AFRAID: Fraud Detection via Active Inference in Time-evolving Social Networks

Véronique Van Vlasselaer* Tina Eliassi-Rad† Leman Akoglu‡ Monique Snoeck* Bart Baesens*§
* Department of Decision Sciences and Information Management, KU Leuven, Naamsestraat 69, B-3000 Leuven, Belgium
{Veronique.VanVlasselaer,Monique.Snoeck,Bart.Baesens}@kuleuven.be
† Department of Computer Science, Rutgers University, 110 Frelinghuysen Road, Piscataway, NJ 08854-8019, US
eliassi@cs.rutgers.edu
‡ Department of Computer Science, Stony Brook University, 1425 Computer Science, Stony Brook, NY 11794-4400, US
leman@cs.stonybrook.edu
§ School of Management, University of Southampton, Highfield Southampton, SO17 1BJ, United Kingdom

***Abstract*—Fraud is a social process that occurs over time. We introduce a new approach, called AFRAID, which utilizes active inference to better detect fraud in time-varying social networks. That is, classify nodes as fraudulent vs. non-fraudulent. In active inference on social networks, a set of unlabeled nodes is given to an oracle (in our case one or more fraud inspectors) to label. These labels are used to seed the inference process on previously trained classifier(s). The challenge in active inference is to select a small set of unlabeled nodes that would lead to the highest classification performance. Since fraud is highly adaptive and dynamic, selecting such nodes is even more challenging than in other settings. We apply our approach to a real-life fraud data set obtained from the Belgian Social Security Institution to detect social security fraud. In this setting, fraud is defined as the intentional failing of companies to pay tax contributions to the government. Thus, the social network is composed of companies and the links between companies indicate shared resources. Our approach, AFRAID, outperforms the approaches that do not utilize active inference by up to 15% in terms of precision.**

I. Introduction

Data mining techniques offer a good solution to find patterns in vast amounts of data. Human interaction is often an indispensable part of data mining in many critical application domains [1], [2]. Especially in fraud detection, inspectors are guided by the results of data mining models to obtain a primary indication where fraudulent behavior might situate. However, manual inspection is time-consuming and efficient techniques that dynamically adapt to a fast changing environment are essential. Due to the limited resources of fraud inspectors, fraud detection models are required to output highly precise results, i.e. the hit rate of truly identified fraudsters should be maximal. In this work, we investigate how *active inference* fosters the fraud detection process for business applications over time. Active inference is a subdomain of active learning where a network-based algorithm (e.g., collective inference) iteratively learns the label of a set of unknown nodes in the network in order to improve the classification performance. Given a graph at time t with few known fraudulent nodes, which k nodes should be probed – that is, inspected to confirm the true label – such that the misclassification cost of the collective inference (CI) algorithm is minimal. In this work, we consider *across-network* and *across-time* learning, as opposed to within-network learning [3]. We combine the results of CI with local-only features in order to learn a model at time t and predict which entities (i.e., nodes) are likely to commit fraud at time $t+1$.

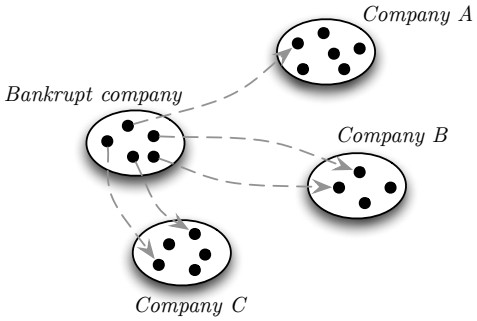

Fig. 1: Fraud process: a fraudulent company files for bankruptcy in order to avoid paying taxes and transfers its resources to other companies that are part of the illegal setup, also known as a spider construction.

Each time period fraud inspectors have a limited budget b at their disposal to investigate suspicious instances. This budget might refer to time, money or the number of instances to be inspected. If we invest k of budget b to ask inspectors about the true label of a set of instances selected based on a selection criterion, will the total budget b be better spent? That is, do we achieve more precise results by investing a part of the budget (k) in learning an improved algorithm while the remaining budget $l = b - k$ is used to investigate the re-evaluated results, rather than by using the complete budget b to inspect the initial results without learning?

We propose AFRAID (short for: Active Fraud Investigation and Detection) and apply our developed approach to social security fraud. In social security fraud, companies set up illegal constructions in order to avoid paying tax contributions. While detection models can rapidly generate a list of suspicious companies, which k companies should be inspected such that the expected label of all other companies minimizes the tax loss due to fraud?

Our contributions are the following:

- Fraud is *dynamic* and evolves over time. We propose a new approach for active inference in a timely manner by (1) using time-evolving graphs, and (2) weighing inspectors' decisions according to recency. (1) The influence that nodes exercise on each other varies over time. We capture the extent of influence in time-varying edge weights of the graph. Additionally, we attach greater importance to recent fraud. (2) Given that an inspector labels a specific node as legitimate at

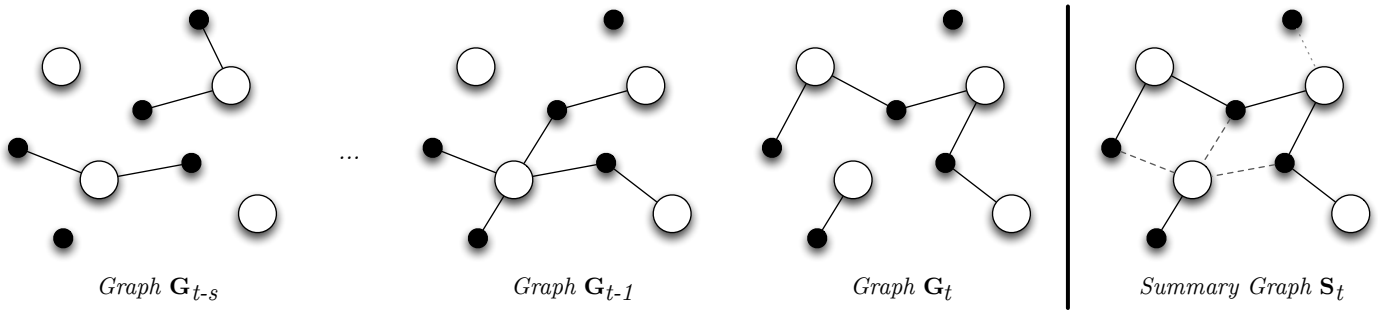

Fig. 2: A summary graph $\mathcal{S}_t$ at time t contains all nodes and edges observed between time t and time $t-s$.

time t, can we assume that the node is still legitimate at time $t+1$? We introduce how to temporarily integrate an inspector's decision in the network model, decreasing the value of the decision over time.

- We propose a combination of *simple and fast probing strategies* to identify nodes that might possibly distort the results of a collective inference approach and apply these strategies to a large real-life fraud graph. We evaluate probing decisions made by (1) a committee of local classifiers, and (2) by insights provided by inspectors. (1) A committee of local classifiers collectively votes for the most uncertain nodes without relying on domain expertise. (2) Inspectors use their intuition to formalize which nodes might distort the collective inference techniques.
- We investigate the benefits of investing k of the total budget b in learning a better model, and find that active inference boosts the performance of the classifier in terms of precision and recall.

The remainder of the paper is organized as follows: background (§ II), network definition (§ III), problem definition and active inference (§ IV), results (§ V), related work (§ VI) and conclusion (§ VII).

II. Background

The data used in this study is obtained from the Belgian Social Security Institution, a federal governmental service that collects and manages employer and employee social contributions. Those contributions are used to finance various branches of social security including the allowance of unemployment funds, health insurance, family allowance funds, etc. Although the contributions both concern employees and employers (i.e., companies), the taxes are levied at employer level. That means that the employer is responsible for transferring the taxes to the Social Security Institution.

We say that a company is fraudulent if the company is part of an illegal set up to avoid paying these taxes. Recent developments have shown that fraudulent companies do not operate by themselves, but rely on other associate companies [4], [5]. They often use an interconnected network, the so-called *spider constructions*, to perpetrate tax avoidance. Figure 1 illustrates the fraud process. A company that cannot fulfill its tax contributions to the government files for bankruptcy. If the company is part of an illegal setup, all its resources (e.g., address, machinery, employees, suppliers, buyers, etc.) are transferred to other companies within the setup. While the economical structure of the company is disbanded by means of bankruptcy, the technical structure is not changed as all resources are re-allocated to other companies and continue their current activities. Network analysis is thus a logical enrichment of traditional fraud detection techniques.

Companies can be related to each other by means of common resources they use(d). Although we cannot specify the exact type of resources, the reader can understand resources in terms of shared addresses, employees, suppliers, buyers, etc. The data we have at our disposal contains 10M records of which resources belong(ed) to which companies for which time period. The data consists of 390k companies and 5,6M resources. Remark that resources can be associated with more than one company at the same time. Although resource sharing (or transferring) might indicate a spider construction, non-fraudulent companies also exchange resources (e.g., in an acquisition or merger between companies all resources of one company are allocated to the other, employees changing jobs creates a relationship between two companies, etc.). Also, fraudulent setups use innocent resources to cover up their tracks. Given a set of companies, resources and the relations between them at time t, our objective is to identify those companies that have a high likelihood to perpetrate fraud at time $t+1$.

III. Network definition

In this section, we will elaborate on how to use the temporal-relational data to create time-evolving fraud networks. Given relational data at time t, the corresponding graph is defined as $\mathcal{G}_t = (\mathcal{V}_t, \mathcal{E}_t)$, with $\mathcal{V}_t$ the set of nodes (or points, or vertices) and $\mathcal{E}_t$ the set of edges (or lines, or links) observed at time t. Graph $\mathcal{G}_t$ describes the static network at time t.

Besides current relationships, dynamic graphs keep track of the evolution of past information, e.g. nodes that are added to or removed from the network, edges that appear and disappear, edge weights that vary over time, etc. In order to include a time aspect in the network, we define the summary graph $\mathcal{S}_t$ at time t as all the nodes and edges observed between time $t-s$ and t. Figure 2 depicts how a summary graph is created. For our problem setting, we include all historical information available ($s = t$), as fraud is often subtle and takes a while before the relational structure is exhibited. Although historical links hold important information about possible spread of fraud, their impact differs from more recent links. Based on work of [6], [7], we exponentially decay the edge weight over time as follows

$$w(i,j) = e^{-\alpha h} \tag{1}$$

with α the decay value (here: $\alpha = 0.02$) and h the time passed since the relationship between node i and j occurred and where $h = 0$ depicts a current relationship. Mathematically, a network is represented by an adjacency matrix $\mathbf{A}$ of size $n \times n$ where

$$\begin{cases} a_{i,j} = w(i,j) & \text{if } i \text{ and } j \text{ are connected} \\ a_{i,j} = 0 & \text{otherwise} \end{cases} \tag{2}$$

Since companies are explicitly connected to the resources they use, our fraud graph has a dual structure: every edge in the network connects a company to a resource. The network composed of n companies and m resources is called a *bipartite* network, and is of size $n \times m$. The corresponding adjacency matrix is $\mathbf{B}_{n \times m}$. As we know when a resource was assigned to a company, the edge weight corresponds to the recency of their relationship, exponentially decayed over time. In case multiple relationships exist between a company and a resource, we only include the most recent one. An edge weight with maximum value 1 refers to a current assignment.

IV. Active Inference

Collective inference is a network analysis technique where the label of a node in the network is said to depend on the label of the neighboring nodes. In social network analysis, this is often referred to as *homophily* [8], where one tends to adopt the same behavior as one's associates (e.g., committing fraud if all your friends are fraudsters). A change in the label of one node might cause the label of the neighboring nodes to change which in turn can affect the label of their neighbors, and so on. As a consequence, a wrong expectation of one node strongly affects the estimated label of the other nodes. Active inference is analogous to active learning. It selects an observation to be labeled in order to improve the classification performance. While active learning iteratively re-learns and updates a classifier by the newly acquired label, active inference re-evaluates the labels of the neighboring nodes using an existing model. For a profound literature survey of active learning, we refer the reader to [9].

In this work, we train a set of out-of-time local classifiers $\vec{\mathsf{LC}}$ at time t where each observation i is composed of a set of features $\vec{x}_i$ derived at time $t-1$ and the corresponding label $\mathcal{L}_i = \{\texttt{fraud}, \texttt{non-fraud}\}$ observed at time t. The set of features consists of (1) intrinsic features $\vec{a}_i$, and (2) neighborhood features (see IV-A). Intrinsic features are features that occur in isolation and do not depend on the neighborhood. The intrinsic features that describe the companies in our analysis include age, sector, financial statements, legal seat, etc. The neighborhood features are derived by a collective inference technique. We apply each classifier LC_m to observations from time t in order to predict which observations are likely to commit fraud at time $t+1$. In active inference, we ask inspectors their most probable label at time $t+1$ and already integrate this label in the current network setting to infer a new expectation of the neighbors' label. We say that we learn *across-time* and *across-network*. Recall that inspectors have a total budget b at their disposal each timestamp, and are able to invest $k < b$ budget in improving the current collective inference algorithm. Using the updated feature set, the LC re-learns a new estimate of each of the nodes' fraud probability. However, as inspectors' decisions are only temporarily valid, we temporally weigh the belief in a decision, by decreasing its value in time. Algorithm 1 provides more details on the procedure for active inference in time-varying fraudulent networks, and will be discussed in the remainder of this section.

Algorithm 1: Active inference for time-varying fraud graphs.

input : Summary graph $\mathcal{S}_{t-1}$ and $\mathcal{S}_t$ where $\mathcal{S}_t = (\mathcal{V}_{s,t}, \mathcal{E}_{s,t})$, time-weighted collective inference algorithm wRWR', budget k, set of labeled fraudulent nodes $\mathcal{L}_{t-1}$ and $\mathcal{L}_t$.
output: Labeled nodes $\mathcal{L}_{t+1}$.

 # Initialize LC_t
$\vec{\xi}_{t-1} \leftarrow \mathsf{wRWR}'(\mathcal{S}_{t-1}, \mathcal{L}_{t-1})$; IV-A
$\mathsf{LC}_t \leftarrow \mathsf{LC}(\vec{x}_{t-1}[\vec{a}_{t-1}, \mathsf{aggr}(\vec{\mathcal{N}}_{t-1}), \vec{\xi}_{t-1}], \mathcal{L}_t)$;
 # Active inference
$\ell \leftarrow 0$
while $\ell < k$ **do**
  $\vec{\xi}_t \leftarrow \mathsf{wRWR}'(\mathcal{S}_t, \mathcal{L}_t)$;
  $\mathcal{L}_{t+1} \leftarrow \mathsf{LC}_t(\vec{x}_t[\vec{a}_t, \mathsf{aggr}(\vec{\mathcal{N}}_t), \xi_t], \mathcal{L}_t)$;
  Select node v_i to probe; IV-B
  if $y(v_i)$ = fraudulent **then**
    $\mathcal{L}_t(v_i) \leftarrow$ (fraud,t); IV-C1
  else if $y(v_i)$ = non-fraudulent **then**
    $\forall v_j \in \mathcal{N}_i : w(j,i) = 0$; IV-C2
  end
  $\ell \leftarrow \ell + 1$
end

A. Collective Inference Technique

Many collective inference algorithms have been proposed in the literature (see [10] for an overview). We employ a set of local classifiers that evaluates the classification decision on both intrinsic and neighborhood features. For the neighborhood features, we make a distinction between (1) local neighborhood features and (2) a global neighborhood feature. The local neighborhood features are based on the labels of the direct neighbors. Recall that in our bipartite graph only the labels of the companies are known, and that the first order neighborhood of each company is composed of its resources. We define the direct neighborhood of a company as the company's resources and their associations. As the number of neighbors for each node differs, the neighborhood labels are aggregated in a fixed-length feature vector [10] (here: length = 3). The following aggregated features $\mathsf{aggr}(\mathcal{N}_i)$ are derived from the network for each company i.

- **Weighted Sum**: the number of fraudulent companies associated through a similar resource, weighted by the edges.
- **Weighted Proportion**: the fraction of fraudulent companies associated through a similar resource, weighted by the edges.

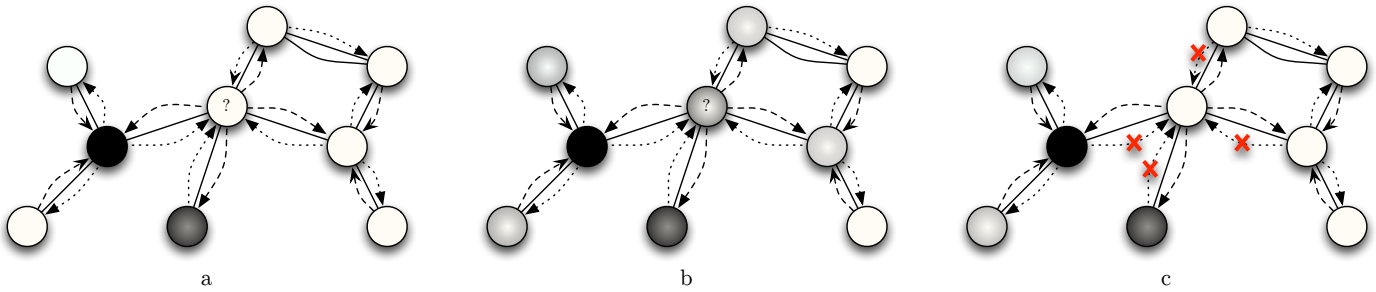

Fig. 3: Time-weighted collective inference algorithm. (a) At time t, two companies in the subgraph are fraudulent. The intensity of the color refers to the recency of the fraudulence. (b) Propagation of fraud through the network by the wRWR′ algorithm. (c) Cutting the incoming edges after probing node '?' and confirming its non-fraudulent label.

- **Weighted Mode**: binary indicator for whether the neighborhood is mainly fraudulent or non-fraudulent.

If – due to probing companies – the label of one of the neighbors changes, the local neighborhood is directly impacted. After each iteration of Algorithm 1, the local neighborhood features are locally updated.

The global neighborhood feature is inferred using a variant of Random Walk with Restarts (RWR) [11]. RWR computes the relevance score between any two nodes in the network. Given a time-varying fraudulent network, RWR allows us to compute the extent to which each company is *exposed* to a fraudulent company at time t using the following equation:

$$\vec{\xi_t} = c \cdot \mathbf{A}_t \cdot \vec{\xi_t} + (1-c) \cdot \vec{e_t} \tag{3}$$

with $\vec{\xi_t}$ a vector containing the exposure (or relevance) scores at time t, $\mathbf{A}_t$ the adjacency matrix, c the restart probability (here: 0.85) and $\vec{e_t}$ the restart vector. The exposure score of a node in the network depends with a probability c on its neighboring nodes, and with a probability of $(1-c)$ on a personalized vector $\vec{e_t}$. Considering the problem-specific characteristics that concur with fraud, Equation 3 is modified such that it satisfies (1) the bipartite structure defined by our problem setting, (2) the temporal effect of confirmed fraudulent companies on the network and (3) the fact that fraud should equally affect each resource, regardless whether the resource is assigned to a large or small company.

(1) Given our bipartite network of companies and resources, we only know the label of fraudulent companies and need to decide on how each (non-fraudulent) company is currently exposed by the set of fraudulent companies. Therefore, the adjacency matrix $\mathbf{B}_t$ of a bipartite summary graph $\mathcal{S}_t$ with n nodes of type 1 (here: companies) and m nodes of type 2 (here: resources), is transformed to a network with an equal number of rows and columns according to [12], and

$$\mathbf{M}_{(n+m)\times(n+m)} = \begin{pmatrix} \mathbf{0}_{n\times n} & \mathbf{B}_{n\times m} \\ \mathbf{B'}_{m\times n} & \mathbf{0}_{m\times m} \end{pmatrix} \tag{4}$$

Remark that matrix $\mathbf{M}$ represents an undirected graph where $m(i,j) = m(j,i)$. The row-normalized adjacency matrix is then denoted as $\mathbf{M}_{norm}$, where all rows sum up to 1.

(2) As we want to determine how fraud affects the other nodes, we initialize the restart vector with fraud. The restart vector $\vec{e_i}$ is constructed as follows

$$\begin{cases} e_i = e^{-\beta h} & \text{if } i \text{ is a fraudulent company} \\ e_i = 0 & \text{otherwise} \end{cases} \tag{5}$$

with β the decay value, and h the time passed at time t since fraud is detected at that company. Equation 5 weighs fraud in time and assigns a higher weight to more recent fraud.

(3) Finally, fraudulent companies with many resources will have a smaller effect on their neighbors than companies with only few resources. In order to avoid emphasizing low-degree companies, we modify the starting vector with the degree:

$$\vec{e}' = \vec{e} \times \vec{d} \tag{6}$$

where $\vec{e}'$ is the element-wise product of the time-weighted restart vector $\vec{e}$ and the degree vector $\vec{d}$. The normalized starting vector $\vec{e}'_{norm}$ defines the starting vector where all elements sum to 1.

Equation 3 (referred to as wRWR′) is then re-written as

$$\vec{\xi_t} = c \cdot \mathbf{M}_{norm,t} \cdot \vec{\xi_t} + (1-c) \cdot \vec{e}'_{norm,t} \tag{7}$$

In order to compute the exposure scores, Equation 7 requires a matrix inversion. As this is often unfeasible to compute in practice, we use the power-iteration method iterating Equation 7 until convergence [11]. Convergence is reached when the change in exposure scores is marginal or after a predefined number of iterations. The modified RWR algorithm as described above is illustrated in Figure 3a and 3b.

B. Probing strategies

Given a set of observations with an estimated label by a local classifier LC, which observation should be probed (i.e., checked for its true label) such that the predicted label of the other observations are maximally improved? Recall that the feature set of each observation from which the LC estimates the label depends on the neighborhood of that observation. Any change made in the label of one node has a direct impact on the feature set of the neighbors. We define 5 probing strategies: committee-based, entropy-based, density-based, combined and random strategy.

*1) **Committee-based strategy**:* Rather than to rely on the decision of one LC, many LCs decide on which node to pick in a committee-based strategy. An often used approach is uncertainty sampling. That is, sample that observation about which all the members of the committee are the most uncertain. Our committee is composed of the set of local classifiers $\vec{\mathsf{LC}}$. Each local classifier LC_m expresses how confident it is in the estimated label of each observation by means of a probability. Sharma and Bilgic [13] distinguishes between two types of uncertainty: most-surely and least-surely uncertainty. The most-surely uncertain node is that node for which the estimated probabilities of the local classifiers provide equally strong evidence for each class. For example, when half of the committee members vote for `fraud`, and the other half vote for `non-fraud`, we say that the committee is most-surely uncertain about the node's label. Least-surely uncertainty refers to that node for which the estimated probabilities do not have significant evidence for either class. The committee is least-surely uncertain about a node's label if the probability of the node to belong to a class is close to 0.5 for many classifiers. Based on w[13], we combine positive (i.e., belonging to class `fraud`) and negative (i.e., belonging to class `non-fraud`) evidence learned from the set of models. Each local classifier LC_m assigns a fraud estimate to each node x. A model is in favor for a positive label of node x when $P_x(+|\mathsf{LC}_m) > P_x(-|\mathsf{LC}_m)$, then $\mathsf{LC}_m \in \mathcal{P}$ for node x, otherwise $\mathsf{LC}_m \in \mathcal{N}$. Evidence in favor of node x being fraudulent is

$$E^+(x) = \prod_{\mathsf{LC}_m \in \mathcal{P}} \frac{P_x(+|\mathsf{LC}_m)}{P_x(-|\mathsf{LC}_m)} \tag{8}$$

Evidence in favor of node x being a non-fraudulent is

$$E^-(x) = \prod_{\mathsf{LC}_m \in \mathcal{N}} \frac{P_x(-|\mathsf{LC}_m)}{P_x(+|\mathsf{LC}_m)} \tag{9}$$

The most-surely uncertain node (MSU) in the set of unlabeled nodes $\mathcal{U}$ is the node which has the maximal combined evidence.

$$x^* = \underset{x \in \mathcal{U}}{\operatorname{argmax}}\, E(x) = E^+(x) \times E^-(x) \tag{10}$$

The least-surely uncertain node (LSU) is the node which has the minimal combined evidence.

$$x^* = \underset{x \in \mathcal{U}}{\operatorname{argmin}}\, E(x) = E^+(x) \times E^-(x) \tag{11}$$

We define four types of committee-based strategies to sample nodes: (1) most-surely uncertain (MSU), (2) least-surely uncertain (LSU), (3) most-surely uncertain using the best performing local classifiers $(\mathsf{MSU}+)$ and (4) least-surely uncertain using the best performing local classifiers $(\mathsf{LSU}+)$. We implemented sampling strategy (3) and (4), as we found that some poorly performing classifiers fail to appropriately weigh the feature set and distort the results of the uncertainty sampling. Therefore, in $\mathsf{MSU}+$ and $\mathsf{LSU}+$, we allowed only well-performing committee members (i.e., above average precision of all local classifiers) to vote on the node to be probed.

*2) **Entropy-based strategy**:* Fraud is highly imbalanced, having only a limited set of confirmed fraudulent nodes available. However, our network exhibits statistically significant signs of homophily (p-value < 0.02) which indicates that fraudulent nodes tend to cluster together. Some non-fraudulent nodes lie on the boundary between a cluster of fraudulent and non-fraudulent nodes. The entropy-based strategy measures the impurity of the neighbors' labels and identifies these nodes that are associated with a similar amount of fraudulent and non-fraudulent nodes, and

$$\begin{aligned} x^* &= \underset{x \in \mathcal{U}}{\operatorname{argmax}}\, \text{Entropy}(x) \\ &= -d^{(2)}_{rel,x} \log(d^{(2)}_{rel,x}) - (1 - d^{(2)}_{rel,x}) \log(1 - d^{(2)}_{rel,x}) \end{aligned} \tag{12}$$

with $d^{(2)}_{rel,x}$ the fraction of fraudulent nodes associated with node x in the second-order neighborhood (i.e., the companies) at time t.

*3) **Density-based strategy**:* Spider constructions are subgraphs in the network that are more densely connected than other subgraphs. The density-based strategy aims to find those nodes of which the neighborhood is highly interconnected.

$$x^* = \underset{x \in \mathcal{U}}{\operatorname{argmax}} \frac{\text{\# of observed edges}}{\text{\# of all possible edges}} \tag{13}$$

*4) **Combined strategy**:* Based on experts' expertise, the combined strategy searches for companies that are located in (1) a dense neighborhood (= high density), and (2) an impure neighborhood (= high entropy). Evidence is aggregated by multiplication [13]. The node with the maximum value for the combined strategy is selected for probing, and

$$x^* = \underset{x \in \mathcal{U}}{\operatorname{argmax}}\, \text{Combined}(x) = \text{Entropy}(x) \times \text{Density}(x) \tag{14}$$

*5) **Random strategy**:* The random probing strategy randomly picks a node in the network for probing.

Probing strategy (1) does not rely on domain expertise, while (2)-(4) are guided by experts' insights. Strategy (5) is employed as baseline.

C. Temporal weighing of label acquisition

Based on the previous selection technique, the probed node is sent to inspectors for further investigation. Inspectors will confirm the true label of the node. Recall that in our setting only companies can be directly attributed to fraud, resources cannot be passed to the inspectors for investigation. Inspectors will thus only label company nodes. At label acquisition, two scenarios can occur for each node that is probed:

*1) **Classified as fraudulent**:* In this case, the node is added to the bag of fraudulent nodes, and affects (1) the local neighborhood features of the neighbors, and (2) the global neighborhood feature of all nodes. (1) Up until now, the sampled node was considered to be non-fraudulent. Hence, we locally update the feature set of the company's neighbors. (2) The starting vector of the wRWR' algorithm (see IV-A) is re-created, treating the node as a fraudulent one. The global neighborhood feature for each node is then updated.

*2) **Classified as non-fraudulent**:* Inspectors do not find any evidence that this node will be involved in fraud at time $t+1$. However, this does not imply that the node will always be non-fraudulent. The inspectors' decision is only valid for a limited time period. This decision does not impact the local neighborhood features, as the node was treated as non-fraudulent before. It only temporarily affects the exposure scores computed by the wRWR′ algorithm. If we know for certain that node i is legitimate at time t – based on e.g., inspectors' labeling – the node should block any fraudulent influence passing through. By temporarily cutting all the incoming edges to node i, node i will not receive any fraudulent influences, and as a result cannot pass fraudulent influences to its neighbors. The edge weight in the adjacency matrix $\mathbf{M}$ is changed as follows:

$$\forall j \in \mathcal{N}_i : w(j,i) = (1 - e^{-\beta d})e^{-\alpha h} \tag{15}$$

with α and β decay values, t the time passed since the decision that i is non-fraudulent where $d = 0$ if it is a current decision, and h is the time passed since a relation between i and j occurred. Remark that only incoming edges are cut from the non-fraudulent node. The outgoing edges are still intact. This mitigates the effect of fraud on its neighbors. This is illustrated in Figure 3c.

V. Results

We applied our proposed approach for active inference in time-evolving graphs to a real-life data set obtained from the Belgian Social Security Institution. We use historical data for evaluation, allowing us to appropriately interpret results and the value of active inference for our application domain. We trained five local classifiers (i.e., Logistic Regression, Random Forest, Naive Bayes, SVM and Decision Tree) for two timestamps t_1 and t_2. Due to confidentiality issues, we will not specify the exact timestamps of analysis. The local classifiers $\vec{\mathsf{LC}}$ of time t_1 are learned using data features of time t_0 and their corresponding label at time t_1. The model is tested on data features of time t_1 aiming to predict the corresponding label at time t_2. Because inspection is time-consuming, the number of companies that are passed on for further inspection is limited. In our problem setting, we focus on the top 100 most probable fraudulent companies, out of more than 200k active companies, and evaluate model performance on precision, recall and F1-measure.

Figure 4 shows the F1-measure of the local classifiers obtained when investigating the top 100 most likely fraudulent companies in function of the percentage of companies labeled of the budget b. Precision and recall follow a similar pattern, as the total number of companies that committed fraud between t_1 and t_2 reaches approximately 200 ($< 1\%$). The probing strategy used is (MSU+). While Naive Bayes, SVM and Decision Tree are not significantly impacted, the probing strategy is able to identify nodes that change the top 100 most probable frauds for Logistic Regression and Random Forest. Although the benefits for Logistic Regression are not pronounced, the precision achieved by Random Forest increases from 3% up to 15%. Figure 5 depicts the precision achieved by the probing strategies themselves. On average, more than 50% of the probed nodes are labeled by the inspectors as fraudulent.

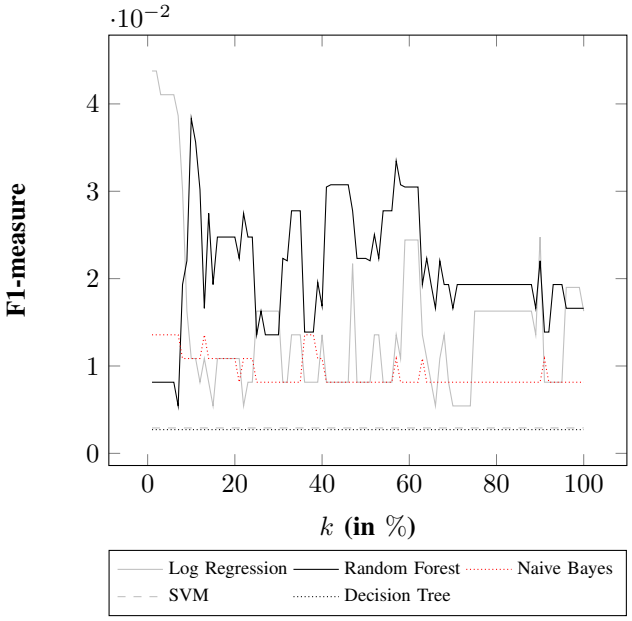

Fig. 4: Model performance of active inference on time t for probing strategy MSU+.

Considering that there are only 200 out of 200k companies that commit fraud during the next time period, this is translated in an increase of approximately 25% recall. These results indicate that the probing strategy on its own is a powerful approach to detect many frauds.

Remark that the curves in Figure 4 vary a lot. This is mainly due to the shift in the top 100 companies, depending on which node is probed. Figure 6 illustrates how the changes in precision (black curve) can be explained by changes in the top 100 most suspicious companies (gray curve, in %). We distinguish three scenarios, as indicated in the figure: (A) The sampled node causes an increase in precision. The sampled node is labeled as non-fraudulent hereby correctly blocking fraudulent influence to the rest of its neighborhood,

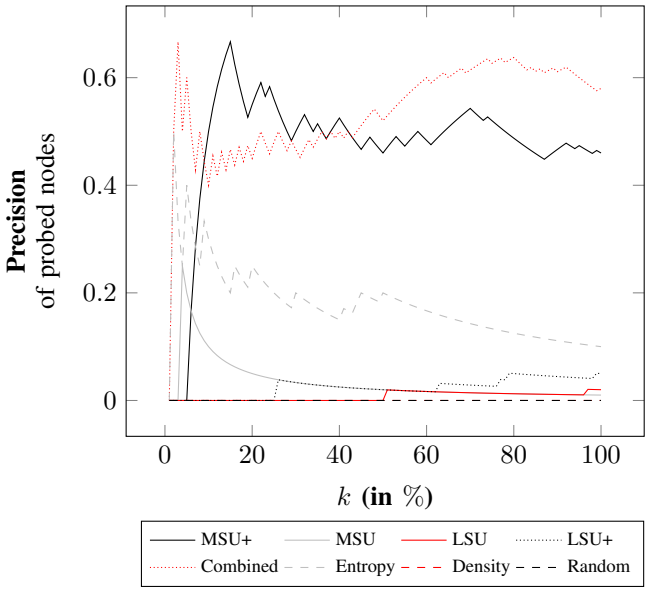

Fig. 5: Precision achieved by the probing strategies.

or the sampled node is labeled as fraudulent intensifying the spread of fraud towards its neighborhood. (B) The sampled node deludes the CI technique. This can be explained by the innocent resources often attached to illegal setups. (C) The sampled node does not have any influence on the top 100, and can be seen as a lost effort.

Figure 7 and 8 compare the different probing strategies. We distinguish between committee-based strategies (Figure 7) and strategies using experts' experience (Figure 8). In general, MSU+, the Entropy-based and Combined strategy achieve approximately the same precision. Consistent with the results of [13], the probing strategies LSU and LSU+ do not contribute to learning, as well as the Density and Random strategy. Surprisingly, we observed that the MSU strategy which uses all classifiers does not perform well. When we apply a committee-based strategy composed of the best members or advanced experts' strategies (i.e., Entropy-based and Combined), we achieve the best performance. We can conclude that a committee of local classifiers can mimic experts' insights, which is often preferred in order to make unbiased inspection decisions.

Finally, we evaluate how model performance is affected by cutting the edges, and gradually re-integrating their influence in time. Figure 9 shows the precision at time t_2 with and without integrating the edge cuts of time t_1. Especially when the probing budget is limited, the precision is positively impacted. When more budget is invested in probing, the effects of transferring decisions of the previous timestamps are similar to the results achieved when no previous edge cuts are taken into account.

VI. Related Work

Active learning iteratively learns a classifier by selecting unlabeled observations to be labeled, and update the classifier accordingly. The label is assigned by an "oracle", which often refers to human interaction present in the learning process. Although active learning is widely explored in the literature (see [9] for an overview), it is only recently applied to networked data. As network-based features often rely on the

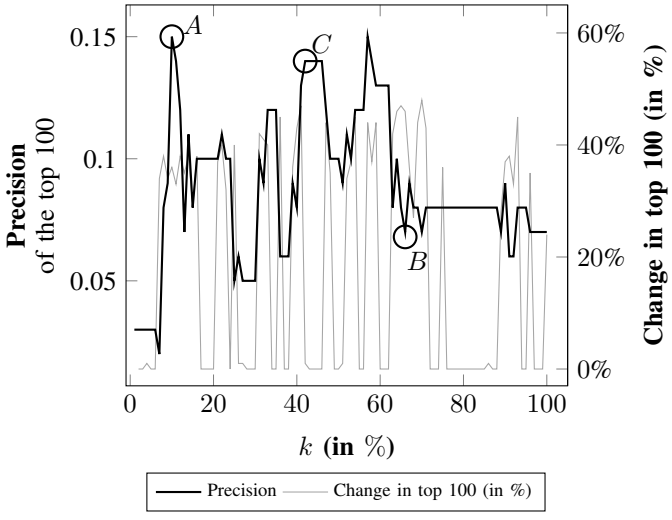

Fig. 6: Changes in precision for Random Forests compared to changes in the evaluation set when using probing strategy MSU+.

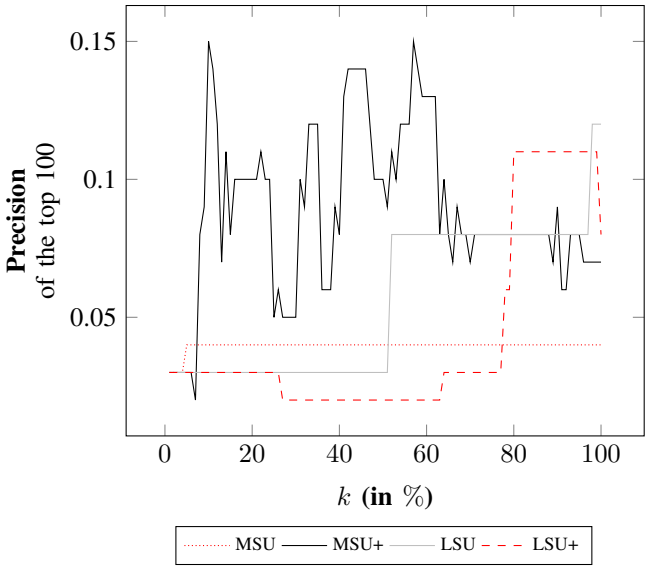

Fig. 7: Precision of the committee-based probing strategies.

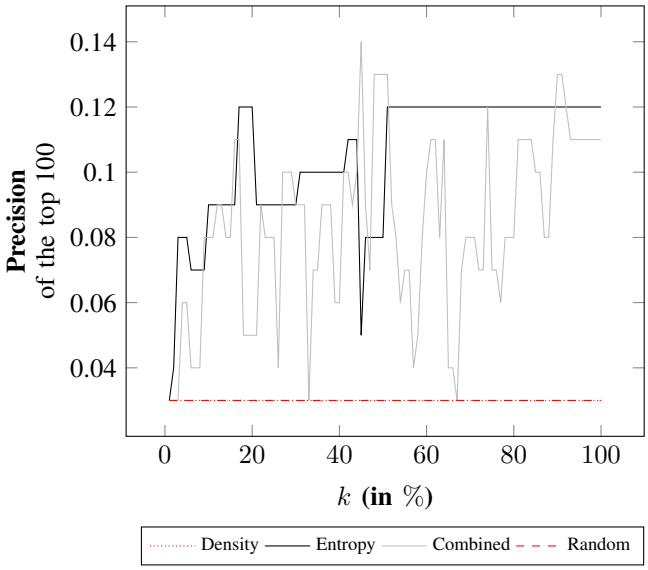

Fig. 8: Precision of the experts-based probing strategies.

neighbors, an update in the neighborhood causes some features to change. This is collective classification, and is proven to be useful for fraud detection in [14], [15]. Active inference refers to the process of iteratively sampling nodes such that the collective classification prediction of all other nodes is optimized. Most studies focus on within-network classification, in which a subset of the nodes are labeled and the labels of the other nodes need to be decided. The goal is to select the most informative (set of) nodes to sample. Rattigan et al. [16] suggest to select those nodes for probing that lie central in the network and impact other nodes more significantly. Macskassy [17] uses the Empirical Risk Minimization (ERM) measure such that the expected classification error is reduced. The Reflect and Correct (RAC) strategy [18], [19] tries to find misclassified islands of nodes by learning the likelihood of each node belonging to such island. In [20], the authors propose ALFNET combining both content-only (or intrinsic) features with features derived from the network. They use local disagreement between a content-only and combined classifier

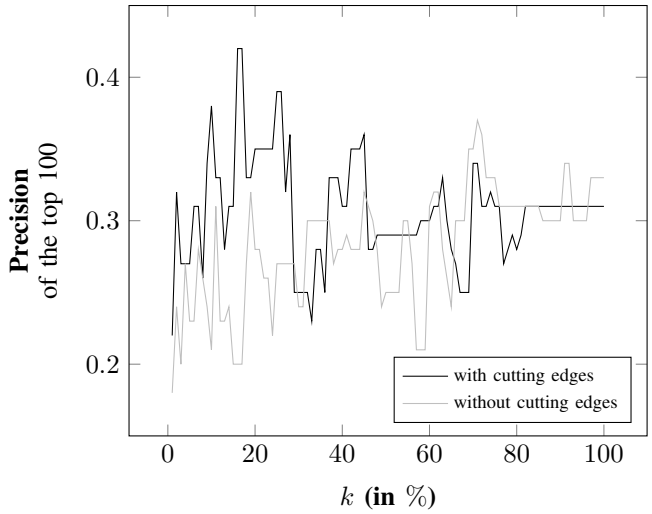

Fig. 9: Precision on graphs when decision is weighted in time.

to decide which node to probe in a cluster. As opposed to within-network learning, Kuwadekar and Neville [3] applied active inference to across-network learning. That is, their Relational Active Learning (RAL) algorithm is bootstrapped on a fully-labeled network and then applied to a new unlabeled network. Samples are chosen based on a utility score that expresses the disagreement within a ensemble classifier. To the best of the authors' knowledge, active inference is not applied to time-evolving graphs so far.

VII. Conclusion

In this work, we discussed how active inference can foster classification in time-varying networks. We applied AFRAID, a new active inference approach for time-evolving graphs, to a real-life data set obtained from the Belgian Social Security Institution with as goal to detect companies that are likely to commit fraud in the next time period. Fraud is defined as those companies that intentionally do not pay their taxes. Given a time-varying network, we extracted (1) intrinsic features and (2) neighborhood features. A change in the label of one node might impact the feature set of the neighbors. This is collective classification. We investigated the effect on the overall performance of a set of classifiers, when we are able to select a limited set of nodes to be labeled. Although the domain requirements are rather strict (i.e., only 100 out of >200k companies can be investigated each time period), Random Forests benefit the most from active inference, achieving an increase in precision up to 15%. We investigated different probing strategies to select the most informative nodes in the network and evaluate (1) committee-based and (2) expert-based strategies. We find that committee-based strategies using high-performing classifiers result in a slightly better classification performance than expert-based strategies which is often preferred in order to obtain an unbiased set of companies for investigation. We see that the probing strategies on their own are able to identify those companies with the most uncertainty, resulting in a total precision of up to 45%.

VIII. Acknowledgements

This material is based upon work supported by FWO Grant No. G055115N, the ARO Young Investigator Program under Contract No. W911NF-14-1-0029, NSF CAREER 1452425, NSF IIP1069147, IIS 1408287 and IIP1069147, a Facebook Faculty Gift, an R&D grant from Northrop Grumman Aerospace Systems, and Stony Brook University Office of Vice President for Research. Any conclusions expressed in this material are of the authors' and do not necessarily reflect the views, either expressed or implied, of the funding parties.

References

[1] B. Baesens, *Analytics in a Big Data World: The Essential Guide to Data Science and Its Applications*. John Wiley & Sons, 2014.

[2] B. Baesens, V. Van Vlasselaer, and W. Verbeke, *Fraud Analytics Using Descriptive, Predictive, and Social Network Techniques: A Guide to Data Science for Fraud Detection*. John Wiley & Sons, forthcoming.

[3] A. Kuwadekar and J. Neville, "Relational active learning for joint collective classification models," in *ICML*, 2011, pp. 385–392.

[4] V. Van Vlasselaer, J. Meskens, D. Van Dromme, and B. Baesens, "Using social network knowledge for detecting spider constructions in social security fraud," in *ASONAM*. IEEE, 2013, pp. 813–820.

[5] V. Van Vlasselaer, L. Akoglu, T. Eliassi-Rad, M. Snoeck, and B. Baesens, "Guilt-by-constellation: Fraud detection by suspicious clique memberships," in *HICSS*, 2015.

[6] R. Rossi and J. Neville, "Time-evolving relational classification and ensemble methods," in *Advances in Knowledge Discovery and Data Mining*. Springer, 2012, pp. 1–13.

[7] U. Sharan and J. Neville, "Exploiting time-varying relationships in statistical relational models," in *WebKDD*. ACM, 2007, pp. 9–15.

[8] M. McPherson, L. Smith-Lovin, and J. M. Cook, "Birds of a feather: Homophily in social networks," *Annual Review of Sociology*, pp. 415–444, 2001.

[9] B. Settles, "Active learning literature survey," University of Wisconsin-Madison, Computer Sciences Technical Report 1648, 2009.

[10] P. Sen, G. Namata, M. Bilgic, L. Getoor, B. Galligher, and T. Eliassi-Rad, "Collective classification in network data," *AI Magazine*, vol. 29, no. 3, p. 93, 2008.

[11] H. Tong, C. Faloutsos, and J.-Y. Pan, "Fast random walk with restart and its applications," in *ICDM*. IEEE, 2006, pp. 613–622.

[12] H. Tong, S. Papadimitriou, S. Y. Philip, and C. Faloutsos, "Proximity tracking on time-evolving bipartite graphs." in *SDM*, vol. 8. SIAM, 2008, pp. 704–715.

[13] M. Sharma and M. Bilgic, "Most-surely vs. least-surely uncertain," in *ICDM*. IEEE, 2013, pp. 667–676.

[14] S. Pandit, D. H. Chau, S. Wang, and C. Faloutsos, "Netprobe: a fast and scalable system for fraud detection in online auction networks," in *Proceedings of the 16th international conference on World Wide Web*. ACM, 2007, pp. 201–210.

[15] L. Akoglu, R. Chandy, and C. Faloutsos, "Opinion fraud detection in online reviews by network effects." *ICWSM*, vol. 13, pp. 2–11, 2013.

[16] M. J. Rattigan, M. Maier, and D. Jensen, "Exploiting network structure for active inference in collective classification," in *ICDM*. IEEE, 2007, pp. 429–434.

[17] S. A. Macskassy, "Using graph-based metrics with empirical risk minimization to speed up active learning on networked data," in *SIGKDD*. ACM, 2009, pp. 597–606.

[18] M. Bilgic and L. Getoor, "Effective label acquisition for collective classification," in *SIGKDD*. ACM, 2008, pp. 43–51.

[19] ——, "Reflect and correct: A misclassification prediction approach to active inference," *ACM Transactions on Knowledge Discovery from Data*, vol. 3, no. 4, pp. 1–32, 2009.

[20] M. Bilgic, L. Mihalkova, and L. Getoor, "Active learning for networked data," in *ICML*, 2010, pp. 79–86.