad trajectories shorter than the longest trajectory, which is of length L in the set, with auxiliary state-action sequences, $\{(\cdot, a_a), (s_a, \cdot)\}$. An example of padding is shown in Fig. 4, where all the trajectories are of length 5 after padding. It should be noted that in our implementation of the IRL reward learning algorithm, we represented each sub-trajectory of length L as a sequence of state-action transitions as $((s_1, a_1), (s_2, a_2), \ldots, (s_t, a_t), \ldots, (s_L, a_a))$ where an author starts from state s_1 and ends at the terminal state s_L in $L - 1$ action steps. The transition (s_t, a_t) refers to the state $s_t \in S$ and action $a_t \in A$ at step t. The last action step, (s_L, a_a), is an added step for the transition from the terminal state s_L to the auxiliary state s_a through the auxiliary action a_a (not shown in Fig. 4). In this work, we set the L value to the length of the shortest coin author trajectory.

3.2.5. Reward learning (Step 9)

We used the ExactMaxEnt IRL algorithm (Snoswell et al., 2020) to compute the rewards of the author topic grid MDP and implemented it in Python. The algorithm can be used to compute three types of rewards: state rewards $R_1(s)$, state-action rewards $R_2(s, a)$, or state-action-state rewards $R_3(s, a, s')$. Each reward type is defined as a linear function of the respective state, state-action, or state-action-state feature vectors, with reward weight vectors $\boldsymbol{\theta_1}$, $\boldsymbol{\theta_2}$, and $\boldsymbol{\theta_3}$, respectively. The objective of the algorithm is then to fit the reward weights to match the discounted feature expectations in the set ($\mathcal{T}$) of observed trajectories that terminate at the goal state. For the author topic MDP, we used the state and state-action rewards to model the goal-directed behavior of the authors, and they were calculated as follows.

Let n_1 and n_2 represent the number of state and state-action features of the author topic MDP, respectively. The state reward function is written as $R_1(s) = \boldsymbol{\theta}_1^{\mathrm{T}} \boldsymbol{f_1}(s)$, where $\boldsymbol{f_1}(s)$ is the feature vector of state $s \in S$ of size $n_1 \times 1$, and $\boldsymbol{\theta_1}$ is the state reward weight vector of size $n_1 \times 1$. The state-action reward function is written as $R_2(s, a) = \boldsymbol{\theta}_2^{\mathrm{T}} \boldsymbol{f_2}(s, a)$, where $\boldsymbol{f_2}(s, a)$ is the feature vector of the state-action pair $(s, a) \in (S, A)$ of size $n_2 \times 1$, and $\boldsymbol{\theta_2}$ is the state-action reward weight vector of size $n_2 \times 1$.

The actual state and state-action features of the author topic MDP are not known, and such information may not be readily available, or may be difficult to obtain or learn from data. Therefore, we considered the rewards to be independent of these features by setting the state and state-action feature vectors as unit vectors of length $n_1 = N$ and $n_2 = NM$, respectively. For example, the state-feature vector for state S_i is specified as $\boldsymbol{f_1}(s = S_i) = (b_k)_{k=1}^{N}$ such that $b_k = 1$, if $k = i$ and $b_k = 0$, otherwise. Similarly, the state-action feature vector for all state-action pairs (S_i, A_j) is specified as $\boldsymbol{f_2}(s = S_i, a = A_j) = (c_k)_{k=1}^{MN}$ such that $c_k = 1$, if $k = (i-1)M + j$ and $c_k = 0$, otherwise. This makes the reward weights equivalent to the respective rewards. In our work, we fitted the reward weights (rewards) to predict and match the average state and state-action visitation frequencies in the set of expert trajectories, as described below.

If there are n_T trajectories in the expert trajectory set $\mathcal{T}$, and each trajectory τ_k is a sequence of L states, represented as $(s_{k,t}, a_{k,t})_{t=1}^{L}$, where $s_{k,t} \in S$ and $a_{k,t} \in A$, then the average state visitation frequency for each state in S is calculated as

$$\overline{\boldsymbol{f_1}} = \frac{1}{n_T} \sum_{k=1}^{n_T} \sum_{t=1}^{L} \boldsymbol{f_1}(s_{k,t}), \tag{4}$$

and the average state-action visitation frequency for each state-action pair in $\{(S_i, A_j) | S_i \in S, A_j \in A\}$ is calculated as

$$\overline{\boldsymbol{f_2}} = \frac{1}{n_T} \sum_{k=1}^{n_T} \sum_{t=1}^{L-1} \boldsymbol{f_2}(s_{k,t}, a_{k,t}). \tag{5}$$

In addition to the state and action spaces, S and A, we also include an auxiliary state space $\{S_a\}$ and an auxiliary action space $\{A_a\}$. The combined state and auxiliary state spaces is denoted as $S^* = S \cup \{S_a\} = \{S_j | j = 1, \ldots, N+1\}$, where $S_{N+1} = S_a$. The combined action and auxiliary action spaces is denoted as $A^* = A \cup \{A_a\} = \{A_j | j = 1, \ldots, M+1\}$, where $A_{M+1} = A_a$.

In the trajectories, the auxiliary state is treated as a self-absorbing state and is accessible from all states only through the auxiliary action. Thus, we set the transition probability for all state-action-state transitions involving the auxiliary state and action as follows:

$$T(s_a|s,a) = \begin{cases} 1, \forall s \in S, a = a_a \\ 0, \forall s \in S, a \neq a_a \\ 1, s = s_a, a = a_a \\ 0, s = s_a, a \in A \end{cases}. \quad (6)$$

To remove the effect of the auxiliary state and action on the rewards accumulated along a trajectory, we set $R_1(s_a) = 0$, $R_2(s_a, a_a) = 0$, and $R_2(s_a, a) = 0 \; \forall \; a \in A$.

$$\alpha_{t+1}(s') = \sum_{s \in S^*, a \in A^*} \alpha_t(s) \, T(s'|s,a) \exp\left(\gamma^t R_1(s')\right), \text{ for } 1 \leq t < L, \quad (10)$$

$$\beta_1(s) = \exp\left(\gamma^{L-1} R_1(s)\right), \text{and} \quad (11)$$

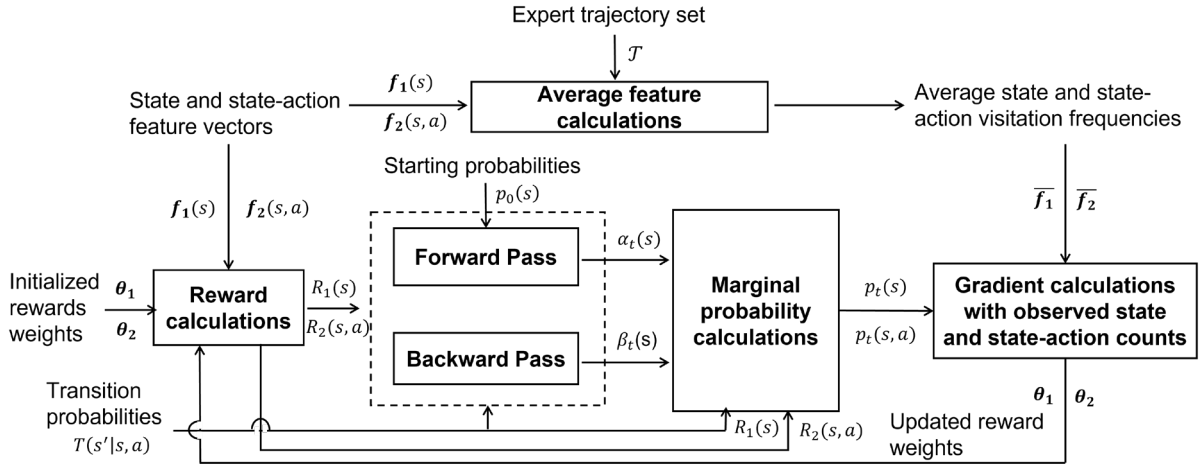

Figure 5. Steps of the reward learning process based on the ExactMaxEnt IRL for the author topic MDP.

To compute the rewards, we first initialized the reward weights with uniformly distributed random values and updated them until convergence. Fig. 5 shows the basic steps involved in a single iteration of the reward learning process based on the ExactMaxEnt IRL algorithm (Snoswell et al., 2020). The discount factor, $\gamma = 0.99$ in all the calculations. We define the frequency of times a state $s \in S$ occurred as the starting state in the expert trajectory set, as its starting probability, which is denoted as $p_0(s)$. We then use the forward-backward algorithm for a first-order MDP to efficiently compute the state marginal probability, $p_t(s)$, that an author will visit each state s at step t, and the state-action marginal probability, $p_t(s,a)$, that the author perform an action a at step t, from all observed starting states, respectively. The state marginal probability, $p_t(s)$, is calculated for $t = 1$ to $L - 1$ as,

$$p_t(s) = \frac{\alpha_t(s)}{Z} \sum_{a \in A^*, s' \in S^*} T(s'|s,a) e^{\gamma^{t-1} R_2(s,a)} \beta_{L-t}(s'), \quad (7)$$

and for $t = L$ as,

$$p_L(s) = \frac{\alpha_L(s)}{Z}, \quad (8)$$

where $\alpha_t(s)$ and $\beta_t(s)$ are the forward and backward message variables, defined as

$$\alpha_1(s) = p_0(s) \exp\left(R_1(s)\right) \quad (9)$$

$$\beta_{t+1}(s) = \sum_{s' \in S^*, a \in A^*} T(s'|s,a) \exp\left(\gamma^{L-t-1} R_1(s)\right) \beta_t(s'), \text{ for } 1 \leq t < L. \quad (12)$$

The partition function, Z, is calculated as

$$Z = \sum_{t=1}^{L} \sum_{s \in S} \alpha_t(s). \quad (13)$$

The state-action marginal probability, $p_t(s,a)$ is calculated as

$$p_t(s,a) = \frac{\alpha_t(s)}{Z} \sum_{s' \in S^*} T(s'|s,a) \exp\left(\gamma^{t-1} R_2(s,a)\right) \beta_{L-t}(s'), \text{ for } t = 1 \text{ to } L-1. \quad (14)$$

We update the state and state-action reward weights using a gradient descent approach based on the difference between the observed and IRL-predicted values for both the state as well as the state-action visitation frequencies, respectively. The state reward weight is updated as,

$$\boldsymbol{\theta_1} \coloneqq \boldsymbol{\theta_1} + \varepsilon_1 \nabla \boldsymbol{\theta_1}, \quad (15)$$

where ε_1 is the learning rate and the state reward gradient $\nabla \boldsymbol{\theta_1}$ is calculated as

$$\nabla \boldsymbol{\theta_1} = \overline{\boldsymbol{f_1}} - \sum_{s \in S} \boldsymbol{f_1}(s) \sum_{t=1}^{L} p_t(s). \quad (16)$$

The state-action reward weight is updated as,

$$\boldsymbol{\theta_2} \coloneqq \boldsymbol{\theta_2} + \varepsilon_2 \boldsymbol{\nabla \theta_2}, \quad (17)$$

where ε_2 is the learning rate and the state-action reward gradient $\boldsymbol{\nabla \theta_2}$ is calculated as

$$\boldsymbol{\nabla \theta_2} = \overline{\boldsymbol{f_2}} - \sum_{s \in S, a \in A} \boldsymbol{f_2}(s,a) \sum_{t=1}^{L-1} p_t(s,a). \quad (18)$$

The reward learning steps (Fig. 5) were repeated 3000 times with learning rates equal to 0.05 for both ε_1 and ε_2. These settings were needed to obtain converged results. We assessed the convergence of the IRL simulations based on the root mean squared values of the reward gradients and the strength of correlation between the observed and predicted values of both the state and state-action visitation frequencies. The lower the reward gradients and the higher the correlation, the better the convergence.

3.2.6. Goal probability calculation (Step 10)

Goal probability calculations were performed to infer which state out a selected set of possible goal states, an author is most likely to publish in, given a partial trajectory of their observed state transitions in the author topic MDP grid space. We have developed a Bayesian formulation for computing the goal probabilities, like how the driver destination probabilities are calculated from partial trip trajectories (Ziebart et al., 2008). In the case of driver destination prediction, it is assumed that the driver's intent towards a destination occurs at the start of the trip. In the case of author trajectories, which are based on their sequences of publications in time, it cannot be known at what point (in time) along the trajectory, the author begins to have an intent to publish in a particular goal state. Therefore, in our Bayesian formulation, we considered the publication intent to begin at the previous state of each step along the trajectory, and computed the goal probability as follows.

Let's consider a set of N_g goal states, denoted as $G = \{G_i | i = 1{:}N_g\}$, where $G \subset S$. One of the goal states is the coin state, which is the state where the coin paper is located. The objective of the goal inference is to determine which of the N_g goal states is an author pursuing, given the observed steps (or states) of a partial trajectory. In other words, which reward policy (behavior) is the author following? Let π_i denote the RL agent's reward policy based on the rewards calculated for each goal state $G_i \in G$.

Let's consider a partial trajectory of t state transitions, $(s_{k-1}, s_k)_{k=1}^{t}$. For each step s_{k-1} to s_k in the partial trajectory and for each reward policy π_i, we calculate the posterior probability of reaching each state $S_j \in S$ of the MDP in $L-1$ steps, using the formula,

$$P_{\pi_i}(S_j | s_{k-1} \to s_k) = \frac{P_{\pi_i}(s_{k-1} \to s_k | S_j) P_{\pi_i}(S_j | s_{k-1})}{\sum_j P_{\pi_i}(s_{k-1} \to s_k | S_j) P_{\pi_i}(S_j | s_{k-1})}. \quad (19)$$

Here, the likelihood,

$$P_{\pi_i}(s_{k-1} \to s_k | S_j) = \frac{P_{\pi_i}(s_k \to S_j)}{P_{\pi_i}(s_{k-1} \to S_j)}, \quad (20)$$

is the probability that the agent with policy π_i will move from state s_{k-1} to state s_k, if the agent's intended (desired) goal state is S_j. $P_{\pi_i}(S_j | s_{k-1})$ is the prior probability based on all paths of length $\leq L$ observed in the expert trajectory set from state s_{k-1} to S_j. This is computed with a starting probability of 1 for the state s_{k-1}. $P_{\pi_i}(s_{k-1} \to S_j)$ and $P_{\pi_i}(s_k \to S_j)$ are the total path probabilities of reaching the state S_j within 1 to $L-1$ steps through all possible paths in the author topic MDP from s_{k-1} and s_k, respectively. The total path probabilities to the S_j state from each state of the partial trajectory were computed using the forward pass algorithm, where the starting probability of each state s_k was set as

$$p_0(s_k) = \begin{cases} 1, & \text{if } k = 1 \\ P_{\pi_i}(s_{k-1} \to s_k), & \text{if } k > 1 \end{cases}. \quad (21)$$

To infer the reward policy (or the associated goal state) of an author along each step s_{k-1} to s_k of the author's partial trajectory, we compute the probability of reaching each of the N_g goal states, $G_i \in G$, in $L-1$ steps using the reward policy π_i, as,

$$P_{\pi_i}(G_i | s_{k-1} \to s_k) = \frac{P_{\pi_i}(s_{k-1} \to s_k | G_i) P_{\pi_i}(G_i | s_{k-1})}{\sum_j P_{\pi_j}(s_{k-1} \to s_k | G_j) P_{\pi_i}(G_j | s_{k-1})}, \quad (22)$$

where, the likelihood is calculated using Eq. (20), and $P_{\pi_i}(G_i | s_{k-1})$ is the prior probability based on all paths of length $\leq L$ observed in the expert trajectory set from state s_{k-1} to G_i.

Finally, the goal probability of reaching the goal state G_i by the agent with policy π_i based on observing t state transitions is given by the normalized cumulative sum of the probabilities $P_{\pi_i}(G_i | s_{k-1} \to s_k)$ from each step (s_{k-1}, s_k), where $k = 1, 2, \ldots, t$, and is written as,

$$P_{\pi_i}(G_i | (s_{k-1}, s_k)_{k=1}^{t}) = \frac{\sum_1^t P_{\pi_i}(G_i | s_{k-1} \to s_k)}{\sum_j \sum_1^t P_{\pi_j}(G_j | s_{k-1} \to s_k)}. \quad (23)$$

4. Results

We present the topic modeling, IRL and goal inference results for the OPAL case study application to demonstrate technology-directed goal inference of author behavior in the nuclear research topic MDP grid environment. For demonstrating the inference problem, we selected four goal states; one of them is the coin state where the OPAL coin paper is located.

- *Crystal structure and diffraction* (Topic 4),
- *Surface reaction-based analysis of materials* (Topic 5),
- *Properties of compounds* (Topic 6), and
- *Electron orbitals as the basis for crystal properties* (Topic 7).

Based on the topic weight distribution, a paper may cover multiple topics with varying weights, and one topic may be highly weighted than all

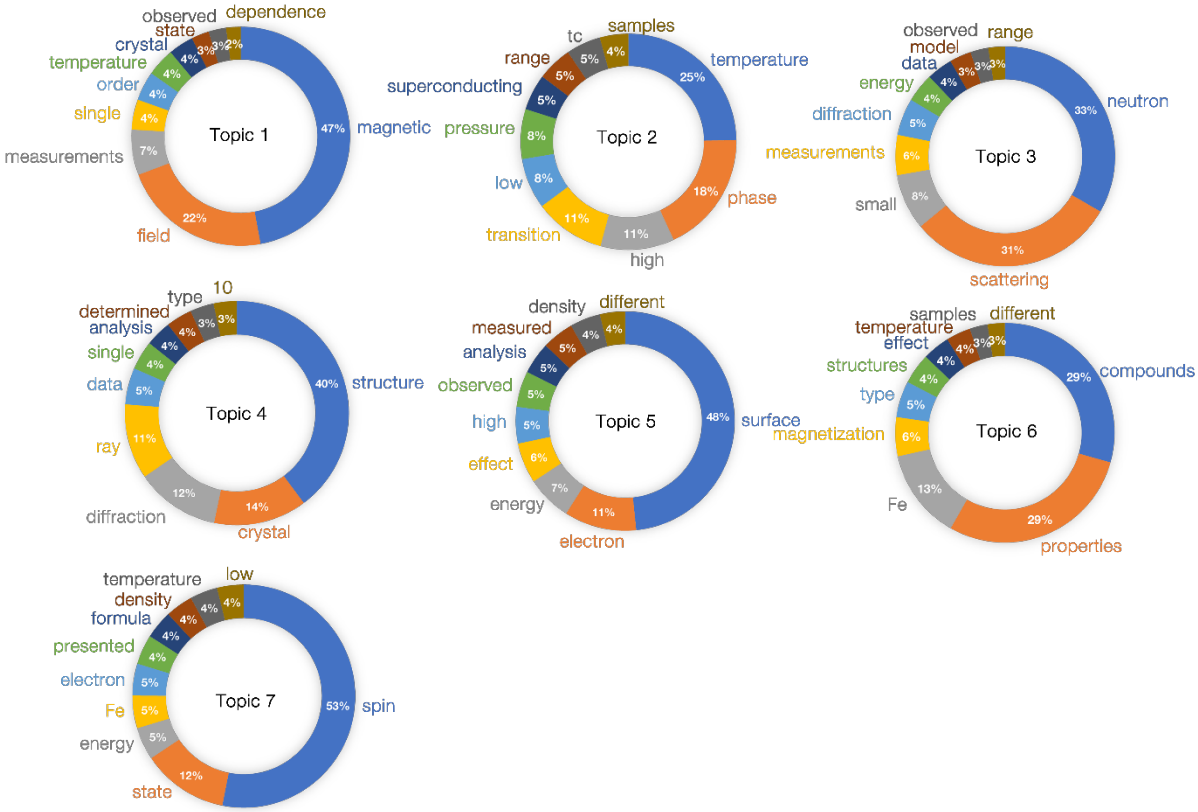

Figure 6. Normalized weight distribution of top ten keywords in each of the seven topics used for characterizing the topic vector space of 29,196 Scopus records.

4.1 Topic modeling

Based on NMF topic modeling, we defined the topic vector space of the 29,196 Scopus records in the OPAL case study application with seven topics. Fig. 6 presents the seven topics with normalized weight contributions of the top ten distinct keywords per topic, where the topics are numbered from 1 to 7. The keywords are distinct enough to provide a unique meaning to each topic. All the topics indicate some measurement study or analysis related to nuclear research. Based on these keywords, we interpret the topics as:

- *Magnetic field effects* (Topic 1),
- *Temperature-based phase transition* (Topic 2),
- *Small angle neutron scattering measurements and models* (Topic 3),

others. For example, the coin paper has two topics, Topics 1 and 3, with weights 0.27 and 0.73, respectively. The abstract of the coin paper mentions the use of OPAL cold neutron source for small angle neutron scattering experiments (Topic 3) and the use of horizontal field HTS magnet (Topic 1) in its cold neutron instruments. Approximately 22.24% of the records have a dominant topic weight greater than 0.7 (6494 out of 29196 records). Specifically for topics 1 to 7, this number was 577, 1818, 999, 1214, 845, 622, and 419, respectively.

4.2 OPAL state-action-state transition graph

We discretized the seven-dimensional topic vector space by dividing each dimension into six intervals with 0.1 width for the first and last intervals, and 0.2 for the intermediate intervals. Mapping the topic weights of the records on to

the grid resulted in 1,276 grid cells with occupancies ranging from 1 to 942 records, as shown in Fig. 7. About 75.6% of the grid cells (i.e., 965 out of 1,276) contained less than 26 records each, which accounted for 51.3% of the total records (i.e., 6,265 out of 12,196). The highest number of 942 records, which accounted for 3.2% of the total records, was found in the state with the lowest range of weight values across all seven topics in the topic grid space.

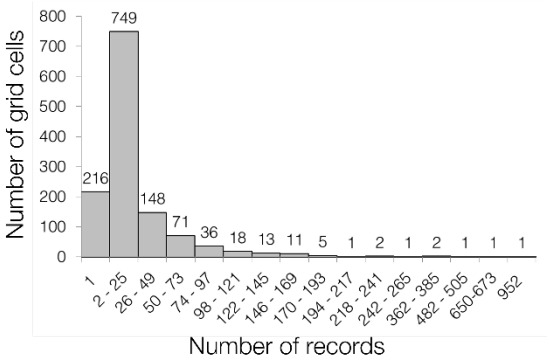

Figure 7. Frequency distribution of the number of grid cells occupied according to the number of records

The occupied grid cells were selected as the states for the author topic MDP. Our action definition resulted in 15 actions, where the maximum year difference observed between two sequential publications was 15 and the minimum was 0. A year difference of 13 was not observed in the data. The data contained 403 author publication sequences, out of which 402 were represented as state-action-state trajectories using the 1,276 states and 15 actions. The states were numbered from 1 to 1276 for identification. One author trajectory was ignored because it had only one state transition with a self-loop. The frequency of transitions in the 402 trajectories were used to compute the transition probabilities. The state-action transitions present in the author trajectories formed the OPAL state-action-state transition graph with 1276 nodes (states) and a maximum of 15 incoming/outgoing edges (actions) per node. Thus, the author topic MDP was represented as a stochastic MDP with 1,276 states, 15 actions, transitional probabilities, and a discount factor $\gamma = 0.99$. We note that the author trajectories are of different lengths and within a trajectory, an author can re-visit a state two or more times. For example, the length (number of states) of the five coin author trajectories were 17, 37, 58, 80, and 98. Fig. 8 shows the state transitions of one coin author based on their whole publication sequence, starting from their initial publication to the coin publication.

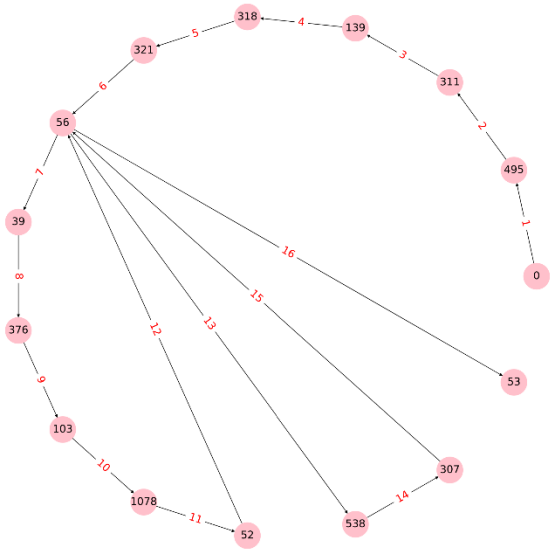

Figure 8. State transitions of a coin author based on their whole publication sequence in the dataset. The author begins in state 0 and ends in coin state 53 through 16 steps. The number along each arrow represents the step number.

4.3 IRL-based Bayesian goal inference

We computed the state and state-action rewards for four sets of author trajectories. These trajectories terminate at one of four different goal states, with one of the states being the coin state. We identify the four goal states by numbers, as 53, 132, 145, and 212, and denote the corresponding trajectory sets as $\mathcal{T}_{53}$, $\mathcal{T}_{132}$, $\mathcal{T}_{145}$, and $\mathcal{T}_{212}$, respectively. Goal state 53 is the coin state. Goal states 132, 145, and 212 were arbitrarily picked while ensuring their corresponding trajectory sets do not contain the coin state. The length of all the trajectories used in the IRL training set was set to $L = 17$, based on the length of the shortest coin-author trajectory. The coin state trajectory set $\mathcal{T}_{53}$ contained 45 author trajectories, out of which six were of the five coin authors. There were six instead of five coin-authored trajectories in the set because one coin author, whose full trajectory had 98 states, had visited the coin state twice and they were coincidentally 17 steps apart, resulting in two trajectories of length ≤ 17 for this author. For the IRL reward calculations, we excluded the six coin-author trajectories from $\mathcal{T}_{53}$ and used them as a test set for external validation. Thus, the number of $\mathcal{T}_{53}$ trajectories used for the reward learning was 39. The sets $\mathcal{T}_{132}$, $\mathcal{T}_{145}$, and $\mathcal{T}_{212}$ contained 14 trajectories, and all of them were used for reward learning.

The IRL simulations converged with root mean squared values below 0.003 for the reward gradients. The linear correlation coefficients between the observed and predicted values of

both the state as well as state-action visitation frequencies were approximately 0.99 for all four trajectory sets, which provided further validation for convergence after 3000 reward updates. Each IRL simulation took about 17 hours to complete on a Windows laptop with 32 GB RAM and 20 2.4-GHz processors.

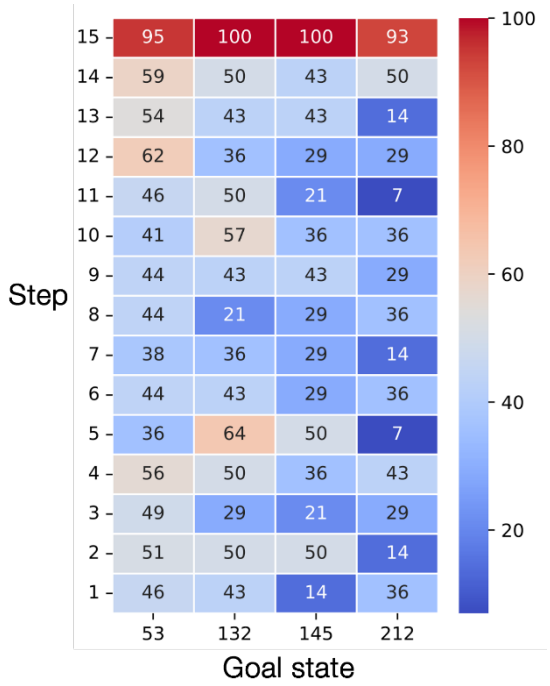

Figure 9. Percentage frequency of times (trajectories) each goal state (53, 132, 145, and 212) was ranked first at each step by their respective reward policy-based posterior probabilities. The last step, step 16, is not shown because the authors have already reached their goal state.

After learning the reward policies of the authors for the four goal states, we performed three types of validation for goal inference, on the trajectories that were used for learning the reward policies (the training sets). First, we evaluated whether the posterior probabilities that are computed based on each trajectory set's actual goal state reward policy (Eq. (19)) can be used to infer the goal state of the trajectories in the set. Specifically, we determined if the reward policy captures the correct goal-directed behavior of the authors by resulting in higher posterior probability values for the actual goal state than that for the other 1,275 states. In the second validation, we performed goal inference with all four reward policies on each trajectory set to determine if the posterior probabilities are lower for the reward policies that are not of the actual goal state. In the third validation, we computed the goal probabilities using Eq. (23), to determine if they are higher for the reward policy associated with the actual goal state of each trajectory set.

To further evaluate the performance of the learned reward policy for inferring the goal-directed behavior of coin authors, we performed a fourth validation by testing the goal inference on the six coin-author trajectories and comparing the goal probabilities based on each reward policy. These tests provided an external validation for the reward policies by testing on trajectories that are not in the training sets.

4.3.1. Goal inference via ranking all the MDP states by their posterior probabilities in each training trajectory set using the set's goal state reward policy

Using the reward policy learned from each trajectory set, we computed the posterior probability of reaching each of the 1,276 OPAL MDP states at every step of a trajectory in the set (using Eq. (19)). We then ranked the states in descending order of their posterior probability values. The state with the highest rank, i.e., the state with the maximum posterior probability, was considered the most probable goal state. After identifying the highest ranked state in the first 15 steps of each trajectory, we computed the frequency of times (percentage of trajectories) each state was ranked first in every step. Fig. 9 shows the frequency values of the goal states 53, 132, 145, and 212, when they ranked first at each step in their respective trajectory sets. In most steps, the frequency was the highest for the actual goal state of each trajectory set (although not fully at 100%), signifying that the reward policies are able to capture the correct goal-directed behavior of the authors in their respective training sets. For example, with reward policy π_{53}, the state 53 was inferred to be the most probable goal state at step 4 in 56% of the trajectories ($\mathcal{T}_{53}$) (i.e., approximately 22 out of 39 trajectories). For the set $\mathcal{T}_{212}$, at steps 5 and 11, the goal state was correctly inferred for only one out of the 14 trajectories in the set. In this case, the inferred goal state varied across all the 14 trajectories. For each set of trajectories, the rankings indicate that other states can be inferred as the goal state besides the actual goal state, but at lower frequencies. In total, there were 173, 56, 87, and 96 states inferred at least once as the goal state in the trajectory sets $\mathcal{T}_{53}$, $\mathcal{T}_{132}$, $\mathcal{T}_{145}$, and $\mathcal{T}_{212}$, respectively. It can also be seen that the frequencies are nearly 100% in the step before the trajectories terminate at their respective actual goal states.

Although the actual goal states were not ranked first all the time (i.e., in all steps) in their respective trajectory sets using the goal state's reward policy, they ranked among the top 5 states, in at

least 90% of the steps per trajectory, as shown in Fig. 10. The rankings varied from 1 to 13 for state 53, 1 to 12 for states 132 and 145, and 1 to 10 for state 212. From these results, we can generally expect that the posterior probabilities, computed using the actual goal state's reward policy (Eq. (19)), will rank the goal state among the top 5 (out of the 1,276) states in at least 90% of the steps observed (on average per trajectory) in the training sets. This is the best ranking and frequency profile that can be achieved if an author were to follow the correct reward policy in a goal inference test.

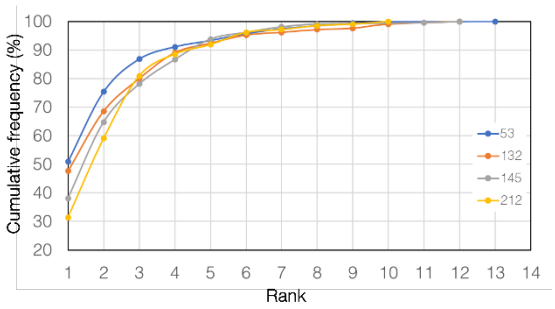

Figure 10. Cumulative frequency of inferred goal rankings for the states 53, 132, 145, and 212 based on their respective reward policy in trajectory sets $\mathcal{T}_{53}$(a), $\mathcal{T}_{132}$ (b), $\mathcal{T}_{145}$ (c), and $\mathcal{T}_{212}$ (d).

4.3.2. Goal inference via ranking the MDP states by their posterior probabilities in each training trajectory set using the reward policies of all goal states

The above goal inference tests were performed on each trajectory set using the reward policy learned for that set. That is, the goal state associated with the reward policy and with the trajectory set used for inference test are both the same. If the goal inference was performed with a reward policy other than that associated with the actual goal state of a trajectory set, then we should expect the ranking for the goal state associated with the reward policy to be lower. To verify this, we computed the posterior probabilities based on all four reward policies on all trajectory sets and determined the rankings (and their frequencies) for the goal state of each reward policy. As seen in Figs. 11 (a-d), the frequency of rankings among the top 1 to 10 states are generally 30 to 40 % lower for all goal states inferred with reward policies different from that of the actual goal state. A few exceptions are noted in Fig. 11(c) and (d), where the percentage of times state 53 ranked first was higher by 7.6% and 9% in trajectories whose actual goal states were 145 and 212, respectively. Overall, the ranking results in Fig. 11 verify that only the reward policy for the actual goal state captures the correct behavior of authors in each trajectory set. For example, as shown in Fig. 11(a), state 53 ranked among the

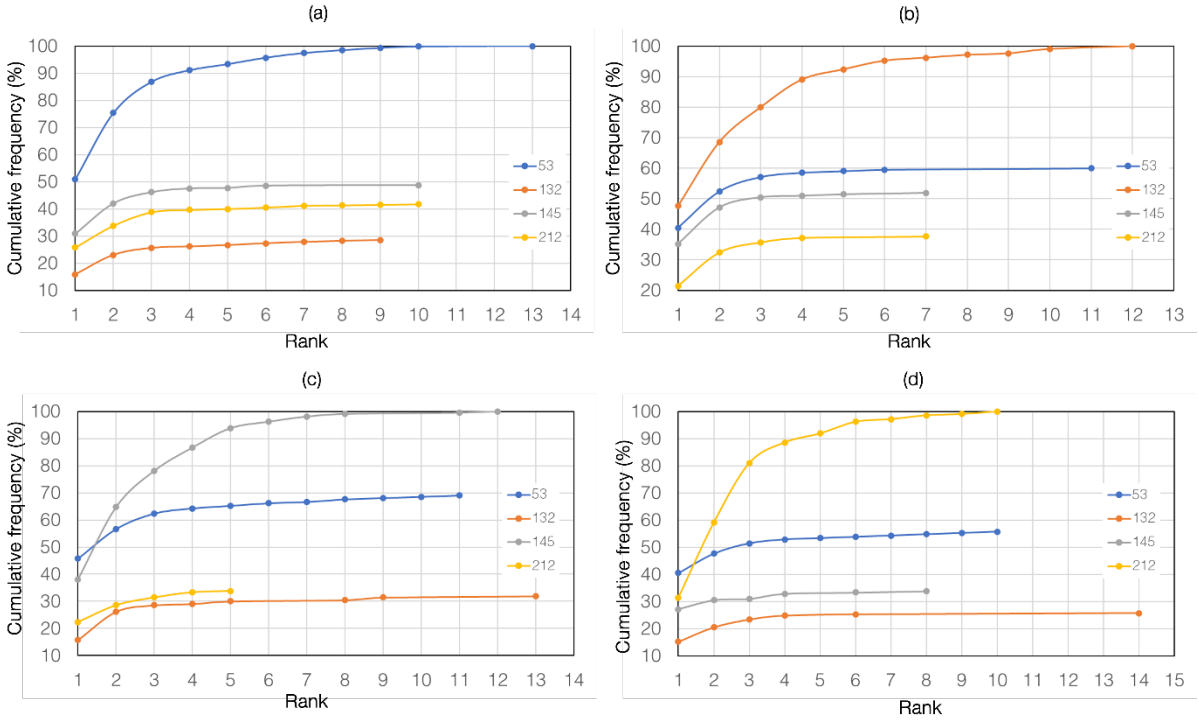

Figure 11. Cumulative frequency (%) of inferred goal rankings for the states 53, 132, 145, and 212 in trajectory sets $\mathcal{T}_{53}$(a), $\mathcal{T}_{132}$ (b), $\mathcal{T}_{145}$ (c), and $\mathcal{T}_{212}$ (d), respectively. For each trajectory set, ranking was done by comparing the posterior probabilities of all states computed based on the reward policy learned for the trajectory set's goal state.

top 10 states 100% of the time (steps per trajectory). State 132 ranked among the top 10 states in less than 30% of the time. States 145 and 212 ranked among the top 10 in less than 49% and 42% of the time, respectively. States 132, 145, and 212 ranked 394th, 367th, and 634th, respectively at other times (not shown). Thus the ranking results in Fig. 11 (a) verify that only the reward policy for goal state 53 predicts the correct behavior of authors in the trajectory set $\mathcal{T}_{53}$. Similar goal ranking results can be observed for the other trajectory sets, as shown in Figs. 11 (b-d).

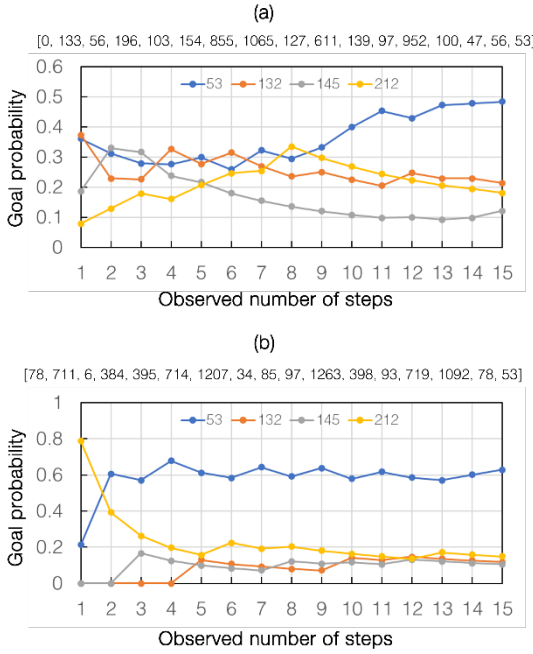

Figure 12. Goal probabilities of each state (53, 132, 145, and 212) as more number of steps (state transitions) are observed along two author trajectories that terminate at state 53 ((a) and (b)). Goal inference for each state was based on its respective reward policy. The sequence of states in the two trajectories are listed above each figure.

4.3.3. Goal inference via ranking the four goal states in each training trajectory set by their respective reward policy-based goal probabilities

To further validate the learned reward policies for goal inference, we determined which one out of the four states is the preferred goal state as an author progresses along a trajectory, by ranking the four states in decreasing values of their reward policy-based goal probabilities (calculated using Eq. (23)). For sake of illustration, we show in Fig. 12, the goal probabilities of two author trajectories that terminate at goal state 53. For the first author (Fig. 12(a)), state 53 is the most probable goal state after observing the first 9 state transitions (steps) of the trajectory. In the

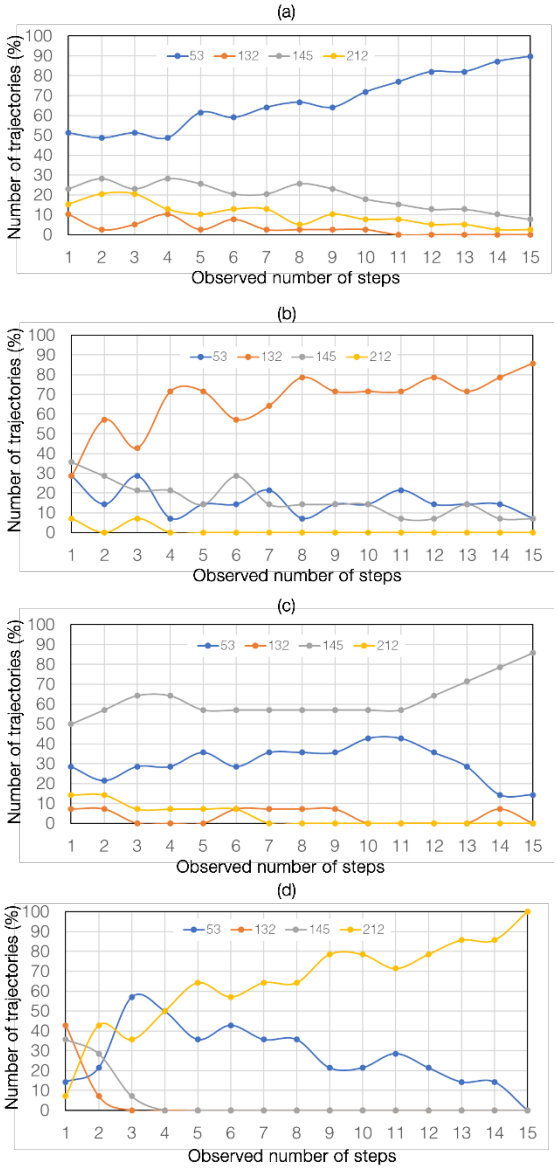

Figure 13. Percentage number of trajectories in sets $\mathcal{T}_{53}$ (a), $\mathcal{T}_{132}$ (b), $\mathcal{T}_{145}$ (c), and $\mathcal{T}_{212}$ (d), in which each state (53, 132, 145, and 212) was inferred as the goal state, based on the respective state's reward policy

first 8 states of the trajectory, it is not clear which of the four goal states, the author is pursuing. For the second author (Fig. 12(b)) we can see that the author is pursuing state 53 after two state transitions.

Based on their respective reward-based goal probabilities, we ranked the four states in all trajectories of each set and counted how many times each state ranked first as the goal state, as more state transitions (steps) are observed along each trajectory. The results for each trajectory set are plotted in Figs. 13 (a-d). We can see in Figs. 13(a) and 13(c) that the actual goal states, 53 and 145, were inferred in at least 50% of the trajectories in their respective trajectory sets ($\mathcal{T}_{53}$ and $\mathcal{T}_{145}$), as the observed number of steps

varied from 1 to 15. State 132 was inferred as the goal state in the first step of only 30% of the trajectories in set $\mathcal{T}_{132}$ (Fig. 13(b)), but this percentage rose above 50 after 4 steps. In the case of trajectories in set $\mathcal{T}_{212}$, the state 212 is the inferred goal state in at least 50% of the trajectories after 5 steps of observation. This number rose to 100% when the author was one step away from the goal state. There were only a small percentage of trajectories (about 10 to 15 %) in $\mathcal{T}_{53}$, $\mathcal{T}_{132}$, and $\mathcal{T}_{145}$, where the goal state could not be inferred. While the inference may be delayed in some cases, these results indicate that early inference of the actual goal state is possible, and the goal-directed behavior can be inferred using the reward policy-based goal probabilities of the four goal states.

their respective reward policy-based goal probabilities

We performed the goal inference test on the six coin-author trajectories that were not included in set $\mathcal{T}_{53}$. This test will determine how well the IRL-Bayesian goal inference approach will perform in inferring the coin state as the goal state out of the four states, 53, 132, 145, and 212. Using Eq. (23), we computed the probability of reaching each of the four states based on the number of steps observed in each coin-author trajectory. Figs. 14 (a-f) show the most likely goal state as more steps are observed along each of the six coin-author trajectories. As seen in Fig. 14, for some trajectories we can predict the coin state as the most likely goal state very early on in the author's observed trajectory. For example,

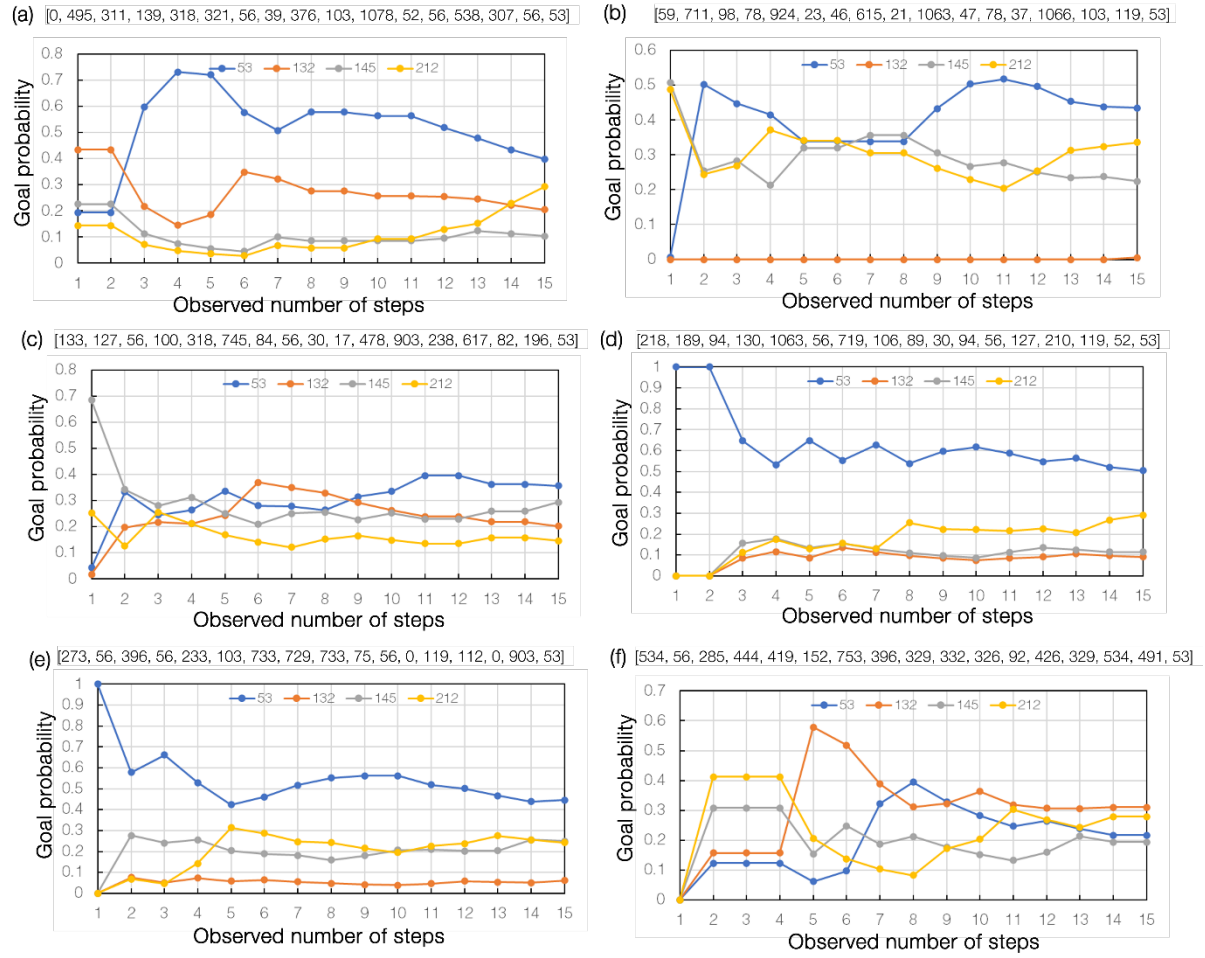

Figure 14. Goal probabilities of each state (53, 132, 145, and 212) as more number of steps (state transitions) are observed along the six coin-author trajectories that terminate at state 53 (a-f). Goal inference for each state was based on its respective reward policy. The sequence of states in the six trajectories are listed above each figure.

4.3.4. Goal inference via ranking the four goal states in previously unseen trajectories using

consider the trajectories in Figs. 14(a), 14(d), and 14(e). For the trajectory in Fig. 14(a), state 132 would be predicted as the goal state after observing the first two state transitions. However, after three state transitions, we can see the coin state as the most likely goal state for the

remaining length of the trajectory. For the author trajectories in Figs. 14(d) and 14(e), the goal probability of the coin state is the highest over the whole length of each trajectory, and therefore, is the most likely goal state. For the trajectory in Fig. 14(f), however, the coin state could not be inferred as the goal state. In other two trajectories (Fig. 14(b) and Fig. 14(c)), the inference for the coin state is delayed by 9 steps, but it is still 6 steps away from reaching the coin state. For the trajectory in Fig. 14(b), it is not clear what the goal state is after observing 5 to 8 steps of the trajectory. But after observing 9 to 15 steps, we can see that the coin state is the predicted goal state.

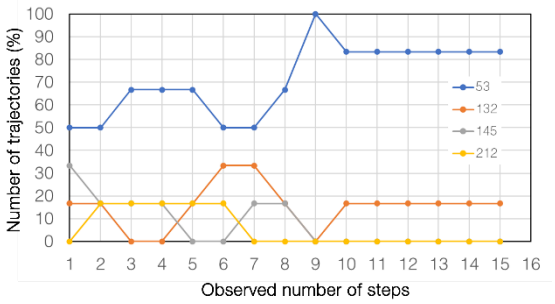

Figure 15. Percentage number of trajectories in the six coin-author trajectory set, where each state (53, 132, 145, and 212) was inferred as the goal state, based on the respective state's reward policy.

To quantify the accuracy of the IRL-based Bayesian goal inference approach, we counted the number of trajectories where the coin state is predicted as the goal state, as more steps are observed along each trajectory. As shown in Fig. 15, the coin state is correctly inferred as goal state in at least 50% of the trajectories (3 out of 6) after 1 to 8 steps of observation, and in 83% (5 out of 6) of the trajectories after 10 steps of observation. These results signify that with our IRL-based Bayesian formulation of goal probabilities, it is possible to infer whether an author is pursuing the coin state after observing a fraction of the author's trajectory before the state is reached.

5. Discussion

The results for the OPAL case study provide a proof-of-concept demonstration of the IRL-based Bayesian goal inference method to detect research activities of authors before they reach their goal state in a nuclear technology area. The case study focuses on a technology area related to the cold neutron source of the OPAL reactor, which may not directly relate to the building of the reactor, but indirectly points to its existence and operational use. The method used for the OPAL case study is, however, applicable to proliferation potential nuclear technologies, given the relevant datasets. Further research that goes beyond the scope of this paper is necessary to test the validity of the method for various technology use cases, assumptions, and variations in the method. In the following paragraphs, we discuss the different aspects of the method that can affect its performance and provide potential research directions to advance the method for applications in early detection of nuclear research activities.

The results and the performance of the IRL-based Bayesian inference method can vary depending on 1) how the topic weight vector space is defined, 2) which research articles are used in the dataset, 3) what grid spacing is used to discretize the topic weight vector space, 4) how the states and actions are defined for modeling the authors sequential decision-making process to publish in the coin state, 5) how much overlap in the state transitions exist among trajectories in a training set, and 6) how the reward function is defined. Future work may be directed towards understanding the effects of these factors.

In this work, we used NMF-based topic modeling to define the topic weight vector space of all the publications in the dataset. Other topic modeling approaches may be utilized, for example, Latent Dirichlet Association (LDA) method and classification algorithms based on large language models (LLMs) that are specifically trained on nuclear research articles. We trained the NMF topic model using the abstracts and titles of 29,196 Scopus records associated with the case study. These records comprised of a primary set of papers written by authors of the coin paper and a secondary set written by non-coin co-authors of the primary paper set. Instead of training the NMF model on the whole set, one could also train the NMF model using only the primary set and then compute the topic weights for the papers in the secondary set. This NMF fitting method can be used to confine the topic space to the primary set, which we have found to reduce the noise-to-signal ratio by resulting in a fewer number of author trajectories terminating in the coin state for IRL training. To avoid any bias due to the primary set, we considered to model the topics using the whole set.

In the case study, we used a single flagship publication of OPAL to define the coin state. The method can be extended to multiple coin states if there are multiple publications associated with the technology of interest. For the case with multiple coin states, the expert trajectory set

used for IRL training will include trajectories terminating in any of the coin states.

For the IRL framework, we defined the states based on the location of the publications in the discretized topic weight vector space and actions based on the year difference between two consecutive publications in an author's publication sequence. There is no prescribed way to define actions for the author topic MDP. Future work may explore other ways of defining the actions and analyze grid discretization effects on the goal inference. Reducing the grid spacing will increase the number of grid cells (states) and lower the grid occupancy. This can reduce the overlap between author trajectories and create better separation of author behaviors, which in turn may improve the performance of the inference method. Higher resolution grids will increase the size of the MDP problem and will require high performance computing resources to perform the reward learning and inference calculations.

In the OPAL case study, we computed the state and state-action rewards independent of the state and state-action features, respectively. If these features are known and are readily available, then the performance of the inference method may be improved by representing the rewards as a function of these features. The current lack of knowledge of these features can limit the accuracy of the goal inference.

When applying the IRL-based Bayesian inference approach to infer whether an author will pursue the coin state, we must first convert the author's partially observed publication sequence into a sequence of state-action-state transitions in the topic MDP environment. This is done first by computing the topic weight vectors of all the author's publications using the trained NMF model and mapping them to states in the topic weight vector space. There could be states and actions in the author's sequence, that are not part of the topic MDP environment if they were not previously observed. Therefore, we must expand the state and action spaces of the MDP to include previously unobserved states and actions, and then re-compute the transition probabilities based on all the state-action-state transitions observed in the author's partially observed sequence, and subsequently re-learn the rewards for the respective goal states, including the coin state. This will also update the transition probabilities of previously observed state-action-state transitions, and possibly the rewards of previously observed states and state-action pairs. If all the states and actions in the author sequence already exist in the MDP, we could choose not to update the transition probabilities and rewards. But as more data becomes available, it would be necessary to update the transition probabilities and re-train the rewards even if no new states need to be included in the MDP's state space. In the OPAL case study, we included the transitions of the test trajectory set when computing the transition probabilities.

Although we can incorporate previously unvisited states in the MDP before doing the inference, the prior goal probabilities given these states (are observed) will be zero, since these states were not present in any of the expert trajectory sets used for IRL training. The zero observed priors will result in zero goal probabilities for all tested goal states. When no observed goal priors exist for a new state, one could perform an inference with a uniform prior or impute a prior value based on proximity of the new state to previously visited states. The imputation method is like the open-world assumption used by Krumm and Horvitz (2006) for driver destination prediction when a driver can visit locations (grid cells) that have not been visited before. For our application, one may use grid interpolation methods to impute the prior goal probability value for a previously unvisited state based on those of previously visited states that are within some L1 or L2 norm distance from the unvisited state in the topic grid. Alternatively, one may consider replacing the new state in the author trajectory with the closest of the previously visited states in the topic grid based on L1 or L2 norm distance. Further research is necessary to test the validity and accuracy of using uniform and imputed goal priors when there are previously unvisited states in an author's trajectory that is tested for goal inference.

In the OPAL case study, we evaluated the performance of the method using trajectories that terminated at four different goal states (including the coin state). We evaluated how well the IRL-based reward policies captured the correct goal-directed behavior of authors in two ways: 1) by comparing the frequency of rankings based on the posterior probabilities of all states, computed using the reward policy for each goal state (see Fig. 11), and 2) by comparing the goal probabilities of reaching each of the four goal states given an observed fraction of the steps in a test trajectory (see Figs. 12 to 14). Based on the posterior probabilities that were computed using the reward policy for a goal state, we found that the actual goal state would be the inferred goal state among the top 5 states 90% of the steps in a trajectory of the training set. With a different reward policy, the frequency of rankings would drop by 30 to 50%. This suggested that

the reward policies capture the correct goal state behavior of authors. Based on the goal probability calculations, we found that the inference of the actual goal state can be early while in some cases delayed or unobservable. In practice, even though the actual goal state could be inferred five or six steps before it is reached, there is an uncertainty in finding the exact time of the event's occurrence due to the inherent time lag between the inception of a research activity and its publication date. This uncertainty will have to be quantified based on additional technology use cases and larger validation tests.

In this work, the length of the shortest coin-author trajectory was used as the maximum length of the trajectories ($L = 17$) in the IRL training set. But the trained model can be applied for making inferences on trajectories of length smaller or greater than L. In future studies, it will be instructive to look at how different values of L for the training trajectory set might affect the inference results. It is noted that the IRL-based Bayesian inference method is suited for inferring the goal state before an author reaches the goal state, even after one or two steps of observation. This is useful when inference must be made for cases where the observed sequence is short due to delays in publication reviews or due to fewer number of publications as in small (covert) projects compared to large public research programs.

Future work may be directed towards understanding what factors of the author trajectories limit the performance of the IRL-based Bayesian goal inference. For example, inference can be limited by high overlap in the state transitions between author trajectories, and by the fact that authors can re-visit a state multiple times or can access any state from a given state.

6. Conclusion

We have developed an IRL-based Bayesian goal inference method to predict whether an author would pursue a research activity (goal state) in a nuclear technology area, given partial observations of their state transitions in a research topic weight vector space. Our case study results surrounding a civil nuclear activity suggest that it is possible to infer whether an author would publish on a technology-directed research activity before it has occurred. This work represents the first attempt at using nuclear research articles for early detection of technology-directed research activities of authors. We have discussed various research directions to build upon this work and to improve the performance of the inference method. The present work provides a foundational framework for early detection of technology-directed activities from scientific and technical sources of information, where the early detection problem is formulated as a sequential, decision-making problem. The IRL-based Bayesian goal inference method, combined with advanced computing, may be used to assess and monitor activities pertaining to early developmental stages of a nuclear technology or capability, which in turn can help to identify and prioritize activities with nuclear proliferation potential for further investigation.

Acknowledgments

This research was supported by the Mathematics for Artificial Reasoning in Science Initiative – a Laboratory Directed Research and Development Program conducted at the Pacific Northwest National Laboratory (PNNL). The authors acknowledge Samrat Chatterjee for useful discussions during the initial formulation of the research concept. PNNL is a multi-program national laboratory operated for the U.S. Department of Energy by Battelle Memorial Institute under Contract No. DE-AC-5-76RL01830.

7. References

[1] Adams, S., Cody, T., and Beling, P. A. (2022). A survey of inverse reinforcement learning. *Artificial Intelligence Review, 55*(6), pp. 4307-4346.

[2] Arora, S. and Doshi, P. (2021). A survey of inverse reinforcement learning: Challenges, methods and progress. *Artificial Intelligence*, 297, pp. 103500.

[3] Alexander, F.J., Borders, T., Sheffield, A., & Wonders, M. (2020). *Workshop report for next-gen AI for proliferation detection: Accelerating the development and use of explainability methods to design AI systems suitable for nonproliferation mission applications*. Brookhaven National Lab, Upton, NY, USA; Idaho National Lab, Idaho Falls, ID, USA; National Nuclear Security Administration, Washington, DC, USA.

[4] Barletta, M., Feldman, Y., and Ferguson, M. (2014). *Scientific and technical information as source of IAEA safeguards state evaluation*. In 19th Pacific Basin Nuclear Conference; 38th Annual Student Conference of the Canadian Nuclear Society and Canadian Nuclear Association. Canada.

[5] Carlson, J., Russell, L., and Berriman, A. (2006). *Detection of Undeclared Nuclear Activities: Does the IAEA Have the Necessary Capabilities*. In Conference proceedings at Institute of Nuclear Materials Management Annual Meeting. Nasvhille, TN.

[6] Chatterjee, S., Thomas, D., Fortin, D., Pazdernik, K., Wilson, B., & Newburn, L. (2023). Dynamic network analysis of nuclear science literature for research influence assessment. *ESARDA Bulletin – The International Journal of Nuclear Safeguards and Non-Proliferation, 65*, pp. 19-33.

[7] Cojazzi, G., Van Der Goot, E., Verile, M., Wolfart, E., Rutan Fowler M., Feldman, Y., Hammond, W., Schweighardt, J., and Ferguson, M. (2013). Collection and analysis of open source news for information awareness and early warning in nuclear safeguards. *ESARDA Bulletin – The International Journal of Nuclear Safeguards and Non-Proliferation, 50*, pp. 94-105

[8] Ferguson, M. and Norman, C. (2010). *All-source information acquisition and analysis in the IAEA department of safeguards*. In IAEA Symposium on International Safeguards, Vienna, IAEA-CN-184/048.

[9] Kas, M., Khadka, A.G., Frankenstein, W., Abdulla, A.Y., Kunkel, F., Carley, L.R., and Carley, K.M. (2012). Analyzing scientific networks for nuclear capabilities assessment. *Journal of the American Society for Information Science and Technology, 63*, pp. 1294-1312.

[10] Krumm, J. and Horvitz, E. (2006). *Predestination: Inferring destinations from partial trajectories.* In International Conference on Ubiquitous Computing (pp. 243-260). Berlin, Heidelberg: Springer Berlin Heidelberg.

[11] Ng, A. Y. and Russell, S. (2000). *Algorithms for inverse reinforcement learning*. In ICML, 1 (pp. 2).

[12] Olsen, S.R., Kennedy, S.J., Kim, S., Schulz, J.C., Thiering, R., Gilbert, E.P., Lu, W., James, M., and Robinson, R.A. (2008). *Novel cryogenic engineering solutions for the new Australian Research Reactor OPAL*. In AIP Conference Proceedings, 985, 1 (pp. 299-306). American Institute of Physics.

[13] Pabian, F., Renda, G., Jungwirth, R., Kim, L., Wolfart, E., and Cojazzi, G. (2014). *Open source analysis in support to non-proliferation monitoring and verification activities: Using new media to derive unknown new information*. In Proceedings Symposium on International Safeguards: Linking strategy, implementation and people, IAEA-CN-220, 312.

[14] Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V. and Vanderplas, J. (2011). Scikit-learn: Machine learning in Python. *The Journal of machine Learning research, 12*, pp. 2825-2830

[15] Ramachandran, D., and Amir, E. (2007). *Bayesian Inverse Reinforcement Learning*. In International Joint Conferences on Artificial Intelligence (IJCAI), 7 (pp. 2586-2591).

[16] Satopaa, V., Albrecht, J., Irwin, D., and Raghavan, B. (2011). *Finding a "kneedle" in a haystack: Detecting knee points in system behavior*. In 2011 31st international conference on distributed computing systems workshops (pp. 166-171). IEEE.

[17] Sheffield, A. (2020). Developing the next-generation of AI systems to push the detection of foreign nuclear proliferation further "left of boom". *Countering WMD Journal, 21*, pp. 100-102.

[18] Snoswell, A. J., Singh, S. P., and Ye, N. (2020). *Revisiting maximum entropy inverse reinforcement learning: new perspectives and algorithms*. In 2020 IEEE Symposium Series on Computational Intelligence (SSCI) (pp. 241-249). IEEE.

[19] Sutton, R.S., and Barto, A.G. (2018). *Reinforcement learning: An introduction*. MIT press.

[20] Xue, A. Y., Qi, J., Xie, X., Zhang, R., Huang, J., and Li, Y. (2015). Solving the data sparsity problem in destination prediction. *The VLDB Journal, 24*, pp. 219-243.

[21] Ziebart, B. D., Maas, A. L., Bagnell, J. A., and Dey, A. K. (2008). *Maximum entropy inverse reinforcement learning*. In 8th Aaai Conference (pp. 1433-1438).