Predicting Learner Interactions in Social Learning Networks

***Abstract*—We consider the problem of predicting link formation in Social Learning Networks (SLN), a type of social network that forms when people learn from one another through structured interactions. While link prediction has been studied for general types of social networks, the evolution of SLNs over their lifetimes coupled with their dependence on which topics are being discussed presents new challenges for this type of network. To address these challenges, we develop a time-series prediction methodology that uses a recurrent neural network architecture to pass network state between time periods, and that models over three types of SLN features updated each period: neighborhood-based (*e.g.,* resource allocation), path-based (*e.g.,* shortest path), and post-based (*e.g.,* topic similarity). Through evaluation on four real-world datasets from Massive Open Online Course (MOOC) discussion forums, we find that our method obtains substantial improvements over a Bayesian model and an unsupervised baseline, with AUCs typically above 0.75 and reaching 0.97 depending on the dataset. Our feature importance analysis shows that while neighborhood-based features contribute the most to the results, post-based and path-based features add additional information that significantly improve the predictions. We also find that several input features have opposite directions of correlation between link formation and post quality, suggesting that response time and quality are two competing objectives to be accounted for in SLN link recommendation systems.**

I. Introduction

Online education has exploded in popularity over the past few years, with estimates of up to 80% of students having taken an online course [1]. This growth has not been without challenges, however; online learning has raised concerns about its apparent lack of quality control, extraordinarily low teacher-to-student ratios, and scarcity of high-quality teachers [2].

One way course providers have attempted to mitigate these problems is by establishing online forums where students can learn from each other, compensating for a lack of personalized instruction by posting questions, replying with answers, and otherwise exchanging ideas. Massive Open Online Courses (MOOCs), as well as Q&A sites like Quora and StackOverflow, rely on forums extensively, generating a plethora of data about how users interact with one another online for learning purposes. Data-driven studies on the Social Learning Networks (SLN) emerging from these forums have analyzed the benefits of social learning [3], and have also proposed methods for *e.g.,* instructor analytics [4] and news feed personalization [5] towards the ultimate goal of improving learning outcomes.

In this work, we are motivated by the following research question: *Can link formation between learners in an SLN be predicted in advance?* Such predictions would enable several new ways of improving forum experiences, *e.g.,* encouraging early formation of groups of learners who are expected to frequently communicate, or recommending that learners respond to newly-posted questions that they are expected to answer/contribute to later. Predicting how these links develop, however, poses many challenges unique to SLN. For one, unlike social networking sites with clearly defined relationships between nodes (*e.g.,* friendships), links in a discussion forum are more ambiguous [5]. Moreover, whether two users will interact likely depends not only on their "closeness," but also on whether they are interested in discussing similar topics. Further, the topology of a course's SLN will evolve substantially throughout its duration, starting from the extreme case of no observable network when the course starts.

To address these challenges, we develop a time-series link prediction methodology that uses a set of features describing learner pairs in an SLN and how the SLN evolves over time. We evaluate our method on data collected from four MOOC discussion forums. We then investigate how our methodology can be used to make recommendations that may enhance the timing and quality of replies to questions.

A. Related Work

The link prediction problem has been studied extensively in the context of online social networks, due to its usefulness in generating recommendations for *e.g.,* friendships or interactions (see [6] for a survey). Several methods have been proposed for this problem, with earlier ones based on unsupervised approaches and more recent ones using supervised methods. In terms of unsupervised methods, [7] proposed using features based on node proximity and properties, while [8] applied a hierarchical network model to predict missing connections. Supervised approaches have included a supervised random walk algorithm using labels to increase the likelihood of traversing formed links [9], while [10], [11] derived features from exogenous sources and trained models on them to predict future link formation. Unlike these works, in our supervised models we consider characteristics unique to Social *Learning* Networks: Potential dependence on discussion topics, and the need for time-series modeling.

Other works on online social networks have considered problems related to link formation, *e.g.,* predicting the strength/repetition (rather than existence) of future links [12]–[14] or predicting link types [15], [16]. The methods used and developed include linear regression/classification on network features and user demographics [13], [15], latent variable modeling of learner interaction frequencies [12], factor graph models [16], and dynamic models to account for the disappearance and strengthening of links over time [14]. Our models

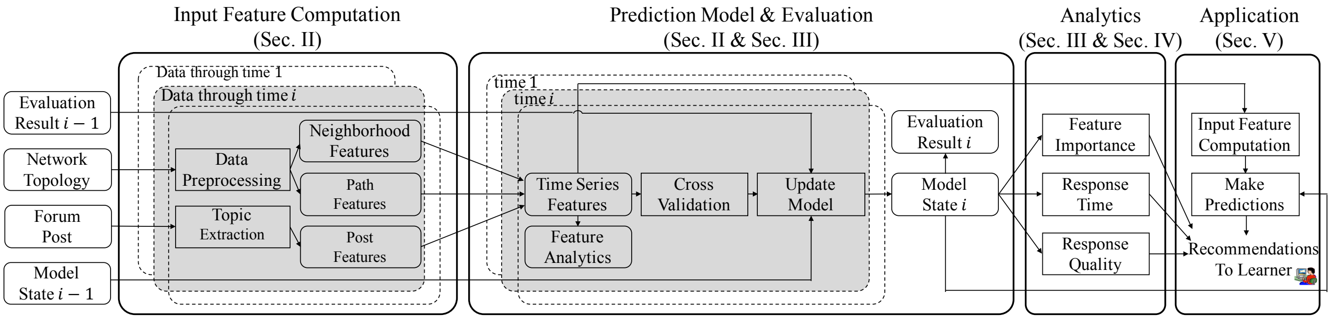


Fig. 1: Summary of the different components of the SLN link prediction methodology we develop in this paper.

utilize some similar network features, but we consider the different prediction objective of pinpointing when links will form. In fact, given its high observed quality (up to 80% AUC), we consider a time-series version of [12] as a potential model.

Some recent works have focused on other research questions about various types of SLNs, *e.g.,* MOOCs [1], [5], [17], Q&A sites [2], [18], and enterprise social networks [19]. Our work is perhaps most similar to [1], [17] in that we study prediction for SLNs. The prediction objectives in these other works, however, are fundamentally different than our focus of predicting interactions between learners: they seek to predict course grades via video-watching behaviors [1] and course completion via learner post and reply frequencies [17].

B. Our Methodology and Contributions

In this paper, we develop time-series link prediction algorithms for an SLN that consider both the network structure and latent learner post characteristics. Fig. 1 summarizes the main components of our methodology. The first part is feature engineering, in which the SLN is extracted from the discussion data (Sec. II-A) and transformed into a set of features for prediction (Sec. II-B). We define three groups of features for each pair of learners: (i) *Neighborhood-based* features that are determined from common neighborhoods, (ii) *Path-based* features based on paths between learners, and (iii) *Post-based* features that are determined from latent topic analysis of learner posts. In quantifying the SLN, a key question we address is what constitutes a link between two learners [5], which must be inferred from the forum data.

The second component in Fig. 1 is the prediction model (Sec. II-C). We consider two different models: (i) Bayesian networks, and (ii) Long-Short Term Memory (LSTM) Recurrent Neural Network (RNN) units. The success of Bayesian models in static link prediction problems [12] motivates us to consider their performance in the time-evolving SLN setting. We develop our core methodology around LSTM, though, because its gating mechanisms can account for long term dependencies between actions [20], allowing it to pass network state across time intervals. As shown in Fig. 1, at each time i, we update our prediction model using the SLN state from $i-1$ and the new features at i, which results in the predictions for time $i+1$; to our knowledge, the methodology we develop around the neural network architecture is the first to encapsulate the time-evolving nature of an SLN.

To assess the quality of our models, we train and evaluate the Bayesian and neural network predictors on four MOOC discussion forums, using an unsupervised method as a baseline (Sec. III). Through our evaluation, we also generate the three types of analytics in the third component of Fig. 1: one is feature importance, which quantifies how important specific features and groups of features are to the prediction. Additionally, we consider the application of our method to recommending link formation (Sec. IV), which involves analyzing how the features relate to response timing and quality.

From the evaluation and associated analytics, our three key findings and contributions are as follows:

- We show that our time-series neural network algorithm obtains substantial improvements over other models for each dataset, with AUCs above 0.75 and up to 0.97.
- We show that while the neighborhood-based features are the most important for link prediction quality, path and post-based features also have a significant impact.
- We find that certain features that have significant positive correlations with link formation have negative associations with response quality, and vice versa.

This last point of opposing correlation directions indicates that response time and quality are two competing objectives in SLN link recommendation. Thus, in addition to developing the first method for link prediction in SLNs, we also give direction for what components link recommendations should be based on to ultimately improve learning experiences (Sec. V).

II. Link Prediction Model

In this section, we formalize our prediction model. We first quantify an SLN from forum data (Sec. II-A) and define the particular features (Sec. II-B), before introducing prediction methods (Sec. II-C) and briefly overviewing our training methods (Sec. II-D).

A. SLN Graph Model

In order to define our features, we must first describe how an SLN and link creation is inferred from online forum data.

Online forums. An online forum is typically comprised of a series of threads, with each thread in turn being comprised of one or more posts. Each post is written by a single user. A post, in turn, can have one or more comments attached to it. Given the observation that SLN forum users do not abide by the designation of post vs. comment consistently [5], we will not distinguish between them, instead referring to them both as posts. This structure of thread posts is depicted in Fig. 2.

Quantifying SLN link creation. We let $\mathcal{T}$ denote a given thread in an online forum and use $p_n \in \mathcal{T}$ to denote the nth

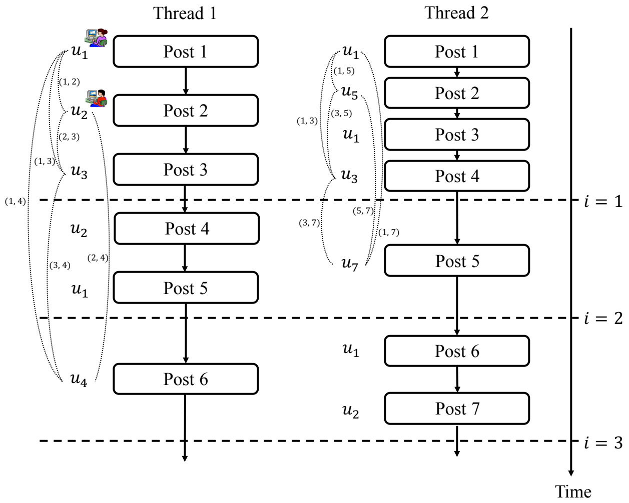


Fig. 2: Example of how posts in two different threads in an online discussion forum are divided into time periods, and how SLN link creation between the learners authoring these posts is modeled.

post created in a course, made by user u at time t_n. This post will contain a body of text $\boldsymbol{x}_n$ written by u. A *link* (u, v) is observed between learner u and another learner v if, at a later time $t_{n'} > t_n$, v writes a post $p_{n'} \in \mathcal{T}$ in the same thread. We use this as the criteria for establishing the link (u, v) in the SLN because it signifies the fact that learner u and learner v have exchanged ideas and interacted in the same thread.

To model the evolution of an SLN, we group its posts into different time intervals. To this end, let $T_c = t_N - t_1$ be the time elapsed between the first p_1 and last p_N posts made in a forum. We divide all posts in this forum into L equally spaced intervals of length $m_L = T_c/L$. Formally, we say that post j will belong to interval i iff $t_j \in (t_1 + (i-1) \cdot m_L, t_1 + i \cdot m_L)$. Fig. 2 illustrates this procedure for two example threads.

We use $y_{uv}(i)$ as an indicator variable for the formation of link (u, v): $y_{uv}(i) = 1$ if a link between u and v has been created in any interval $1, ..., i \leq L$ and $y_{uv}(i) = 0$ otherwise. Thus, as in most social networks [9]–[11], links persist over time in our SLN model. The SLN graph structure in any given interval i is then comprised of nodes corresponding to the learners u and edges (u, v) corresponding to links between them. For the purpose of predicting future responses, we consider this interaction to be bidirectional, *i.e.,* the resulting SLN is an undirected graph. Formally, we define $\mathcal{G}(i) = [y_{uv}(i)]$ as the binary adjacency matrix of the SLN during interval i; since links are bidirectional, $\mathcal{G}(i)$ is symmetric.

Fig. 3 visualizes $\mathcal{G}(i)$ at different points in time for one of our datasets. We see that although it maintains the same qualitative structure, it visibly evolves as the course progresses, suggesting the potential benefit of time-series modeling.

B. SLN Features

We now define our features, computed for each learner pair $(u, v), u \neq v$ for each time $i = 1, ..., L$. These quantities serve as the inputs to our prediction algorithms in Sec. II-C.

Neighborhood-based features. These features, as well as the next group, are extracted from the topology of the graph. Letting $N(\mathcal{G})$ be the set of nodes in the SLN $\mathcal{G}$ and $\Gamma_u(i) \subseteq N(\mathcal{G})$ denote the set of neighbors of u at time i, the neighborhood-based features qualitatively measures the "similarity" of u and v's neighborhoods [7]. They are quantified as follows:

1) Jaccard coefficient (`Ja`): $|\Gamma_u(i) \cap \Gamma_v(i)| / |\Gamma_u(i) \cup \Gamma_v(i)|$

2) Adamic-Adar index (`Ad`): $\sum_{n \in \Gamma_u(i) \cap \Gamma_v(i)} 1/\log|\Gamma_n(i)|$

3) Resource allocation index (`Re`): $\sum_{n \in \Gamma_u(i) \cap \Gamma_v(i)} 1/|\Gamma_n(i)|$

4) Preferential attachment score (`Pr`): $|\Gamma_u(i)| \cdot |\Gamma_v(i)|$

We let $\mathbf{b}_{uv}(i)$ denote the vector of these features for u and v at time i. Note that a larger value of each of these features, roughly speaking, indicates that u and v share more common, low degree neighbors than they do with other learners.

Path-based features. These features measure the proximity of u and v in the SLN at time i. They are as follows:

5) Shortest path length (`Lp`): The length of the shortest path between u and v at time i.

6) Number of paths (`Np`): The number of shortest paths (*i.e.,* of length `Lp`) between u and v at time i.

We let $\mathbf{a}_{uv}(i)$ denote the vector of these features. Note that as `Lp` decreases, u and v become more closely connected, while a larger `Np` indicates more redundancy in these paths.

Post-based features. Besides topology-based attributes, learners' interests in different course topics will also influence their probability of forming links in an SLN. In particular, we would expect those with similar topic interests to be more likely to post in the same thread, *i.e.,* form links. We thus compare the topics of different learners' posts to compute another feature that shows the learners' similarity in interests.

To do this, we let $\boldsymbol{d}_n = (d_{n,1}, d_{n,2}, ...)$ be the sequence of word indices for post n from the dictionary of all course words $\mathcal{X} = \boldsymbol{x}_1 \cup \boldsymbol{x}_2 \cup \cdots$. We then apply an appropriate natural language processing algorithm across the $\boldsymbol{d}_n$ to extract a set of latent topics $\mathcal{K}$ and to model each post p_n as different combinations of these topics, *i.e.,* $\boldsymbol{v}_n = \{v_{n,k} | k \in \mathcal{K}\}$ where $v_{n,k}$ is the proportion of p_n made up of topic k. With this in hand, for each post we choose $k_n = \text{argmax}_{k \in \mathcal{K}} v_{n,k}$, *i.e.,* the topic with highest proportion, to serve as a main topic for p_n. Then, for each learner, we obtain the set of main topics across their posts through time i as $K_u(i) = \{k_n | n \in \mathcal{P}_u(i)\}$, where $\mathcal{P}_u(i)$ is the set of posts written by learner u through time i. With this, we define the last feature:

7) Number of common topics (`To`): $|K_u(i) \cap K_v(i)|$

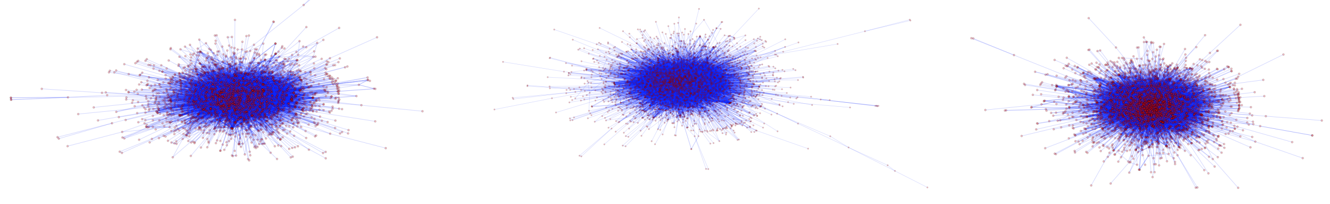
We use $c_{uv}(i)$ as the time-series version of `To`: the number of common topics discussed by u and v through time i.

C. Link Prediction Algorithms

Fig. 4 summarizes our algorithm architecture. At each time i, the input for a given pair (u, v) of users is the feature vector $\boldsymbol{e}_{uv}(i) = [\boldsymbol{b}_{uv}(i), \boldsymbol{a}_{uv}(i), c_{uv}(i)]$ defined in Sec. II-B, while the target output is the link state $y_{uv}(i) \in \{0, 1\}$. The model of the latent state $\boldsymbol{z}_{uv}(i)$ for each algorithm is described next.

Bayesian model. The Bayesian Network (BNet) model [12] defines the probability density of $\boldsymbol{z}_{uv}(i)$ as a Gaussian:

$$P(\boldsymbol{z}_{uv}(i) | \boldsymbol{e}_{uv}(i)) = \mathcal{N}(\boldsymbol{w}^T \boldsymbol{e}_{uv}(i), \sigma^2) \quad (1)$$



(a) $\mathcal{G}(3)$ (b) $\mathcal{G}(6)$ (c) $\mathcal{G}(9)$

Fig. 3: Snapshots of the evolution of the SLN topology for the `ml` dataset at $i = 3, 6,$ and 9 with $L = 10$.

Forum	Title	Beginning	Duration	Users	Threads	Learner pairs $\vert\mathcal{G}(L)\vert$	Posts	L_1	L_2
`ml`	Machine Learning	4/29/13	12	4263	4217	73315	25481	10	20
`algo`	Algorithms: Design and Analysis I	9/22/14	13	3013	4656	50066	16276	11	19
`comp`	English Composition I	7/01/13	8	1862	1256	20083	8255	10	23
`shake`	Shakespeare in Community	4/22/15	5	958	1389	66217	7484	10	16

TABLE I: Background information on the four MOOC online forum datasets. The title, beginning date (m/dd/yy), duration (weeks), number of users, threads, learner pairs by the end, posts, and number of time instances L_1 and L_2 used in evaluation are given for each.

where $\boldsymbol{w}$ is weight vector to be estimated when the model is trained, and σ^2 is the variance (fixed to 0.5 in this work following [12]). From this, $y_{uv}(i)$ is estimated according to

$$P(y_{uv}(i) = 1|\boldsymbol{z}_{uv}(i)) = \sigma(\boldsymbol{\phi}^T \boldsymbol{z}_{uv}(i) + b) \tag{2}$$

where $\boldsymbol{\phi}$ and b are a vector and scalar, respectively, to be estimated during training, and $\sigma(\cdot)$ is the logistic function.

Neural network model. One potential limitation of the Bayesian model is that, while $\boldsymbol{z}_{uv}$ varies over time based on $\boldsymbol{e}_{uv}$, it is not directly evolving from its previous states. This could be important to modeling an SLN for a number of reasons, particularly so that the predictor could respond to sudden changes in the input relative to the prior state. This could occur, for example, when the topic of the course shifts, which could be reflected in a sudden change in $c_{uv}(i)$.

To capture this property, we introduce a time-series model based on LSTM Recurrent Neural Networks (NNet), which have noted success in capturing dependencies over long time periods [20]. We first define $\boldsymbol{d}_{uv}(i) = [\boldsymbol{e}_{uv}(i), \boldsymbol{h}_{uv}(i-1)]^T$ as a summary of the inputs to the model at time i, where $\boldsymbol{h}(0) = \boldsymbol{0}$ and $\boldsymbol{h}(i-1)$ is the output vector from the previous time. We then define interaction gate, relationship gain gate, and relationship fading gate vectors at each time interval i as

$$\begin{aligned}&\boldsymbol{g}_{uv}(i) = \varphi(\boldsymbol{W}_g \boldsymbol{d}_{uv}(i) + \boldsymbol{b}_g), \;\; \boldsymbol{i}_{uv}(i) = \sigma(\boldsymbol{W}_i \boldsymbol{d}_{uv}(i) + \boldsymbol{b}_i),\\ &\boldsymbol{f}_{uv}(i) = \sigma(\boldsymbol{W}_f \boldsymbol{d}_{uv}(i) + \boldsymbol{b}_f)\end{aligned} \tag{3}$$

respectively. Here, $\varphi(\cdot)$ and $\sigma(\cdot)$ are the tanh and sigmoid functions, respectively, and the matrices $\boldsymbol{W}_g$, $\boldsymbol{W}_i$, and $\boldsymbol{W}_f$ as well as the vectors $\boldsymbol{b}_g, \boldsymbol{b}_i$, and $\boldsymbol{b}_f$ contain parameters that are estimated during the model training procedure. By these definitions, the interaction vector $\boldsymbol{g}$ will contain new candidate values from $\boldsymbol{d}_{uv}(i)$, and the gain gate $\boldsymbol{i}$ will specify the degree to which the input values in $\boldsymbol{d}_{uv}(i)$ will be used in updating $\boldsymbol{z}$, the latent cell state. The fading gate $\boldsymbol{f}$ indicates the degree to which prior elements from $\boldsymbol{z}$ will be used in the new state. Formally, $\boldsymbol{z}_{uv}$ is updated as

$$\boldsymbol{z}_{uv}(i) = \boldsymbol{g}_{uv}(i) \odot \boldsymbol{i}_{uv}(i) + \boldsymbol{z}_{uv}(i-1) \odot \boldsymbol{f}_{uv}(i), \tag{4}$$

where $\odot$ denotes element-wise matrix multiplication. We then use an output gate $\boldsymbol{o}$ to determine the factor to which each element of $\boldsymbol{z}$ should be used in the definition of $\boldsymbol{h}$:

$$\boldsymbol{o}_{uv}(i) = \sigma(\boldsymbol{w}_o \boldsymbol{d}_{uv}(i) + \boldsymbol{b}_o), \;\; \boldsymbol{h}_{uv}(i) = \sigma(\boldsymbol{z}_{uv}(i) \odot \boldsymbol{o}_{uv}(i)) \tag{5}$$

With this, $y_{uv}(i)$ is estimated as

$$P(y_{uv}(i) = 1|\boldsymbol{z}_{uv}(i)) = \sigma(\boldsymbol{h}_1(i)) \tag{6}$$

where $\boldsymbol{h}_1(i)$ is the first element of $\boldsymbol{h}(i)$. Note that $\boldsymbol{g}$, $\boldsymbol{i}$, $\boldsymbol{f}$, $\boldsymbol{o}$, $\boldsymbol{z}$ and $\boldsymbol{h}$ are each vectors of the same dimension N.

D. Model Parameter Training

We now briefly sketch the methods that we use to estimate each model's parameters. This is performed during the training/evaluation procedures described in Sec. III.

Bayesian model. Given $y_{uv}(i)$ and $\boldsymbol{e}_{uv}(i)$ in a training dataset, we wish to estimate the model variable $\boldsymbol{z}_{uv}(i)$ for each i and parameters $\boldsymbol{w}$, $\boldsymbol{\phi}$, and b. Following [12], we first define the log-likelihood function of (2) using the definition of $\boldsymbol{z}_{uv}(i)$ in (1), and then use Newton-Raphson iterations to find the parameters that optimize the log-likelihoods. To avoid overfitting, we use L2 regularizers on the parameters $\boldsymbol{w}$ and $\boldsymbol{\phi}$. We update the parameters found at each Newton-Raphson step until convergence.

Given all the known variables and the Gaussian priors, the joint probability can be expressed as follows:

$$\begin{aligned}&P_{joint}\\ &\quad = P(N(\mathcal{G}(i))|\boldsymbol{w}, \boldsymbol{\phi})P(\boldsymbol{w})P(\boldsymbol{\phi})\\ &\quad = \prod_{(u,v)\in N(\mathcal{G}(i))} P(z_{uv}(i)|\boldsymbol{e}_{uv}(i), \boldsymbol{w})P(y_{uv}(i)|z_{uv}(i), \boldsymbol{\phi})P(\boldsymbol{w})P(\boldsymbol{\phi})\\ &\quad \propto \prod_{(u,v)\in N(\mathcal{G}(i))} \left(e^{-\frac{1}{2v}(\boldsymbol{w}^T\boldsymbol{e}_{uv}(i) - z_{uv}(i))^2} \frac{e^{-(\boldsymbol{\phi}^T z_{uv}(i)+b)(1-y_{uv}(i))}}{1+e^{-(\boldsymbol{\phi}^T z_{uv}(i)+b)}}\right)\\ &\quad \cdot e^{-\frac{\lambda_w}{2}\boldsymbol{w}^T\boldsymbol{w}} e^{-\frac{\lambda_\phi}{2}\boldsymbol{\phi}^T\boldsymbol{\phi}}\end{aligned} \tag{7}$$

Next, we take the logarithm of Eq. 7 and get the log-likelihood function L:

$$\begin{aligned} L = & \sum_{(u,v)\in N(\mathcal{G}(i))} -\frac{1}{2v}(\boldsymbol{w}^T \boldsymbol{e}_{uv}(i) - z_{uv}(i))^2 \\ & - \sum_{(u,v)\in N(\mathcal{G}(i))} (1 - y_{uv}(i))(\boldsymbol{\phi}^T z_{uv}(i) + b) \\ & - \log\left(1 + e^{-(\boldsymbol{\phi}^T z_{uv}(i)+b)}\right) \\ & - \frac{\lambda_w}{2}\boldsymbol{w}^T\boldsymbol{w} - \frac{\lambda_\phi}{2}\boldsymbol{\phi}^T\boldsymbol{\phi} + C \end{aligned} \tag{8}$$

Then we use the following first and second derivatives of L. For $\boldsymbol{\phi}$:

$$\begin{aligned} \frac{dL}{d\boldsymbol{\phi}} = & \sum_{(u,v)\in N(\mathcal{G}(i))} \Big(y_{uv}(i) - \sigma(\boldsymbol{\phi}^T z_{uv}(i) + b)\Big) z_{uv}(i) \\ & - \lambda_\phi \boldsymbol{\phi} \end{aligned} \tag{9}$$

$$\begin{aligned} & \frac{d^2 L}{d\boldsymbol{\phi} d\boldsymbol{\phi}^T} \\ & = - \sum_{(u,v)\in N(\mathcal{G}(i))} \frac{e^{-(\boldsymbol{\phi}^T z_{uv}(i)+b)}}{\left(1 + e^{-(\boldsymbol{\phi}^T z_{uv}(i)+b)}\right)^2} z_{uv}(i) z_{uv}(i)^T \\ & - \lambda_\phi \boldsymbol{I} \end{aligned} \tag{10}$$

For $z_{uv}(i)$:

$$\begin{aligned} \frac{dL}{dz_{uv}(i)} = & \frac{1}{v}(\boldsymbol{w}^T \boldsymbol{e}_{uv}(i) - z_{uv}(i)) \\ & + \Big(y_{uv}(i) - \sigma(\boldsymbol{\phi}^T z_{uv}(i) + b)\Big)\boldsymbol{\phi} \end{aligned} \tag{11}$$

$$\frac{d^2 L}{d(z_{uv}(i))^2} = -\frac{1}{v} - \frac{e^{-(\boldsymbol{\phi}^T z_{uv}(i)+b)}\boldsymbol{\phi}^2}{\left(1 + e^{-(\boldsymbol{\phi}^T z_{uv}(i)+b)}\right)^2} \tag{12}$$

We then update $\boldsymbol{\phi}$ and $z_{uv}(i)$ based on Newton-Raphson updates:

$$\boldsymbol{\phi}^{new} = \boldsymbol{\phi}^{old} - \frac{dL}{d\boldsymbol{\phi}} \Big/ \frac{d^2 L}{d\boldsymbol{\phi} d\boldsymbol{\phi}^T} \tag{13}$$

$$z_{uv}(i)^{new} = z_{uv}(i)^{old} - \frac{dL}{dz_{uv}(i)} \Big/ \frac{d^2 L}{d(z_{uv}(i))^2} \tag{14}$$

Finally, we can directly calculate the optimal $\boldsymbol{w}$:

$$\boldsymbol{w}^{new} = (\lambda_w \boldsymbol{I} + \boldsymbol{S}^T \boldsymbol{E})^{-1} \boldsymbol{E}^T \boldsymbol{z} \tag{15}$$

where $\boldsymbol{E} = \left[\boldsymbol{e}_{u_1 v_1}(i), \boldsymbol{e}_{u_2 v_2}(i), ..., \boldsymbol{e}_{u_N v_N}(i)\right]^T$, and $\boldsymbol{z} = \left[\boldsymbol{z}_{u_1 v_1}(i), \boldsymbol{z}_{u_2 v_2}(i), ..., \boldsymbol{z}_{u_N v_N}(i)\right]^T$.

Neural network model. We use the standard backpropagation through time (BPTT) algorithm proposed by [22] to infer the model parameters $\boldsymbol{W}_g, \boldsymbol{W}_i, \boldsymbol{W}_f$, $\boldsymbol{W}_o$, $\boldsymbol{b}_g$, $\boldsymbol{b}_i$, $\boldsymbol{b}_f$ and $\boldsymbol{b}_o$. We apply the dropout operator only to the non-recurrent connections [23] and use AdaDelta, an adaptive learning rate method.

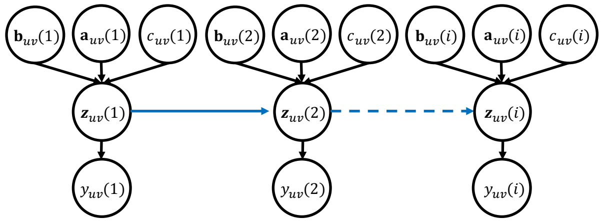


Fig. 4: Architecture of our time-series prediction algorithm. The neural network model includes the links between $\boldsymbol{z}_{uv}(i-1)$ and $\boldsymbol{z}_{uv}(i)$, while the Bayesian model does not.

III. Model Evaluation and Analytics

In this section, we describe our datasets (Sec. III-A), metrics (Sec. III-B), and prediction model evaluation (Sec. III-C). We then examine the model components (Secs. III-D & III-E).

A. Description of Datasets

We scraped the discussion forum data from four MOOCs on Coursera: "Machine Learning" (`ml`), "Algorithms: Design and Analysis, Part 1" (`algo`), "English Composition I" (`comp`), and "Shakespeare in Community" (`shake`). We chose these courses to include a diverse set of subjects, two quantitative in nature and two in the humanities. Table I gives basic statistics of the four datasets.

In what follows, we describe the SLNs in terms of the features in Sec. II-B. We make several observations on associations with link formation before using them for prediction.

Topic extraction. To obtain the post similarities $c_{uv}(i)$, we must first extract the topics $\mathcal{K}$ and distributions $\boldsymbol{d}_n$ for each post. We do so with Latent Dirichlet allocation (LDA), a generative model for extracting topics from a set of documents [24]. In our application, we view each post as a separate "document," since learners are likely to discuss many distinct topics over time. Prior to building the dictionary $\mathcal{X}$, all URLs, punctuations, and stopwords are removed from each post's text $\boldsymbol{x}_n$, and all words are stemmed. Table II summarizes the topic extraction results for each dataset using $|\mathcal{K}| = 20$ topics; the the top three words shown for the five topics that have the highest support across posts. With this value of $|\mathcal{K}|$, the topics are reasonably disjoint but have broad supports.

Feature correlations. Letting $\Omega = \{(u,v) : u, v \in N(\mathcal{G}), u \neq v\}$, *i.e.,* all possible learner pairs in the SLN, we define two subsets of Ω: $\mathcal{G}(L)$, which is the set of formed links at the final time $i = L$ (*i.e.,* with $y_{uv}(L) = 1$), and $\mathcal{G}^c(L) = \Omega \setminus \mathcal{G}(L)$, the complement graph of un-formed links (*i.e.,* $y_{uv}(L) = 0$). Note from Table I that $|\mathcal{G}^c(L)| \gg |\mathcal{G}(L)|$ for each dataset: most learners are never linked. Thus, to obtain a comparison of the features corresponding to formed and unformed links, we do the following: (i) for $\mathcal{G}(L)$ we compute $\mathbf{b}_{uv}(L)$, $\boldsymbol{a}_{uv}(L)$, and $c_{uv}(L)$ for all $(u,v) \in \mathcal{G}(L)$, while (ii) for $\mathcal{G}^c(L)$ we draw five random samples of size $|\mathcal{G}(L)|$, compute the features for each sample, and average the results.

Table III summarizes the distributions of the formed (top row) and unformed (bottom row) link groups, with the top

k	Support	Top-3 words
1	9.18%	code correct answer
2	8.76%	learn machin ng
3	7.33%	post code thread
4	6.25%	octav file work
5	5.42%	network layer neural

(a) `ml`

k	Support	Top-3 words
1	14.06%	test case code
2	7.60%	program python languag
3	6.93%	algorithm class comput
4	6.27%	problem lectur set
5	5.84%	question answer problem

(b) `algo`

k	Support	Top-3 words
1	13.62%	write read good
2	8.91%	write time idea
3	8.34%	write writer stori
4	7.37%	write writer work
5	7.34%	english languag write

(c) `comp`

k	Support	Top-3 words
1	10.26%	shakespear read english
2	9.50%	shakespear play year
3	8.27%	play shakespear charact
4	7.90%	post mooc discuss
5	6.37%	romeo juliet love

(d) `shake`

TABLE II: Summary of the top five topics extracted by LDA for each online discussion forum. For each course, the topics tend to be reasonably disjoint, with the exception of common words.

5% of outliers removed.[1] We show the means and standard deviations (s.d.) of each feature for both groups, as well as the signal-to-noise ratio (SNR) for each feature. The SNR measures how effectively a feature can distinguish between the two groups, with a higher magnitude indicating more efficacy [25]. We make a few observations from these statistics:

(i) Infrequent short paths: The length `Lp` and number `Np` of shortest paths between learners are both negatively associated with link formation. Intuitively, we would expect that learners that are closer together (*e.g.,* with several mutual neighbors) would be more likely to form links, but this result implies that the opposite is the case, perhaps due to many new learners with small `Lp` and `Np` entering conversation threads. An interesting analogy can be drawn here to the small world phenomenon, where all nodes tend to be connected by short paths [7].

(ii) Low-degreed shared neighbors: `Ja`, `Ad` and `Re` are each positively associated with link formation, in order of increasing SNR. Each of these measures the common neighborhood of two learners, with increasing penalty for the degrees of their neighbors (*i.e.,* `Ja` does not include degree at all, while `Re` is inversely proportional to it). The fact that `Re` has the highest SNR, then, implies that shared neighbors with fewer links are more prone to facilitate link formation.

(iii) Topology vs. post properties: `Pr` and `To` are both positively associated with link formation, as one would expect: those with higher degrees (`Pr`) and focusing on similar topics (`To`) should be more likely to interact in the discussions. Surprisingly, though, these features have lower SNRs than the other neighborhood-based features, indicating that the network topology drives link formation in an SLN more than individual learner properties like tendency to post and topic interest. However, we will see in Sec. III-E that `To` alone is actually more useful to our tuned link prediction model than the combination of both path features is.

(iv) Quantitative vs. humanities: `Pr` is higher in `comp` and `shake` (particularly `shake`) than in `ml` and `algo`. This is consistent with humanities courses tending to invite more open-ended discussions, whereas quantitative courses have questions requiring explicit answers [5]. More learners would then be motivated to post in the forums of humanities courses– indeed, such participation may be a course requirement– leading to more links forming. Table I confirms this intuition.

B. Model Evaluation Procedure

To evaluate the models proposed in Sec. II, we use the following training procedure, metrics, and baseline.

Training and testing. Following Sec. III-A, we again consider the link sets $\mathcal{G}(L)$ and $\mathcal{G}^c(L)$. In the kth iteration of training/testing, we draw a random sample $\mathcal{G}_k^c(L)$ of size $|\mathcal{G}(L)|$ from $\mathcal{G}^c(L)$. Then, we randomly select 80% of the links from $\mathcal{G}(L)$ and $\mathcal{G}_k^c(L)$ as the kth training set Ω_k^r, and use the other 20% of both of these sets as the kth test set Ω_k^e. Following [26], this sampling ensures that the prediction algorithms are not biased towards predicting only the links that never form, since the prediction accuracies on the formed and non-formed links count equally towards the overall performance.

In each of the k iterations, we consider each time $i = 1, ..., L$ sequentially. At time i, the model parameters are estimated considering each pair $(u, v) \in \Omega_k^r$, using the procedures in Sec. II-D. Then, for each $(u, v) \in \Omega_k^e$, the inputs are used to make a prediction $\hat{y}_{uv}(i) \in (0, 1)$ of the link state $y_{uv}(i)$. Note that we manually tune the number of hidden dimensions in $\boldsymbol{z}$ for each course, arriving at $N = 10, 12, 15$, and 10 for `ml`, `algo`, `comp`, and `shake` respectively.

Metrics. We use three metrics to evaluate prediction performance. First, we compute the overall Accuracy (ACC), or the fraction of predictions over all time that are correct. For iteration k, it is obtained as:

$$\frac{1}{|\Omega_k^e| \cdot L} \sum_{(u,v) \in \Omega_k^e} \sum_{i=1}^{L} \mathbb{1}\{y_{uv}(i) = \tilde{y}_{uv}(i)\}$$

where $\tilde{y}_{uv}(i) \in \{0, 1\}$ is the binary prediction made based on $\hat{y}_{uv}(i)$ and $\mathbb{1}$ is the indicator function. Second, we compute the Area Under the ROC Curve (AUC), which assesses the tradeoff between true and false positive rates for a classifier [4]. Third, we define a metric called Time Accuracy (TAC) to be the fraction of links that are predicted to form within a fixed window w of when they actually form (among those that eventually form). Letting $n_{uv} = \min_i\{y_{uv}(i) = 1\}$ be the actual time at which link $(u, v) \in \Omega_k^f$ forms and $\tilde{n}_{uv} = \min_i\{\tilde{y}_{uv}(i) = 1\}$ the predicted time, the TAC is defined as

$$\frac{1}{|\Omega_k^f|} \sum_{(u,v) \in \Omega_k^f} \mathbb{1}\{|\tilde{n}_{uv} - n_{uv}| \leq w\}$$

for iteration k, where $\Omega_k^f \subset \Omega_k^e$ is the set of links in the test set that will eventually form. We compute the mean and standard deviation of each metric across three evaluation iterations.

Baseline. We include one algorithm as a benchmark for the Bayesian and neural network models. Choosing the feature from Table III most associated with link formation, we follow [9] and turn `Re` into an unsupervised predictor. To do this, we compute `Re` for each $(u, v) \in \Omega_k^e$ at time i, normalize the vector of values to $(0, 1)$, and use this as $\hat{y}_{uv}(i)$.

[1] The variances across the random samplings of un-formed links are omitted because they were less than 1% of the means.

Feature	SNR	Mean	s.d.
Ja	0.3419	0.0373 0.0044	0.0871 0.0092
Ad	0.5188	1.6160 0.0562	2.8910 0.1152
Re	0.5312	0.1146 0.0013	0.2098 0.0034
Pr	0.4497	3582.3132 390.7671	6466.5811 630.5023
Lp	-0.5582	1.8028 2.7439	0.93578 0.7502
Np	-0.2605	4.5765 9.8947	7.3835 13.0303
To	0.4288	1.4720 0.3407	1.9872 0.6550

(a) ml

Feature	SNR	Mean	s.d.
Ja	0.7574	0.2813 0.0090	0.3463 0.0131
Ad	0.8764	8.1937 0.1943	8.8156 0.3115
Re	0.9454	0.3736 0.0056	0.3798 0.0095
Pr	0.5709	7265.8376 1452.8778	7928.5770 2253.0294
Lp	-0.6798	1.4662 2.5208	0.8310 0.7205
Np	-0.3843	2.7604 9.1165	4.4494 12.0888
To	0.2145	0.5486 0.2527	0.7889 0.5909

(b) algo

Feature	SNR	Mean	s.d.
Ja	0.5893	0.0887 0.0016	0.14190 0.0060
Ad	0.8422	2.1076 0.01456	2.4323 0.0530
Re	0.8461	0.2220 0.0005	0.2599 0.0019
Pr	0.5723	1545.4065 71.2912	2464.5056 111.1170
Lp	-0.8219	1.3535 3.1625	1.0024 1.1988
Np	-0.5082	1.3894 6.2459	1.692 7.8643
To	0.4986	2.3198 0.6795	2.4061 0.8841

(c) comp

Feature	SNR	Mean	s.d.
Ja	2.2012	0.7761 0.0221	0.3125 0.02999
Ad	2.2570	45.1399 0.7479	18.5130 1.1560
Re	2.5159	0.8276 0.0143	0.3009 0.02241
Pr	2.07241	89417.4319 2715.8250	37274.8495 4561.2664
Lp	-1.0229	1.0604 2.1431	0.2873 0.7712
Np	-0.6559	1.0530 10.5295	0.4926 13.9558
To	0.3515	1.3447 0.6232	1.2429 0.8095

(d) shake

TABLE III: Summary statistics – SNR, mean and standard deviation (s.d.) – for the network features of the two link groups. The top row for each feature corresponds to formed links ($y_{uv}(L) = 1$), and the bottom to non-formed links ($y_{uv}(L) = 0$). Taken individually, the neighborhood-based features Re and Ad have the strongest correlations with link formation, while the topic-based To tends to have the least.

		ml	algo	comp	shake
Re	AUC	0.7234 ± 0.002	0.7897 ± 0.003	$\mathbf{0.8120 \pm 0.002}$	0.9710 ± 0.001
	ACC	0.5018 ± 0.001	0.5000 ± 0.003	0.5036 ± 0.009	0.5013 ± 0.005
BNet	AUC	0.6107 ± 0.043	$\mathbf{0.8527 \pm 0.006}$	0.5059 ± 0.075	0.9340 ± 0.006
	ACC	0.8813 ± 0.010	0.8521 ± 0.002	0.8589 ± 0.007	0.6804 ± 0.001
NNet	AUC	$\mathbf{0.7368 \pm 0.012}$	0.8343 ± 0.038	0.7102 ± 0.067	$\mathbf{0.9739 \pm 0.003}$
	ACC	$\mathbf{0.9224 \pm 0.005}$	$\mathbf{0.9005 \pm 0.013}$	$\mathbf{0.9277 \pm 0.000}$	$\mathbf{0.9364 \pm 0.023}$

TABLE IV: Performance of the baseline (Re), Bayesian (BNet), and neural network (NNet) models on each dataset, with $L = L_1$. The best algorithm for each course-metric is bolded. The NNet has the best performance in 6 of the 8 cases.

L		Set	ml	algo	comp	shake
L_1	AUC	1 to L	0.7368 ± 0.012	0.8343 ± 0.038	0.7102 ± 0.067	0.9739 ± 0.003
		L	0.7259 ± 0.002	0.7840 ± 0.005	0.8215 ± 0.010	0.9596 ± 0.003
	ACC	1 to L	0.9224 ± 0.005	0.9005 ± 0.013	0.9277 ± 0.000	0.9364 ± 0.022
		L	0.6609 ± 0.021	0.6938 ± 0.036	0.7683 ± 0.006	0.9127 ± 0.012
L_2	AUC	1 to L	0.7343 ± 0.031	0.7901 ± 0.111	0.6328 ± 0.003	0.9731 ± 0.010
		L	0.6838 ± 0.030	0.7790 ± 0.141	0.8139 ± 0.004	0.9688 ± 0.001
	ACC	1 to L	0.9362 ± 0.006	0.8813 ± 0.009	0.9007 ± 0.010	0.9407 ± 0.009
		L	0.5925 ± 0.029	0.7582 ± 0.114	0.5406 ± 0.030	0.9339 ± 0.006

TABLE V: Performance of the time series neural network model on each dataset, for two different numbers of intervals L and two different sets of intervals $i = 1, ..., L$ (*i.e.,* all) vs. $i = L$ (*i.e.,* the final). The less granular L obtains higher performance in the majority of cases, while performance on intervals varies between courses.

C. Performance Evaluation

Table IV gives the overall performance of the baseline, Bayesian, and neural network models in terms of the AUC and ACC metrics (we discuss the TAC metric in Sec. IV-A), for the default number of intervals L_1 in Table I. Overall, we see that *the neural network model outperforms the other predictors* in $6/8$ cases across the metrics and datasets, reaching AUCs between 0.71 and 0.97 and ACCs between 0.90 and 0.94. The ACC of the baseline is nearly random, but it performs better in terms of AUC when averaging across different thresholds; in fact, in terms of AUC, the baseline outperforms the Bayesian model in $3/4$ cases. The relatively low performance of the Bayesian model confirms the hypothesis from Sec. II that *the evolution of the state of an SLN between different time periods is important to predicting learner interactions*, an aspect which the LSTM neural network includes.

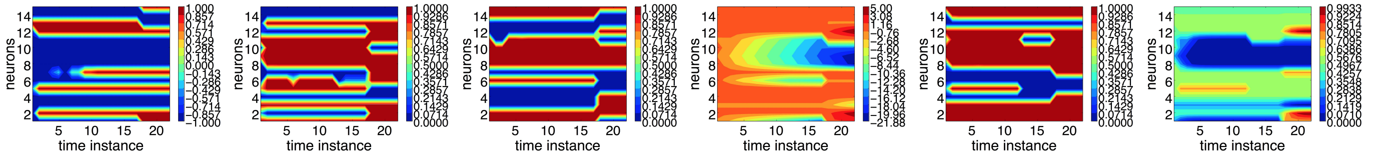
Table V shows how the performance of the neural network model varies based on the number of time intervals L_1 and L_2. In each case, we also show how the metrics vary between considering predictions over all intervals $i = 1, ..., L$ (the default, as in Table IV) versus just considering the final time $i = L$. Notice that *the less granular set of time intervals L_1 obtain higher AUC than L_2 in each case of interval set and metric* (except $i = L$ for shake), with up to a 0.08 increase for comp (on $i = 1, ..., L$). A more granular value, as in L_2, gives the model access to more frequently updated features; while ensuring up-to-date inputs to the predictor, it also requires pinpointing the time of link formation more precisely, leading to worse performance overall.

Notice that for algo, the AUC taken across all intervals in Table V is substantially better than that on the final interval for either case of L, while the opposite is true for comp. This indicates that *links that never form may have more discernible features in the composition course*, given that the AUC improves when considering the question of whether links form or not rather than when they form. This result is consistent with our observations on feature correlations in Sec. III-A: humanities courses tend to have larger learner neighborhoods and more formed links, indicating that unformed links may have particularly distinct characteristics. Like algo, the other quantitative course ml also shows a slight improvement when considering all links, but so does shake. However, shake also has above 0.95 AUC in each case, indicating high overall predictability, consistent with the feature SNR values in Table III having highest magnitude for this course.

D. Time-Series Variable Evolution

We next consider how the LSTM model parameters specified in Sec. II-C evolve over time. By examining the relationship fading gate $\boldsymbol{f}$ in particular, we are able to see how the inputs from time interval $i - 1$ affect the model output at time interval i, *i.e.,* how much information is carried over from interval to interval. To do so, we choose a link $(u, v) \in \mathcal{G}(L)$ at random from comp, and feed $\boldsymbol{e}_{uv}(i)$ into the trained model for $L = 23$ to generate the predictions $\hat{y}_{uv}(i)$. The prediction has high accuracy on the chosen link, which forms within one time interval of when it is predicted to form (around $i = 17$).

The neuron activation values for the gates $\boldsymbol{g}$, $\boldsymbol{i}$, $\boldsymbol{f}$, $\boldsymbol{o}$ and the state $\boldsymbol{z}$ and output $\boldsymbol{h}$ are shown in Fig. 5. The vertical axis is the vector dimension (*i.e.,* neuron number), and the horizontal is the time instance i. A few of the input gate dimensions $\boldsymbol{g}$ change at about the time the link is formed (around $i = 17$). These changes propagate through the network, causing the output $\boldsymbol{h}$ as well as some dimensions of the intermediate gates $\boldsymbol{f}, \boldsymbol{i}$, and $\boldsymbol{o}$ to change around here as well, thus forming an accurate prediction. The fact that $\boldsymbol{i}$ and $\boldsymbol{f}$ tend to take



(a) $\boldsymbol{g}_{uv}(i)$ (b) $\boldsymbol{i}_{uv}(i)$ (c) $\boldsymbol{f}_{uv}(i)$ (d) $\boldsymbol{z}_{uv}(i)$ (e) $\boldsymbol{o}_{uv}(i)$ (f) $\boldsymbol{h}_{uv}(i)$

Fig. 5: Neuron activations of each gate $\boldsymbol{g}$, $\boldsymbol{i}$, $\boldsymbol{f}$, $\boldsymbol{o}$ and the state $\boldsymbol{z}$ and output $\boldsymbol{h}$ over time, for a particular link (u, v) in `comp` on L_2. The fact that several gate dimensions are non-zero indicates that information is propagating across multiple time periods for prediction.

Set		`ml`	`algo`	`comp`	`shake`
Nei + Path	AUC	$\mathbf{0.7162 \pm 0.016}$	0.8241 ± 0.034	0.7043 ± 0.068	$\mathbf{0.9753 \pm 0.002}$
	ACC	$\mathbf{0.9235 \pm 0.004}$	$\mathbf{0.9059 \pm 0.007}$	$\mathbf{0.9273 \pm 0.001}$	0.9364 ± 0.024
Nei + Post	AUC	0.7061 ± 0.020	$\mathbf{0.8322 \pm 0.038}$	$\mathbf{0.7262 \pm 0.058}$	0.9671 ± 0.006
	ACC	$\mathbf{0.9175 \pm 0.010}$	$\mathbf{0.9071 \pm 0.003}$	$\mathbf{0.9238 \pm 0.002}$	0.9323 ± 0.016
Path + Post	AUC	0.3845 ± 0.068	0.4661 ± 0.096	0.3091 ± 0.192	0.8914 ± 0.017
	ACC	0.8913 ± 0.003	0.7907 ± 0.008	0.8705 ± 0.003	0.8191 ± 0.026
Nei	AUC	0.6851 ± 0.025	0.8189 ± 0.032	0.7206 ± 0.058	0.9641 ± 0.008
	ACC	0.9195 ± 0.008	$\mathbf{0.9129 \pm 0.004}$	0.9219 ± 0.002	$\mathbf{0.9400 \pm 0.006}$
Path	AUC	0.2111 ± 0.023	0.3892 ± 0.011	0.2400 ± 0.097	0.8277 ± 0.029
	ACC	0.8888 ± 0.001	0.7817 ± 0.005	0.8705 ± 0.003	0.7137 ± 0.069
Post	AUC	0.3318 ± 0.058	0.4444 ± 0.080	0.3167 ± 0.185	0.6340 ± 0.079
	ACC	0.8919 ± 0.003	0.7888 ± 0.006	0.8705 ± 0.003	0.5618 ± 0.101

(a) L_1

Set		`ml`	`algo`	`comp`	`shake`
Nei + Path	AUC	0.6906 ± 0.065	$\mathbf{0.7846 \pm 0.112}$	0.5939 ± 0.020	$\mathbf{0.9715 \pm 0.012}$
	ACC	$\mathbf{0.9358 \pm 0.003}$	$\mathbf{0.8943 \pm 0.028}$	0.9015 ± 0.011	$\mathbf{0.9402 \pm 0.009}$
Nei + Post	AUC	$\mathbf{0.7312 \pm 0.074}$	0.7660 ± 0.123	$\mathbf{0.6392 \pm 0.003}$	0.9606 ± 0.009
	ACC	$\mathbf{0.9364 \pm 0.004}$	0.8312 ± 0.061	$\mathbf{0.9027 \pm 0.011}$	0.9355 ± 0.013
Path + Post	AUC	0.4354 ± 0.067	0.4980 ± 0.498	0.1860 ± 0.010	0.5707 ± 0.171
	ACC	0.9204 ± 0.003	0.7959 ± 0.039	0.8885 ± 0.001	0.6353 ± 0.055
Nei	AUC	0.6952 ± 0.101	0.7607 ± 0.130	0.6005 ± 0.018	0.9574 ± 0.010
	ACC	0.9356 ± 0.002	0.8408 ± 0.049	$\mathbf{0.9036 \pm 0.012}$	0.9344 ± 0.014
Path	AUC	0.2890 ± 0.111	0.5536 ± 0.116	0.1768 ± 0.006	0.4745 ± 0.060
	ACC	0.9187 ± 0.001	0.7294 ± 0.062	0.8885 ± 0.001	0.6001 ± 0.012
Post	AUC	0.3576 ± 0.094	0.4512 ± 0.020	0.1917 ± 0.013	0.4627 ± 0.024
	ACC	0.9199 ± 0.002	0.7648 ± 0.152	0.8885 ± 0.001	0.5886 ± 0.004

(b) L_2

TABLE VI: Performance of the LSTM model with selected input feature groups on L_1 and L_2. The 1-2 highest performing groups for each course-metric are bolded. The combinations of *Nei + Post* and *Nei + Path* outperform the other feature combinations, indicating that while the neighborhood-based features are most important for prediction, the other feature types contribute significant information as well.

extreme values indicates that the input $\boldsymbol{g}$ and prior state $\boldsymbol{z}$ components are either fully passed or blocked.

We also observe that several dimensions in $\boldsymbol{z}$ evolve gradually over time, with several non-zero dimensions in $\boldsymbol{f}$ passing information across multiple time periods. This result explains why the neural network performs better than the Bayesian model: passing information from one time interval to another increases the prediction quality compared to only updating the input features at each time.

E. Feature Importance Analysis

Recall in Sec. II-B that we define three groups of features: (i) Neighborhood-based $\boldsymbol{b}$ (*Nei*), which quantify the overlap between learner neighborhoods, (ii) Path-based $\boldsymbol{a}$ (*Path*), which are the length and number of shortest paths, and (iii) Post-based c (*Post*), or the similarity in what learners discuss. To complement the correlation analysis in Table III that was done for each feature individually, we analyze the contribution of each feature type to the LSTM prediction quality by evaluating it using different input feature combinations.

Table VI shows the results for L_1 and L_2 time periods. None of the combinations reach the performance of the original model with all input variables in Table IV, indicating that each feature group contributes to the prediction quality. The *Nei* + *Path* and *Nei* + *Post* combinations show the highest overall performance across all four forums, though, indicating that the *Nei* features contribute the most, consistent with them having the highest SNRs in Table III.

If we compare the individual feature groups, we generally find that the *Nei* features perform the best, followed by *Post*, and then *Path*. This ordering of *Post* and *Path* is opposite of the SNR magnitudes from Table III: here, the single feature `To` outperforms the combined impact of *Path*. Given that Table III is concerned with the eventual formation of links but not the time at which they form, we conjecture that in the absence of *Nei* features, *Post* is more important to pinpointing the time of link formation while *Path* is more important to whether they form at all. This makes sense considering that the set of topics covered in the course will evolve over time.

IV. Recommending Link Formation

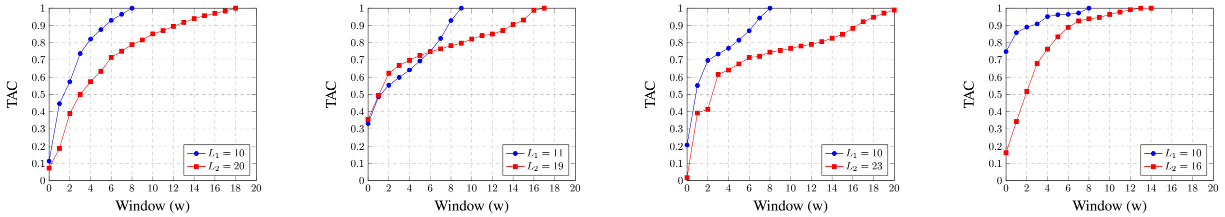
Predicting interactions in an SLN can improve learners' forum experiences in several ways. In this section, we consider the application of our methodology to SLN link recommendation. We first show that our input features can also be used to recommend early formation and eventual reconnection of links (Sec. IV-A), and then explore their relationship with the quality of posts (Sec. IV-B).

A. Early Detection of Link Formation

While our evaluation in Sec. III-C considers the ability of the models to predict link formation in the next time interval, it does not consider links that will form earlier or later. These cases, however, can be of import to learners: if we can predict in advance which learners may form connections, then we can encourage them to connect sooner, which may lead to faster replies from learners expected to have delayed responses. On the other hand, if we find that a link forms much sooner than our prediction expects, this may indicate that the learners would benefit from re-connecting on the current topic.

To study these cases, we evaluate the TAC metric from Sec. III-B for our neural network model, *i.e.,* we measure whether links form within a given window w of when they are predicted to. Fig. 6 shows the TAC values as w is increased from 0 to L; while the L_1 TACs are generally to the left of the L_2 TACs, much of this variation can be explained by the L_1 time intervals being approximately twice as long as L_2 ones. Since the TAC is only defined for links that do eventually form, it always reaches 1 for sufficiently high w.

The sharp increase of each TAC curve for small w indicates that many links form close to when they are predicted to



(a) ml (b) algo (c) comp (d) shake

Fig. 6: TAC with different windows w. The TAC curves all exhibit sharp increases initially, indicating many links form around the time they are predicted to. The links at higher w, on the other hand, indicate potential for recommending early link formation and future reconnection.

form, reinforcing our observations of model quality from other performance metrics in Sec. III-C. A window of $w = 2$ on L_1, for example, is already sufficient for all four forums to reach a TAC of 0.5 or above. On the other hand, *there are still a substantial number of links with large* w, which implies that these predictions present a significant opportunity to recommend early formation of links (when predictions are early) and potential times for learners to reconnect (when predictions are late). Though there is less room for change on links with smaller w, learners may be more willing to act on recommendations in these cases since they induce less change to actual behavior [5]; after all, a learner may be reluctant to reach out to others on the basis of outdated threads or on the assumption that they will eventually collaborate.

B. Analyzing Post Response Quality

Learners' experiences in online forums are affected by not only the speed of replies, but also by whether the replies actually answer their questions and/or facilitate further discussion. To this end, we now analyze how the SLN features in our model are associated with the quality of post replies, to see whether the recommendations based on the method in Sec. IV-A would be consistent with higher quality links.

Since "quality" can be subjective, we use two different possible measures. First, we consider the *votes* $s_{v \to u}$ that learner v has amassed on all responses v made to posts of learner u. By considering the net votes (up-votes minus down-votes), we capture other learners' judgements of the response quality. Second, we consider the *length* $l_{v \to u}$ in words of the replies from v to u, following proxies of quality in other works [5]. To obtain $s_{v \to u}$ and $l_{v \to u}$, let $\mathcal{P}_v$ and $\mathcal{P}_u$ be the set of posts made by v and u, and let $\mathcal{T}(n)$ be the thread in which post n appears. We obtain $\mathcal{P}_{v \to u} = \{p_n \in \mathcal{P}_v : \exists p_m : p_m \in \mathcal{P}_u, p_m \in \mathcal{T}(n), t_n > t_m\}$, *i.e.,* the subset of posts made by v that occur in the same thread after (*i.e.,* responding to) a post p_m by u. Then, $l_{v \to u} = \sum_{p_n \in \mathcal{P}_{v \to u}} |\boldsymbol{x}_n|$ and $s_{v \to u} = \sum_{p_n \in \mathcal{P}_{v \to u}} s_n$, where s_n is the net votes on p_n.

Feature	ml	algo	comp	shake
Ja	-1.201^*	-0.100^*	-0.914^*	-1.292^*
Ad	0.082^*	-0.017^*	-0.019^*	-0.013^*
Re	-0.358^*	0.226^*	1.440^*	2.906^*
Pr	0.00001^*	0.000005^*	0.00006^*	-0.00006^*
Np	0.0002^*	0.0033^*	0.002^*	0.004^*
Lp	0.083^*	-0.093^*	0.027^*	0.255^*
To	-0.043^*	-0.0002^*	-0.047^*	0.061^*
Intercept	1.148	0.898	0.139	0.265

(a) Votes $s_{v \to u}$

Feature	ml	algo	comp	shake
Ja	-85.594^*	-17.150^*	-440.668^*	-334.123^*
Ad	-2.431^*	-6.003^*	-17.814^*	3.745^*
Re	84.003^*	38.841^*	549.699^*	637.452^*
Pr	0.0014^*	0.0003^*	0.034^*	-0.003^*
Np	-0.050^*	0.325^*	1.009^*	2.048^*
Lp	8.156^*	2.310^*	16.556^*	51.308^*
To	1.801^*	17.349^*	-0.697^*	13.159^*
Intercept	130.364	135.583	129.787	114.544

(b) Length $l_{v \to u}$

TABLE VII: Regression results between the SLN features and the two measures of quality. A * indicates a significance of $p < 0.01$. For each feature, the direction of correlation tends to be consistent between quality measures, but contrasting to those for link formation in Table III.

We perform two linear regressions across $(u, v) \in \Omega$ with the features $\boldsymbol{e}_{uv}$ as independent variables and $s_{v \to u}$ and $l_{v \to u}$ as the targets. Table VII shows the resulting coefficients; they are all statistically significant with p-values < 0.01.[2] Note that, in each course except for ml, each feature has the same direction of correlation with respect to the two quality measures. For instance, when a learner u posts, those learners v that have higher Re with u tend to provide longer answers that receive more votes. This strong, positive correlation with Re is also consistent with the SNR values for link formation in Table III. Similarly, the topic feature To is in most cases positively correlated with both post quality and link formation.

For the other features, however, *we find opposite trends between feature associations in Table VII and Table III.* The other neighborhood-based features Ja and Ad tend to be negatively correlated with both quality metrics, whereas they are positive predictors of link formation. This difference with Re indicates that when common neighbors themselves have smaller degrees, this is more likely to facilitate higher quality links, whereas neighbors with high degrees may encourage more, lower quality responses. Also, while path-based features Lp and Np are negatively associated with link formation, they are positively associated with quality: learners who are closer together in the SLN (lower Lp) are more likely to respond to each other, but the highest quality answers have a tendency to come from learners further away in the network.

Overall, these results indicate that *response time and quality are two competing objectives for a link recommendation system.* Such a system, then, may include two separate predictors: the one developed in Sec. II that can lead to enhanced response times, and another that predicts the quality of responses across learners to encourage the participation of experts.

[2] We omit the R^2 values because our purpose here is to analyze feature associations rather than to build a predictor of post quality.

V. Conclusion and Future Work

In this paper, we developed a time-series methodology for predicting link formation in a Social *Learning* Network (SLN). Our algorithm uses neighborhood-based, path-based, and post-based quantities between learners as modeling features. Through evaluation on four discussion forums from Massive Open Online Courses (MOOCs), we demonstrated that our neural network-based model obtains superior quality over both a Bayesian model and an unsupervised baseline, indicating that passing model state between time periods is critical for the link prediction problem in SLN. By examining the contribution of each type of input feature, we also confirmed that while the neighborhood-based features are most important, they each contribute significantly to model quality.

While our work establishes an initial framework and results for link prediction in SLNs, many avenues remain for exploring the challenges of link prediction in this new type of online social network. One is additional feature engineering: other features that we did not consider–such as learners' background knowledge, level of education, and post frequency–may also be associated with link formation, and may allow further improvements in link prediction quality. Another is additional evaluation variants: the results found here can be compared with other types of time series predictors and other types of SLN, *e.g.,* those on Q&A sites. Last but not least is forum implementation: we showed how our method can be used as the basis for a link recommendation system to improve learner experiences, but our analyses indicated that such a system would benefit from an additional predictor for post quality.

References

[1] T.-Y. Yang, C. G. Brinton, C. Joe-Wong, and M. Chiang, "Behavior-Based Grade Prediction for MOOCs via Time Series Neural Networks," *IEEE Journal of Selected Topics in Signal Processing*, 2017.

[2] C. G. Brinton and M. Chiang, "Social Learning Networks: A Brief Survey," in *CISS*. IEEE, 2014, pp. 1–6.

[3] L. F. Pendry and J. Salvatore, "Individual and Social Benefits of Online Discussion Forums," *IEEE Trans. Learning Technol.*, vol. 50, pp. 211–220, 2015.

[4] C. G. Brinton and M. Chiang, "MOOC Performance Prediction via Clickstream Data and Social Learning Networks," in *INFOCOM*. IEEE, 2015, pp. 2299–2307.

[5] C. G. Brinton, S. Buccapatnam, F. M. F. Wong, M. Chiang, and H. V. Poor, "Social learning networks: Efficiency optimization for mooc forums," in *Proc. of IEEE INFOCOM*. IEEE, 2016, pp. 1–9.

[6] M. Al Hasan and M. J. Zaki, "A Survey of Link Prediction in Social Networks," in *Social Network Data Analytics*. Springer, 2011, pp. 243–275.

[7] D. Liben-Nowell and J. Kleinberg, "The Link-Prediction Problem for Social Networks," *Journal of the Association for Information Science and Technology*, vol. 58, no. 7, pp. 1019–1031, 2007.

[8] A. Clauset, C. Moore, and M. Newman, "Hierarchical Structure and the Prediction of Missing Links in Networks," *Nature*, vol. 452, no. 7191, pp. 98–101, 2008.

[9] L. Backstrom and J. Leskovec, "Supervised Random Walks: Predicting and Recommending Links in Social Networks," in *WSDM*. ACM, 2011, pp. 635–644.

[10] S. Scellato, A. Noulas, and C. Mascolo, "Exploiting Place Features in Link Prediction on Location-based Social Networks," in *SIGKDD*. ACM, 2011, pp. 1046–1054.

[11] Y. Dong, J. Tang, S. Wu, J. Tian, N. V. Chawla, J. Rao, and H. Cao, "Link Prediction and Recommendation across Heterogeneous Social Networks," in *ICDM*. IEEE, 2012, pp. 181–190.

[12] R. Xiang, J. Neville, and M. Rogati, "Modeling Relationship Strength in Online Social Networks," in *WWW*. ACM, 2010, pp. 981–990.

[13] I. Kahanda and J. Neville, "Using Transactional Information to Predict Link Strength in Online Social Networks," in *ICWSM*, 2009, pp. 74–81.

[14] Z. Huang and D. K. Lin, "The Time-Series Link Prediction Problem with Applications in Communication Surveillance," *Journal on Computing*, vol. 21, no. 2, pp. 286–303, 2009.

[15] J. Leskovec, D. Huttenlocher, and J. Kleinberg, "Predicting Positive and Negative Links in Online Social Networks," in *WWW*. ACM, 2010, pp. 641–650.

[16] J. Tang, T. Lou, and J. Kleinberg, "Inferring Social Ties Across Heterogenous Networks," in *WSDM*. ACM, 2012, pp. 743–752.

[17] J. Qiu, J. Tang, T. X. Liu, J. Gong, C. Zhang, Q. Zhang, and Y. Xue, "Modeling and predicting learning behavior in moocs," in *Proc. of ACM WSDM*. ACM, 2016, pp. 93–102.

[18] F. M. F. Wong, Z. Liu, and M. Chiang, "On the Efficiency of Social Recommender Networks," *IEEE/ACM Transactions on Networking*, vol. 24, pp. 2512–2524, 2016.

[19] J. Cao, H. Gao, L. E. Li, and B. Friedman, "Enterprise Social Network Analysis and Modeling: A Tale of two Graphs," in *INFOCOM*. IEEE, 2013, pp. 2382–2390.

[20] Z. C. Lipton, J. Berkowitz, and C. Elkan, "Critical Review of Recurrent Neural Networks for Sequence Learning," *arXiv:1506.00019*, 2015.

[21] Technical report. https://1drv.ms/f/s!AkD1QHtGlOY4as3OA6Qzwv4AfF0.

[22] P. J. Werbos, "Backpropagation Through Time: What it Does and How to do it," *Proceedings of the IEEE*, vol. 78, no. 10, pp. 1550–1560, 1990.

[23] W. Zaremba, I. Sutskever, and O. Vinyals, "Recurrent Neural Network Regularization," *arXiv:1409.2329*, 2014.

[24] D. M. Blei, A. Y. Ng, and M. I. Jordan, "Latent Dirichlet Allocation," *JMLR*, vol. 3, no. 3, pp. 993–1022, 2003.

[25] S. Y. Kung, *Kernel Methods and Machine Learning*. Cambridge University Press, 2014.

[26] M. Fire, L. Tenenboim, O. Lesser, R. Puzis, L. Rokach, and Y. Elovici, "Link Prediction in Social Networks using Computationally Efficient Topological Features," in *SocialCom*. IEEE, 2011, pp. 73–80.