Abstract

We formulate a new problem at the intersection of semi-supervised learning and contextual bandits, motivated by several applications including clinical trials and ad recommendations. We demonstrate how Graph Convolutional Network (GCN), a semi-supervised learning approach, can be adjusted to the new problem formulation. We also propose a variant of the linear contextual bandit with semi-supervised missing rewards imputation. We then take the best of both approaches to develop multi-GCN embedded contextual bandit. Our algorithms are verified on several real world datasets.

1 Introduction

We formulate the problem of Online Partially Rewarded (OPR) learning. Our problem is a synthesis of the challenges often considered in the semi-supervised and contextual bandit literature. Despite a broad range of practical cases, we are not aware of any prior work addressing each of the corresponding components. Next we justify each of the keywords and give motivating examples.

- **Online**: data is often naturally collected over time and systems are required to make predictions (take an action) before they are allowed to observe any response from the environment.

- **Partially**: oftentimes there is no response available, e.g. a missing label or environment not responding to system's action.

- **Rewarded**: in the context of online (multiclass) supervised learning we assume that the environment will provide the true label - however in many practical systems we can only hope to observe feedback indicating whether our prediction is good or bad (1 or 0 reward), the latter case obscuring the true label for learning.

Practical scenarios that fall under the umbrella of OPR range from clinical trials to dialog orchestration. In clinical trials, reward is partial, as patients may not return for followup evaluation. When patients do return, if feedback on their treatment is negative, the best treatment, or true label, remains unknown and the only available information is a reward of 0 for the treatment administered. In dialog systems, a user's query is often directed to a number of domain specific agents and the best response is returned. If the user provides negative feedback to the returned response, the best available response is uncertain and moreover, users can choose to not provide feedback at all.

In many applications, obtaining labeled data requires a human expert or expensive experimentation, while unlabeled data may be cheaply collected in abundance. Learning from unlabeled observations to improve prediction performance is the key challenge of semi-supervised learning [Chapelle et al., 2009]. One of the possible approaches is the continuity assumption, i.e. points closer to each other in the feature space are more likely to share a label [Seeger, 2000]. When the data has a graph structure, another approach is to perform node classification using graph Laplacian regularization, i.e. penalizing difference in outputs of the connected nodes [Zhu et al., 2003]. The latter approach can also be applied without the graph under the continuity assumption by building similarity based graph. We note that the problem of online semi-supervised leaning is rarely considered, with few exceptions [Yver, 2009; Valko et al., 2012; Bouneffouf et al., 2020b]. In our setting, the problem is further complicated by the bandit-like feedback in place of labels, rendering the existing semi-supervised approaches inapplicable. We will however demonstrate how one of the recent approaches, Graph Convolutional Networks (GCN) [Kipf and Welling, 2016], can be extended to our setting.

The multi-armed bandit problem provides a solution to the exploration versus exploitation trade-off, informing a player how to pick within a finite set of decisions while maximizing cumulative reward in an online learning setting. Optimal solutions have been developed for a variety of problem formulations [Auer et al., 2002a; Auer et al., 2002b; Allesiardo et al., 2014; Bouneffouf and Féraud, 2016; Balakrishnan et al., 2018; Bouneffouf et al., 2019; Lin et al., 2018; Bouneffouf et al., 2017a; Lin et al., 2020; Bouneffouf et al., 2017b]. These approaches do not take into account the relationship between context and reward, potentially inhibiting overall performance. In Linear Upper Confidence Bound (LINUCB) [Li et al., 2010; Chu et al., 2011; Bouneffouf and Rish, 2019; Bouneffouf et al., 2020a; Bouneffouf, 2020] and in Contextual Thompson Sampling (CTS) [Agrawal and Goyal, 2013], the authors assume a lin-
ear dependency between the expected reward of an action and its context; the representation space is modeled using a set of linear predictors. These algorithms assume that the bandit can observe the reward at each iteration, which is not the case in our setting. Several authors have considered variations of partial/corrupted rewards [Bartók et al., 2014; Gajane et al., 2016], however the case of entirely missing rewards has not been studied to the best of our knowledge.

In this paper we focus on handling the problem of online semi-supervised learning with bandit feedback. We first review some existing methods in the respective domains and propose extensions to each of them to accommodate our problem setup. Then we proceed to combine the strengths of both approaches to arrive at an algorithm well suited for the Online Partially Rewarded learning as demonstrated with experiments on several real datasets.

2 Preliminaries

In this section we review two approaches coming from the respective domains of semi-supervised learning and contextual bandits, emphasizing their relevance and shortcomings in solving the OPR problem.

2.1 Graph Convolutional Networks

Neural networks have proven to be powerful feature learners when classical linear approaches fail. Classical neural network, Multi Layer Perceptron (MLP), is dramatically over-parametrized and requires copious amounts of labeled data to learn. On the other hand, Convolutional Neural Networks are more effective in the image domain [Krizhevsky et al., 2012], partially due to parameter sharing exploiting relationships between pixels. Image structure can be viewed as a grid graph where neighboring pixels are connected nodes. This perspective and the success of CNNs inspired the development of convolution on graphs neural networks [Henaff et al., 2015; Defferrard et al., 2016; Bronstein et al., 2017] based on the concept of graph convolutions known in the signal processing communities [Shuman et al., 2013]. Though all these works are in the realm of classical supervised learning, the idea of convolving signal over graph nodes is also widely applied in semi-supervised (node classification) learning [Belkin et al., 2006], where the graph describes relationships among observations (cf. grid graph of features (pixels) in CNNs). [Kipf and Welling, 2016] proposed Graph Convolutional Network (GCN), an elegant synthesis of convolution on graphs ideas and neural network feature learning capability, which significantly outperformed prior semi-supervised learning approaches on several citation networks and knowledge graph datasets.

To understand the GCN method, let $X \in \mathbb{R}^{T \times D}$ denote a data matrix with $T$ observations and $D$ features and let $A$ denote a positive, sparse, and symmetric adjacency matrix $A$ of size $T \times T$. The GCN embedding of the data with one hidden layer of size $L$ is $g(X) = \hat{A} \text{ReLU}(\hat{A}XW) \in \mathbb{R}^{T \times L}$, where $\hat{A}$ is degree normalized adjacency with self connections: $\hat{A} = (D + \mathcal{I}_T)^{-1/2}(A + \mathcal{I}_T)(D + \mathcal{I}_T)^{-1/2}$ and $\mathcal{D}_{ii} = \sum_{j=1}^{T} A_{ij}$ is the diagonal matrix of node degrees. $W \in \mathbb{R}^{D \times L}$ is a trainable weight vector. Resulting embedded data goes into the softmax layer and the loss for backpropagation is computed only on the labeled observations. The product $\hat{A}X$ gives the one-hop convolution — signal from a node is summed with signals from all of its neighbours achieving smooth transitions of the embeddings $g(X)$ over the data graph. Although a powerful semi-supervised approach, GCN is not suitable for the Online and Rewarded components of OPR. It additionally requires a graph as an input, which may not be available in some cases.

2.2 Contextual Bandit

Following [Langford and Zhang, 2008], the contextual bandit problem is defined as follows. At each time $t \in \{1, ..., T\}$, a player is presented with a context vector $x_t \in \mathbb{R}^D$, $\|x_t\| \leq 1$ and must choose an arm $k \in \{1, ..., K\}$. $r_{t,k} \in [0,1]$ is the reward of the action $k$ at time $t$, and $r_t \in [0,1]^K$ denotes a vector of rewards for all arms at time $t$. We operate under the linear realizability assumption, i.e., there exist unknown weight vectors $\theta_k \in \mathbb{R}^D$ with $\|\theta_k\|_2 \leq 1$ for $k = 1, \ldots, K$ so that

$$\forall k, t : E[r_{t,k}|x_t] = \theta_k^T x_t$$

Hence, the $r_{t,k}$ are independent random variables with expectation $\theta_k^T x_t$.

One solution to the contextual bandit problem is the LINUCB algorithm [Li et al., 2010] where the key idea is to apply online ridge regression to incoming data to obtain an estimate of the coefficients $\hat{\theta}_k$ for $k = 1, \ldots, K$. At each step $t$, the LINUCB policy selects the arm with the highest upper confidence bound of the reward $k(t) = \text{argmax}_k(\mu_k + \sigma_k)$, where $\mu_k = \hat{\theta}_k^T x_t$ is the expected reward for arm $k$, $\sigma_k = \alpha \sqrt{x_t^T A_k^{-1} x_t}$ is the standard deviation of the corresponding reward scaled by exploration-exploitation trade-off parameter $\alpha$ (chosen a priori) and $A_k$ is the covariance of the $k$-th arm context. LINUCB requires a reward for the chosen arm, $r_{t,k(t)}$, to be observed to perform its updates. In our setting reward may not be available at every step $t$, hence we need to adjust the LINUCB algorithm to learn from data with missing rewards.

3 Proposed algorithms

In this section we formally define Online Partially Rewarded (OPR) problem and present a series of algorithms, starting with natural modifications of GCN and LINUCB to suit the OPR problem setting and conclude with an algorithm building on strengths of both GCN and LINUCB.

3.1 Problem setting

We now formally define each of the OPR keywords:

- **Online**: at each step $t = 1, \ldots, T$ we observe observation $x_t$ and seek to predict its label $\hat{y}_t$ using $x_t$ and possibly any information we had obtained prior to step $t$.

- **Partially**: after we made the prediction $\hat{y}_t$, environment may not provide any feedback (we will use -1 to encode absence of feedback) and we have to proceed to step $t+1$ without knowledge of the true $y_t$. 
that there is no dependence on \(X\) shall see, there is a natural path from GCN to ROGCN. Sup-
warded Online GCN (ROGCN): (i) online learning; (ii) the
There are three challenges to be addressed to formulate Re-

Note on absence of environment response. We assume that there is no dependence on \(x_t\) in whether environment will respond or not. This is a common setting in semi-supervised learning [Chapelle et al., 2009] — we have access to limited samples from the joint distribution of data and labels \(\mathbb{P}(x,y)\) and larger amount of samples from the data marginal \(\mathbb{P}(x)\) with the goal to infer \(\mathbb{P}(y|x)\) using both. This assumptions is justified in some applications of interest, e.g. whether user will provide feedback to the dialog agent is independent of what the user asked.

3.2 Rewarded Online GCN

There are three challenges to be addressed to formulate Rewarded Online GCN (ROGCN): (i) online learning; (ii) the environment only responds with 0 or 1 to our predictions and (iii) the potential absence of graph information. As we shall see, there is a natural path from GCN to ROGCN. Suppose there is a small portion of data and labels available from the start, \(X_0 \in \mathbb{R}^{D \times T_0}\) and \(y_0 \in \{-1, 1, \ldots, K\}^{T_0}\), where \(D\) is the number of features, \(K\) is the number of classes and \(T_0\) is the size of initially available data. When there is no graph available we can construct a \(k\)-NN graph \((k\) is a parameter chosen a priori) based on similarities between observations - this approach is common in convolutional neural networks on feature graphs [Henaff et al., 2015; Defferrard et al., 2016] and we adopt it here for graph construction between observations \(X_0\) to obtain graph adjacency \(A_0\). We provide details in Section 4. Now that we have \(X_0, y_0, A_0\), we can train GCN with \(L\) hidden units (a parameter chosen a priori) to obtain initial estimates of hidden layer weights \(W_1 \in \mathbb{R}^{D \times L}\) and softmax weights \(W_2 \in \mathbb{R}^{L \times K}\). Next we start to observe the stream of data — as new observation \(x_t\) arrives, we add it to the graph and data matrix, and append -1 (missing label) to \(y\). Then we run additional training steps of GCN and output a prediction to obtain environment response \(h_t \in \{-1, 0, 1\}\). Here 1 indicates correct prediction, hence we include it to the set of available labels for future predictions; 0 indicates wrong prediction and -1 an absence of a response, in the later two cases we continue to treat the label of \(x_t\) as missing. This procedure is summarized in Algorithm 1.

3.3 Bounded Imputation LINUCB

Contextual multi-armed bandits offer a powerful approach to online learning when true labels are not available and the environment’s response to a prediction is observed at every observation instead. However, in our OPR problem setting, the environment may not respond to the agent for every observation. Classic bandit approach such as Linear Upper Confidence Bound (LINUCB) [Li et al., 2010] may be directly applied to OPR, however it would not be able to learn from observations without environment response. We propose to combine LINUCB with a user defined imputation mechanism for the reward when environment response is missing. In order to be robust to variations in the imputation quality, we only allow imputed reward to vary within agent’s beliefs. To make use of the context in the absence of the reward, we consider a user defined imputation mechanism \(I(\cdot): \mathbb{R}^D \rightarrow \Delta^{K-1}\), which is expected to produce class probabilities for an input context \(x_t\) to impute the missing reward. Typically any imputation mechanism will have an error of its own, hence we constrain the imputed reward to be within one standard deviation of the expected reward for the chosen arm:

\[
 r_t(k, x_t) = \max(\mu_k - \sigma_k, \min(I(x_t)_k, \mu_k + \sigma_k)).
\]  

As in ROGCN, we can take advantage of small portion of data to initialize bandit parameters \(b_k = \sum_{t:y_t=k} x_t\) and \(A_k = \sum_{t:y_t=k} x_t x_t^\top\) for \(k = 1, \ldots, K\). Bounded Imputation LINUCB (BILINUCB) is summarized in Algorithm 2.
### 3.4 Multi-GCN embedded UCB

We have presented two algorithms for OPR learning, however both approaches pose some limitations: ROGCN is unable to learn from misclassified observations and has to treat them as missing labels, while BILINUCB assumes linear relationship between data features and labels and even with perfect imputation is limited by the performance of the best linear classifier. Note that the bandit perspective allows one to learn from misclassified observations, i.e. when the environment response $h_t = 0$, and the neural network perspective facilitates learning better features such that linear classifier is sufficient. This observation motivates us to develop a more sophisticated synthesis of GCN and LINUCB approaches, where we can combine advantages of both perspectives.

To begin, we note that if $K = 2$, a $h_t = 0$ environment response identifies the correct class, hence the OP reduces to online semi-supervised learning for which GCN can be trivially updated using ideas from ROGCN. To take advantage of this for $K > 2$ we can consider a suite of GCNs for each of the classes, which then necessitates a procedure to decide which of the GCNs to use for prediction at each step. We propose to use a suite of class specific GCNs, where prediction is made using contextual bandit with context of $k$-th arm coming from the hidden layer representation of $k$-th class GCN and, when missing, reward is imputed from the corresponding GCN.

We now describe the multi-GCN embedded Upper Confidence Bound (GCNUCB) bandit in more details. Let $g(X)^{(k)} = \hat{A} \text{ReLU}(\hat{A}XW^{(k)}_1)$ denote the $k$-th GCN data embedding and let $g(X)^{(k)}_t$ denote the embedding of observation $x_t$. We will use this embedding (additionally normalized to unit $l_2$ norm) as context for the corresponding arm of the contextual bandit. The advantage of this embedding is its graph convolutional nature coupled with expressive power of neural networks. We note that as we add new observation $x_{t+1}$ to the graph and update weights of the GCNs, the embedding of the previously observed $x_1, \ldots, x_t$ evolves. Therefore instead of dynamically updating bandit parameters $b_k$ and $A_k$ as it was done in BILINUCB, we maintain set of indices for each of the arms $C_k = \{ t : \hat{y}_t = k \text{ or } h_t = 1 \}$. At any step we can compute corresponding bandit context covariance and weight estimate using current embedding:

$$A_k = \sum_{t \in C_k} g(X)^{(k)}_t g(X)^{(k)\top}_t$$

$$\theta_k = A^{-1}_k \sum_{t \in C_k} r_{t,k} g(X)^{(k)}_t, \quad \theta_k = \theta_k / \| \theta_k \|_2$$

where $r_{t,k}$ is the reward that was observed or imputed at step $t$ for arm $k$ (recall that we are imputing using prediction of the binary GCN corresponding to the arm chosen by the bandit). Now we can compute expected value and standard deviation for the reward on each arm. The prediction is made based on the upper confidence bounds for the rewards of the arms:

$$\mu_k = \theta_k^\top g(X)^{(k)}_t$$

$$\sigma_k = \alpha \sqrt{g(X)^{(k)}_t \hat{A}^{-1}_k g(X)^{(k)}_t}$$

$$\hat{y}_t = \arg\max_k (\mu_k + \sigma_k).$$

Then we observe the environment response $h_t \in \{-1, 0, 1\}$. Unlike ROGCN, GCNUCB is able to learn from mistakes, i.e. when $h_t = 0$ — although as before we don’t know the true class, we can be sure that it was not $\hat{y}_t$, hence we can use this information to improve GCN corresponding to the class $\hat{y}_t$. We summarize GCNUCB in Algorithm 3. Similarly to ROGCN and BILINUCB we can use a small amount of data $X_0$ and labels $y_0$ converted to binary labels $y_0(k) \in \{-1, 0, 1\}^{T_0}$ (as before -1 encodes missing label) for each class $k$ to initialize GCNs weights $W^{(k)}_1, W^{(k)}_2$ for $k = 1, \ldots, K$ and index sets $C_k$ for each of the arms $k = 1, \ldots, K$. Adjacency matrix if not given is obtained as in ROGCN.

#### Algorithm 3 GCNUCB

1: Input: $W^{(k)}_1, W^{(k)}_2, C_k, r_{.,k}, y_0 \forall k, X_0, \hat{A}, 0$
2: Set $y^{(k)} = y_0(k) = 1, \ldots, K, X = X_0, \hat{A} = \hat{A}_0$
3: for $t = T_0 + 1$ to $T$
4:  Append $x_t$ to $X$, -1 to each of $y^{(1)}, \ldots, y^{(K)}$
5:  Update $\hat{A}$ with new edges if graph information is available or build $k$-NN similarity graph from $X$ to obtain $\hat{A}$
6:  Update $W^{(k)}_1$ and $W^{(k)}_2$ through GCN backpropagation with inputs $X, \hat{A}, y^{(k)}$ for $k = 1, \ldots, K$
7:  Retrieve embeddings $g(X)^{(k)}_t \forall k$
8:  Compute $A_k$ (Eq. (2)) and $\theta_k$ (Eq. (3)) $\forall k$
9:  Make prediction $\hat{y}_t$ using Eq. (4) and observe environment response $h_t$
10: if $h_t = 1$ then
11:  For each $k$ replace last entry of $y^{(k)}$ with $1$ if $\hat{y}_t = k$ and 0 otherwise
12:  Append $\hat{y}_t$ to each $C_k$ and 1 to $r_{.,k}$ if $\hat{y}_t = k$ and 0 otherwise
13: else if $h_t = 0$ (learning from mistakes) then
14:  Replace last entry of $y^{(h_t)}$ with 0
15:  Append $\hat{y}_t$ to $C_{h_t}$ and 0 to $r_{.,\hat{y}_t}$
16: else if $h_t = -1$ (imputing) then
17:  Append $\hat{y}_t$ to $C_{\hat{y}_t}$, output of $\hat{y}_t$-th GCN to $r_{.,\hat{y}_t}$
18: end if
19: end for

### 4 Experiments

In this section we compare baseline method LINUCB which ignores the data with missing rewards to ROGCN, BILINUCB and GCNUCB — algorithm proposed in this paper. We consider four different datasets: CNAE-9 and Internet Advertisements from the the UCI Machine Learning Repository\(^1\), Cora\(^2\), and Warfarin [Sharabiani et al., 2015]. Cora is naturally a graph structured data which can be utilized by ROGCN, BILINUCB with ROGCN based imputation and GCNUCB. For other datasets we use a 5-NN graph built online from the available data as follows.

Suppose at step $t$ we have observed data points $x_i \in \mathbb{R}^D$ for $i = 1, \ldots, t$. Weights of the similarity graph computed as

\(^1\)https://archive.ics.uci.edu/ml/datasets.html
\(^2\)https://people.cs.umass.edu/ mccallum/data.html
follows:
\[ A_{ij} = \exp \left( \frac{1}{\sigma^2} \frac{\|x_i - x_j\|^2}{2} \right). \]  
\[ (5) \]

As it was done by [Defferrard et al., 2016] we set \( \sigma = \frac{1}{k} \sum_{i=1}^{k} d(i, i_k) \), where \( d(i, i_k) \) denotes L2 distance between observation \( i \) and its \( k \)-th nearest neighbour indexed \( i_k \). The k-NN adjacency \( A \) is obtained by setting all but \( k \) (excluding itself) corresponding closest entries of \( A_{ij}, i, j = 1, \ldots, t \) to 0 and symmetrizing. Then, as in [Kipf and Welling, 2016], we add self connections and row normalize \( \hat{A} = \left( D + \mathbb{I} \right)^{-1/2} (A + \mathbb{I}) \left( D + \mathbb{I} \right)^{-1/2}, \) where \( D_{ii} = \sum_{j=1}^{t} A_{ij} \) is the diagonal matrix of node degrees.

For pre-processing we discarded features with large magnitudes (3 features in Internet Advertisements and 2 features in Warfarin) and row normalized all observations to have unit \( l_1 \) norm.

For all of our algorithms that use GCN we use default parameters of the GCN and Adam optimizer [Kingma and Ba, 2014]. Default parameters are as follows: 16 hidden units, learning rate of 0.01, 0.0005 weight decay, and dropout of 0.5.

To emulate the OPR setting we randomly permute the order of the observations in a dataset and remove labels for some portion (we experiment with three settings: 25%, 50% and 75% missing labels) of the observations chosen at random. For all methods we consider initial data \( X_0 \) and \( y_0 \) to represent a single observation per class chosen randomly \((T_0 = K)\). At a step \( t = T_0 + 1, \ldots, T \) each algorithm is given a feature vector \( x_t \) and is ought to make a prediction \( \hat{y}_t \). The environment response \( h_t \in \{-1, 0, 1\} \) is then observed and algorithms moves onto step \( t + 1 \). To compare performance of different algorithms at each step \( t \) we compare \( \hat{y}_t \) to true label \( y_t \) available from the dataset (but concealed from the algorithms themselves) to evaluate running accuracy. Defined as such, accuracy is inversely proportional to regret.

**Imputation Methods.** We test two different imputation functions \( I(\cdot) \) for BILINUCB - a ROGCN and simple k-means clustering with 10 clusters. Henceforth, we denote these two approaches as BILINUCB-GCN and BILINUCB-KMeans. In BILINUCB-GCN we update ROGCN with incoming observations and use the softmax class prediction to impute missing reward when needed. In BILINUCB-KMeans, we use the mini-batch k-means algorithm to cluster incoming observations online and impute missing reward with the average non-missing reward of all observations in the corresponding cluster.

**Running accuracy results.** We noticed that BILINUCB with both imputation approaches and GCNUCB are more robust to data ordering when we use baseline LINUCB for first 300 steps and then proceed with the corresponding algorithm (see Figure 1a where aforementioned algorithms and LINUCB perform the same until step 300 and then have individual running accuracies). For all LINUCB based approaches we used exploration-exploitation trade-off parameter \( \alpha = 0.25 \). Results are summarized in Table 1. Since ordering of the data can affect the problem difficulty, we performed 10 data resampling for each setting to obtain error bars.

GCNUCB outperforms the LINUCB baseline and our other proposed methods in all of the experiments, validating our intuition that a method synthesizing the exploration capabilities of bandits coupled with the effective feature representation power of neural networks is the best solution to the OPR problem. We see the greatest increase in accuracy between GCNUCB and the alternative approaches on the Cora dataset which has a natural adjacency matrix. This suggests that GCNUCB has a particular edge in OPR applications with graph structure. Such problems are ubiquitous. Consider our motivating example of dialog systems - for dialog systems deployed in social network or workplace environments, there exists graph structure between users, and user information can be considered alongside queries for personalization of responses.

**Role of bounding the imputed reward.** Notice that on average, a BILINUCB method outperforms LINUCB and ROGCN. To understand the role of these imputation bounds, we analyze the effects of random imputation. We denote this use of random imputation as BILINUCB-Random and LINUCB-Random as the same without bounding the imputed reward. We define BILINUCB-KMeans and ILINUCB-KMeans similarly. We summarize the overall accuracy of each method on CNAE-9 in Table 2.

As the purpose of these bounds is to correct for errors in the imputation method, we expect to see its impact the most when imputation is inaccurate. This is exactly what we see in Table 2 and Figure 1a. When we use a reasonable imputation method, k-means, the imputation bounds do not improve, or only make slight improvements to the results. The improvement gain is much more evident with random imputation, and across both imputation methods, the bounds have a larger impact when there is more reward missing.

**Visualizing GCNUCB context space.** Recall that the context for each arm of GCNUCB is provided by the corresponding binary GCN hidden layer. The motivation for using binary GCNs to provide the context to LINUCB is the ability of GCN to construct more powerful features using graph convolution and neural networks expressiveness. To see how this procedure improves upon the baseline LINUCB utilizing input features as context, we project the context and the corresponding bandit weight vectors, \( \theta_1, \ldots, \theta_K \), for both LINUCB and GCNUCB to a 2-dimensional space using t-SNE [Maaten and Hinton, 2008]. In this experiment we analyzed CNAE-9 dataset with 25% missing labels. Recall that the bandit makes prediction based on the upper confidence bound of the regret: \( \arg\max_k (\theta_k^T x_{k, t} + \sigma_k) \) and that \( x_{k, t} = x_t \) \( \forall k = 1, \ldots, K \) for LINUCB and \( x_{k, t} = g(X)_t^{(k)} \) for GCNUCB. To better visualize the quality of the learned weight vectors, for this experiment we set \( \alpha = 0 \) and hence \( \sigma_k = 0 \) resulting in a greedy bandit, always selecting an arm maximizing expected reward \( \theta_k^T x_{k, t} \). In this case, a good combination of contexts and weight vectors is the one where observations belonging to the same class are well clustered and corresponding bandit weight vector is directed at this cluster. For LINUCB (Figure 1b, 68% accuracy) the bandit weight vectors mostly point in the direction of their re-
spective context clusters, however the clusters themselves are scattered, thereby inhibiting the capability of LINUCB to effectively distinguish between different arms given the context. In the case of GCNUCB (Figure 1c, 77% accuracy) the context learned by each GCN is tightly clustered into two distinguished regions - one with context for corresponding label and binary GCN when it is the correct label (points with bolded colors), and the other region with context for the label and GCN when a different label is correct (points with faded colors). The tighter clustered contexts allow GCNUCB to effectively distinguish between different arms by assigning higher expected reward to contexts from the correct binary GCN than others, thereby resulting in better performance of GCNUCB than other methods.

Table 2: CNAE-9 total average accuracy

| % Reward Missing | LINUCB-Random | BILINUCB-Random |
|------------------|---------------|-----------------|
| 25               | 67.29 ± 4.15  | 67.65 ± 4.30    |
| 50               | 65.67 ± 5.20  | 67.19 ± 5.37    |
| 75               | 49.77 ± 4.68  | 56.36 ± 3.71    |
| % Reward Missing | LINUCB-KMeans | BILINUCB-KMeans |
| 25               | 67.92 ± 3.98  | 67.69 ± 4.30    |
| 50               | 67.14 ± 4.84  | 67.37 ± 5.18    |
| 75               | 56.62 ± 4.40  | 57.16 ± 3.57    |

5 Conclusion and Discussion

We have defined and studied the problem of Online Partially Rewarded (OPR) learning, which combines challenges from semi-supervised learning and multi-armed contextual bandits. We have developed ROGCN and BILINUCB - extensions of popular algorithms in the corresponding domains to solve the OPR problem. Our main contribution, GCNUCB algorithm, is the efficient synthesis of the strengths of the two approaches. Our experiments show that GCNUCB, which combines feature extraction capability of the graph convolution neural networks and natural ability of contextual bandits to handle online learning with reward (instead of labels), is the best approach for OPR across a LINUCB baseline and other algorithms that we proposed.

References

[Agrawal and Goyal, 2013] Shipra Agrawal and Navin Goyal. Thompson sampling for contextual bandits with linear payoffs. In ICML (3), pages 127–135, 2013.

[Allesiardo et al., 2014] Robin Allesiardo, Raphaël Féraud, and Djallel Bouneffouf. A neural networks committee for the contextual bandit problem. In International Conference on Neural Information Processing, pages 374–381. Springer, 2014.

[Auer et al., 2002a] Peter Auer, Nicolò Cesa-Bianchi, and Paul Fischer. Finite-time analysis of the multiarmed bandit problem. Machine Learning, 47(2-3):235–256, 2002.

[Auer et al., 2002b] Peter Auer, Nicolò Cesa-Bianchi, Yoav Freund, and Robert E. Schapire. The nonstochastic multiarmed bandit problem. SIAM J. Comput., 32(1):48–77, 2002.
(a) Accuracy on CNAE-9; 50% missing labels

(b) t-SNE embeddings of context and bandit weight vectors for LINUCB

(c) t-SNE embeddings of context and bandit weight vectors for GCNUCB

[Balakrishnan et al., 2018] Avinash Balakrishnan, Djallel Bouneffouf, Nicholas Mattei, and Francesca Rossi. Using contextual bandits with behavioral constraints for constrained online movie recommendation. In Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, IJCAI 2018, July 13-19, 2018, Stockholm, Sweden., pages 5802–5804, 2018.

[Bartók et al., 2014] Gábor Bartók, Dean P. Foster, Dávid Pál, Alexander Rakhlin, and Csaba Szepesvári. Partial monitoring—classification, regret bounds, and algorithms. Mathematics of Operations Research, 39(4):967–997, 2014.

[Belkin et al., 2006] Mikhail Belkin, Partha Niyogi, and Vikas Sindhwani. Manifold regularization: A geometric framework for learning from labeled and unlabeled examples. Journal of Machine Learning Research, 7(Nov):2399–2434, 2006.

[Bouneffouf and Féraud, 2016] Djallel Bouneffouf and Raphaël Féraud. Multi-armed bandit problem with known trend. Neurocomputing, 205:16–21, 2016.

[Bouneffouf and Rish, 2019] Djallel Bouneffouf and Irina Rish. A survey on practical applications of multi-armed and contextual bandits. CoRR, abs/1904.10040, 2019.

[Bouneffouf et al., 2017a] Djallel Bouneffouf, Irina Rish, and Guillermo A. Cecchi. Bandit models of human behavior: Reward processing in mental disorders. In Tom Everitt, Ben Goertzel, and Alexey Potapov, editors, Artificial General Intelligence - 10th International Conference, AGI 2017, Melbourne, VIC, Australia, August 15-18, 2017, Proceedings, volume 10414 of Lecture Notes in Computer Science, pages 237–248. Springer, 2017.

[Bouneffouf et al., 2017b] Djallel Bouneffouf, Irina Rish, Guillermo A. Cecchi, and Raphaël Féraud. Context attentive bandits: Contextual bandit with restricted context. In Carles Sierra, editor, Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence, IJCAI 2017, Melbourne, Australia, August 19-25, 2017, pages 1468–1475. ijcai.org, 2017.

[Bouneffouf et al., 2019] Djallel Bouneffouf, Srinivasan Parthasarathy, Horst Samulowitz, and Martin Wistuba. Optimal exploitation of clustering and history information in multi-armed bandit. In Sarit Kraus, editor, Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI 2019, Macao, China, August 10-16, 2019, pages 2016–2022. ijcai.org, 2019.

[Bouneffouf et al., 2020a] Djallel Bouneffouf, Raphaël Féraud, Sohini Upadhyay, Yasaman Khazaeni, and Irina Rish. Double-linear thompson sampling for context-attentive bandits, 2020.

[Bouneffouf et al., 2020b] Djallel Bouneffouf, Sohini Upadhyay, and Yasaman Khazaeni. Contextual bandit with missing rewards. CoRR, abs/2006.15194, 2020.

[Bouneffouf, 2020] Djallel Bouneffouf. Online learning with corrupted context: Corrupted contextual bandits. CoRR, abs/2007.06368, 2020.

[Bronstein et al., 2017] Michael M Bronstein, Joan Bruna, Yann LeCun, Arthur Szlam, and Pierre Vandergheynst. Geometric deep learning: going beyond Euclidean data. IEEE Signal Processing Magazine, 34(4):18–42, 2017.

[Chapelle et al., 2009] Olivier Chapelle, Bernhard Scholkopf, and Alexander Zien. Semi-supervised learning. IEEE Transactions on Neural Networks, 20(3):542–542, 2009.

[Chu et al., 2011] Wei Chu, Lihong Li, Lev Reyzin, and Robert E. Schapire. Contextual bandits with linear payoff functions. In Geoffrey J. Gordon, David B. Dunson, and Miroslav Dudík, editors, AISTATS, volume 15 of JMLR Proceedings, pages 208–214. JMLR.org, 2011.

[Defferrard et al., 2016] Michaël Defferrard, Xavier Bresson, and Pierre Vandergheynst. Convolutional neural networks on graphs with fast localized spectral filtering. In Advances in Neural Information Processing Systems, pages 3844–3852, 2016.

[Gajane et al., 2016] Pratik Gajane, Tanguy Urvoy, and Emilie Kaufmann. Corrupt bandits. EWRL, 2016.

[Henaff et al., 2015] Mikael Henaff, Joan Bruna, and Yann LeCun. Deep convolutional networks on graph-structured data. arXiv preprint arXiv:1506.05163, 2015.
