Highlights

Inductive and Transductive Link Prediction for Criminal Network Analysis

Zahra Ahmadi, Hoang H. Nguyen, Zijian Zhang, Dmytro Bozhkov, Daniel Kudenko, Maria Jofre, Francesco Calderoni, Noa Cohen, Yosef Solewicz

- We study the problem of crime linkage and co-offender prediction from two perspectives: transductive and inductive.
- We compare the effect of different similarity metrics in unsupervised transductive link prediction.
- We propose an inductive and transductive link prediction framework based on graph neural networks and node attributes.
- We share a huge anonymized dataset for a burglary case.
- We share our open-source network analysis tool for network analysis, specifically helpful in criminology.

Inductive and Transductive Link Prediction for Criminal Network Analysis

Zahra Ahmadi[a], Hoang H. Nguyen[a,1], Zijian Zhang[a], Dmytro Bozhkov[a], Daniel Kudenko[a], Maria Jofre[b], Francesco Calderoni[b], Noa Cohen[c], Yosef Solewicz[c]

[a] *L3S Research Center, Leibniz University Hannover, Germany*
[b] *Università Cattolica del Sacro Cuore, Italy*
[c] *Ministry of Public Security, The Israel National Police, Israel*

Abstract

The identification of potential offenders, who are more likely to form a new group and co-offend in a crime, plays an essential role in narrowing down law enforcement investigations and improving predictive policing. Once a crime is committed, focusing on linking it to previously reported crimes and reducing the inspections based on shreds of evidence and the behavior of offenders can also greatly help law enforcement agencies. However, classical investigative techniques are generally case-specific and rely mainly on police officers manually combining information from different sources. Therefore, automatic methods designed to support co-offender research and crime linkage would be beneficial. This paper proposes two graph-based machine learning frameworks to address these issues based on a burglary use case, the first being *transductive link prediction*, which seeks to predict emergent links between existing graph nodes (which represent offenders or criminal cases), and the

[1] The first two authors contributed equally. Zahra Ahmadi conceived the original idea and experimental settings and directed the project. Hoang H. Nguyen developed the frameworks and performed most of the evaluation experiments.

other being *inductive link prediction*, where connections are found between a new case and existing nodes. Our experimental results show a prediction accuracy of 68.5% in co-offender prediction, a 75.83% predictive accuracy for transductive crime linkage, and up to 74.8% accuracy in inductive crime linkage.

Keywords: Co-offender prediction, Crime linkage, Repeat offenders, Transductive link prediction, Inductive link prediction, Deep neural networks, Machine learning

---

1. Introduction

Various theoretical and empirical studies indicate that most crimes concentrate in time and space according to the so-called "law of crime concentration" [1]. Furthermore, crime commission and victimization are concentrated among a minority of a population [2, 3]. The evidence of the concentration of crime and offenders paved the way for developing increasingly sophisticated crime prediction approaches.

Predictive policing aims to predict the risk of future crime incidents based on past crimes and other information [4]. It supports the identification and prioritization of potential targets for crime investigation through the application of advanced analytical techniques [5]. Traditionally, police officers have used simple, manual approaches to identify hotspots (i.e., areas with a higher likelihood of crime), for instance, by pinning incidents on maps [6, 7]. These approaches have been improved through the application of a variety of quantitative techniques [8] that can handle the crime prediction problem more efficiently, including various classic machine learning methods

such as Random Forests [9], Naive Bayes [10] and Support Vector Machines (SVMs) [11]. As a result of these advances, predictive policing has found several applications in law enforcement to predict the time and location of future crimes [12] while generating a lively debate about its effectiveness and potential biases [13, 14, 15]. Despite the increasing attention to predictive policing, research has rarely addressed the possibility of identifying potential offenders and, more specifically, co-offending in crime, that is, when two or more individuals participate in the same crime [4]. Yet, similar issues are central to another stream of research that focuses on crime linkage, a process of associating two or more crimes based on evidence and offender behavior [16, 17, 18]. However, this approach has largely relied on case-specific qualitative information (e.g., identifying the modus operandi of a specific offender and linking it to criminal events) and has barely evolved to more general methods [19].

As a crime generally involves an offender and a target and often occurs in a certain place and time, predictive policing techniques should answer, at least in part, one or more of the following questions: (1) who will commit a crime, (2) who will be the victim, (3) what type of crime will be committed, (4) in what location and (5) at what time will a new crime take place [20]. While most of the previous applications of predictive policing have focused on the last three questions and most crime linkage approaches have addressed the first question through case-specific methods, in this paper, we concentrate mainly on answering the first question about who will offend and with whom through advanced machine learning. In particular, the present research aims to address the following research questions:

RQ1: Knowing a network of offenders and their previous collaborations, can we predict potential future burglary attempts made by existing offenders in the network?

RQ2: Knowing the historical information of crimes and their offenders, can we narrow down the inspections of a new case to a list of potential offenders?

To this end, a comprehensive burglary dataset of more than 30,000 real case reports is processed and further transformed into a bipartite graph of offenders and criminal cases to build a network of criminals based on the collected information, which is then used to determine the co-offense likelihood for known criminals. By proposing different machine learning methods, our goal is to contribute to predictive policing and crime linkage research with a general, parsimonious, and automated approach. In doing so, we first propose an unsupervised link prediction framework that uses node neighborhoods and path information to identify possible links based solely on the network topology. On the other hand, several studies show that if labeled instances are available, supervised link prediction approaches outperform unsupervised methods [21, 22]. Although some studies have applied supervised learning for link prediction (e.g., [22, 23]), the use of these methods for co-offense prediction is still scarce [4]. Moreover, the success of the most used supervised algorithms usually depends on the data preparation and feature engineering method that properly describes the phenomenon. Therefore, we attempt to overcome these limitations by exploring more advanced analytical methods, specifically deep neural networks. Since we have prior knowledge of the offending history of criminals, we call this approach *transductive link pre-*

diction. Furthermore, we propose an *inductive link prediction* framework to assess whether offenders of a newly placed crime can be predicted based on the textual reports of the crimes presented as node attributes.

The rest of the paper is organized as follows: Section 2 discusses the related work on machine learning for crime prediction, especially in the case of burglary. Section 3 introduces the burglary use case and describes the data process employed to generate network data. Section 4 explains both transductive and inductive link prediction methods for offenders and crime linkage. Section 5 presents different baseline approaches, explains the metrics used for evaluation, specifies the parameter setting for further modeling, reports the main results of both approaches, and discusses them accordingly. Finally, Section 6 concludes and proposes future work.

2. Related Work

There is a growing interest in using machine learning in criminology and crime research, particularly crime prediction [24]. Most methods focus on predicting the time and/or location of future crimes, while a few studies aim to find connections and links among the crimes. In this section, we review the models based on machine learning to deal with the crime linkage problem and general prediction of the crime.

2.1. Crime Linkage and Machine Learning Methods

Crime linkage studies are generally based on the similarity of criminal behavior, which relies on three assumptions [25, 26]: (1) criminal behavior is consistent, that is, the same offender will behave similarly over time, (2) different criminals show distinctive criminal behaviors that are different

from each other, and (3) criminal behavior is measurable through a direct relationship and homology between the characteristics of offenders and their behavior through quantitative models.

The crime linkage problem can be seen as a binary classification task that aims to find serial crimes committed by the same offenders. In this setting, it is evaluated whether each of all possible crime pairs represents a serial crime pair or not. Due to insufficient evidence, it is sometimes difficult to determine if a crime is serial, so these two-way decisions are prone to error. A recent study attempted to model the problem with a three-way decision, where the data space is divided into three possible regions (i.e., positive, negative, and boundary) based on two thresholds, so those samples that are difficult to distinguish given the existing information are placed in the boundary area [27]. Then, they tried to automatically learn the thresholds of the three-way decisions without the need to establish explicit loss functions. Nonetheless, in many real-world cases, serial crime pairs are much less common than non-serial crime pairs. To address this challenge, some studies applied class imbalance algorithms [28]. In a particular robbery case, they focus on the indistinguishable case pairs at the classification boundary rather than resampling smaller or larger classes to handle the imbalanced dataset. Chi *et al.* also presented a decision system that determined if two robberies belong to the same series of cases [29]. However, their system is quite limited in that it considers only robbery cases and requires police officers to manually mark the characteristics of each case.

Due to the increasing attention to deep learning methods and their promising results in various applications, a recent study uses an adaptive deep

Q-learning network with reinforcement learning to develop a robust crime prediction model [30]. In another work, Wang *et al.* extended the ResNet model [31] for spatiotemporal crime forecasting [32]. A crime forecasting system using an attention-based sequence-to-sequence model and convolutional variational autoencoders has also been proposed [33]. Previously, Simonyan and Zisserman presented a two-stream deep learning approach that learned video representations by dividing video streams into two components, one representing a spatial stream and the other a temporal one [34]. They used a Convolutional Neural Network (CNN) architecture to identify spatial dependencies, further enhanced by a Long-Short Term Memory (LSTM) that captured temporal patterns. The joint use of CNNs and LSTMs showed highly complementary behavior in capturing spatial and temporal features for video classification [35]. Later, Solomon *et al.* proposed machine learning-based approaches for crime linkage to parameterize crimes with spatiotemporal information and automatic and manual language features extracted from police reports [19]. In doing so, they used two types of samples: positive pairs, which were pairs of burglaries committed by the same criminal, and negative pairs, which were pairs of burglaries committed by different criminals. Then, the model predicted crimes committed by new criminals that were not observed in the training phase. Ghazvini *et al.* also proposed a model that detected serial crimes by isolating short-term repetitiveness using neural networks [36].

All related work presented so far relates to transductive link prediction, where approaches have aimed to build new connections between known offenders. Inductive link prediction, by contrast, focuses on cases where new

offenders are added to the scope. Expanding the analysis beyond the study of crime, Bojchevski and Günnemann [37] described network nodes using a Gaussian distribution instead of a simple low-dimensional vector used in previous studies (e.g., [38]). Then, a dissimilarity measure with respect to the Gaussian distribution was defined to minimize the heterogeneity of adjacent nodes: each time a new node was added to the network, the approach would predict its links to known nodes while minimizing the dissimilarity between the connected ones.

2.2. Crime Prediction with Machine Learning Methods

Crime prediction methods in the literature have largely ignored the role of co-offending in committing a crime. Instead, they have focused on modeling observed crimes spatially and temporally to predict the time and location of future crimes. In general, crime prediction methods can be divided into two categories: traditional empirical methods and spatiotemporal methods. Spatiotemporal models, including time series [39] and Kernel Density Estimation (KDE) [40], are commonly applied for crime prediction and are related to the crime linkage problem. Variously, traditional methods consider time and location independently [41], and aim to focus on predicting crime hotspots and crime risk areas [42]. These methods are less relevant to our study as they ignore the connection between the crimes. However, we provide a brief overview of this literature to review the machine learning methods used in this field.

As for early studies, Olligschlaeger examined using Multi-Layer Perceptrons (MLP) in GIS systems to predict drug-related calls at 911 call centers in Pittsburgh, USA [43]. He trained a simple MLP architecture with only

nine neurons and a single hidden layer on a dataset with three collected features indicating the number of calls received in each map cell area related to weapons, robberies, and assaults. Later, Gorr *et al.* compared different regression approaches to predict a set of crime categories using Pittsburgh data [44]. They ran regression functions of different complexity on the same data and found that more sophisticated methods outperformed simple time series. In particular, they found that the predicted mean absolute error was improved through a smoothing coefficient, that is, by applying more weight to recent data. However, results from the Prophet model [45] applied to crime occurrence datasets from three major US cities showed that time series models could outperform neural networks.

SVMs have shown successful results in various applications, including hotspot location prediction [46]. Yu *et al.* compared the competency of SVMs to other well-established machine learning approaches like Naive Bayes and Random Forests [47]. They found supportive evidence aligned with the theory of the recurrence of residential burglary at a particular place. In some types of crimes, such as burglaries, serial offenders remain in a certain area due to familiarity with the region, even though the proximity to their home could compromise their anonymity. Thus, their incidents are concentrated within a ring-shaped area centered on an outbreak point [48]. Furthermore, Mohler *et al.*[49] and Ratcliffe [50] proposed two different temporal crime modeling approaches and gained insights that validated the well-known claim that offenses can be driven by the availability of opportunities.

Many of the methods in the literature rely solely on spatial dimensions of the incidents, including KDE [51]. Originally, Silverman used KDE to divide

the study area into grids of regular cells and estimated a density value for each cell via a kernel function that estimates the probability density of crime incidents [52]. By extending KDE to employ space and time variants, Nakaya and Yano proposed a crime cluster analysis with a temporal dimension that can simultaneously visualize the geographical extent and duration of criminal clusters [40]. Going one step further, Toole *et al.* used criminal records to identify spatiotemporal patterns at multiple scales [53]. They employed various quantitative tools to identify significant correlations across both space and time in the behavioral crime data.

3. Data Collection and Network Creation

3.1. Burglary Dataset

The dataset used in this study, provided by the Israel National Police and duly anonymized according to the standards of Israeli national law and GDPR and further approved by a legal advisor of the Israel National Police, contained around 30,000 reports of solved burglary cases that occurred in Israel between 2012 and 2021. The information contained in the reports included a crime identifier, the respective anonymized identifiers of the offenders, the min-max-scaled site coordinates of the crime that prevent precise retrieval of the localization of the site, timestamps, and a parameterized free-text description of the case in the form of an embedding vector, in particular, a SIF embedding [54]) described next. Even though we used a burglary database similar to Solomon *et al.* [19], our study employed a completely different link prediction approach as they looked at the link prediction task from a classical machine learning perspective, while our work is framed based on

graphs, where crimes and offenders are nodes represented by features[2], hence explicitly considering the structure of the criminal network and not only the features of the sample.

Text Embedding: The properties of the crime nodes in our graphs are transformed into embedding vectors built from the textual description of the incident. Even though the spatiotemporal information of a crime is very important for link detection and prediction, we have ignored these features, which could be seen as a routine independent investigative filtering procedure. Similar to Solomon *et al.* [19], we parameterized the textual information of the crime using Smooth Inverse Frequency (SIF) [54] [3]. The SIF method can encode sequences of words in a sentence or paragraph into a single vector, mathematically representing the crime description. In short, this method intelligently combines the embeddings of each word within a sequence to identify the most relevant ones, a simple semantic text similarity task that has proven to work well [56].

[2]While this study uses only criminal characteristics, future research could also incorporate criminal characteristics.

[3]More sophisticated alternatives to text embedding, such as those based on Bidirectional Encoder Representations from Transformers (BERT) [55], have recently been proposed. However, since the optimization of text embeddings as node attributes is not the main goal of this article, we followed the suggestion of Solomon *et al.* to use SIF instead of BERT. Their findings could be due to the fact that the pretrained AlephBert did not fit the current corpus, while SIF relied on a domain-specific word2vec model trained on a large corpus of our criminal texts. Since police reports lacked the proper language formalism, we believe that the benefits of BERT are diminished, especially when it was not fine-tuned to our data. Further elaboration on text embedding optimization is suggested for future work.

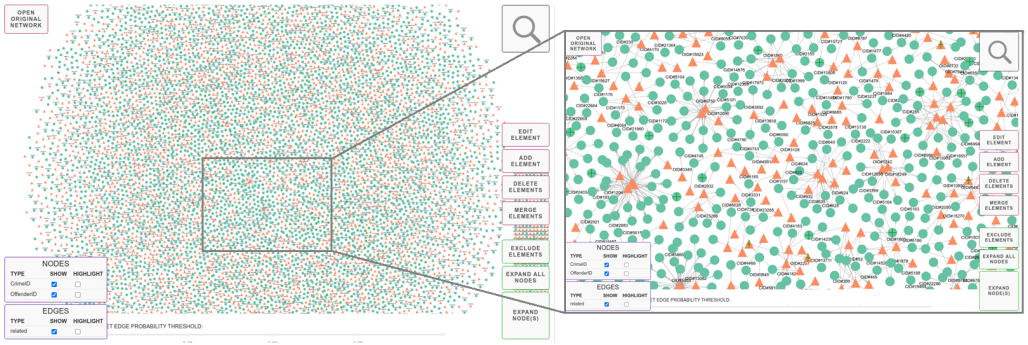


Figure 1: The crime-offender bipartite network[4]. Green circles and orange triangles represent crime cases and offenders, respectively. Edges indicate when an offender participated in a crime.

3.2. Generated Networks

Three different networks were generated based on the original burglary data and text embeddings, including:

1. A *crime-offender network* (Figure 1) with 41,324 nodes and 34,156 edges, where nodes represent both crimes and offenders, and links indicate whether an offender participated in a crime. This network is a bipartite graph with two different types of nodes later divided into two networks for further analysis.
2. An *offender network* (Figure 2), where the nodes represent the offenders and the links indicate whether two offenders are involved in one or more burglary cases. To generate the offender graph, we used the original undirected crime-offender network to connect two offender nodes

[4]Snapshots are taken using our open source network analysis tool, which can be downloaded from `https://github.com/erichoang/criminal-network-visualization`. We present and describe the details of the tool in Section 7.1.

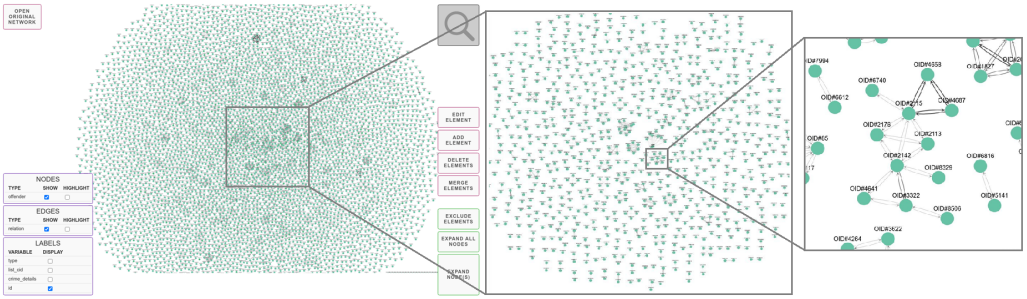


Figure 2: The offender network, in which nodes represent offenders, and edges indicate if two offenders share a crime.

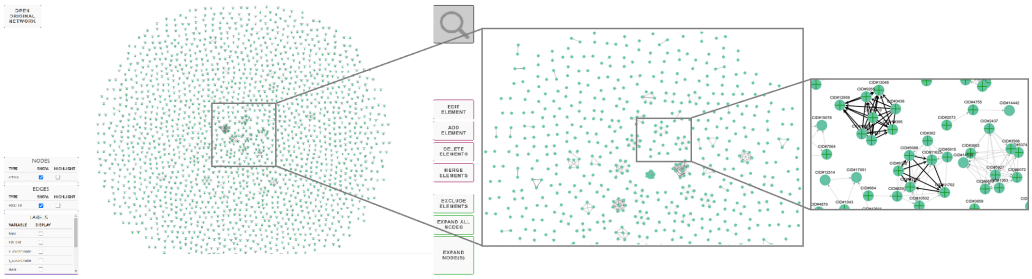


Figure 3: The crime network. The nodes represent placed crimes and the edges indicate whether two crimes share an offender.

each time they shared a crime and then filtered the crime nodes. The offender network resulted in a total of $17,232$ nodes and $21,302$ edges. We used this dataset in the link prediction experiments as the network is large enough for this purpose. Furthermore, we specifically conducted the unsupervised transductive link prediction experiments on this dataset as we only accessed the network structure and not additional embedded attributes for offenders.

3. A *crime network* (Figure 3) with nodes representing offenses and edges indicating the number of shared offenders between the crimes. This net-

work included 23,380 nodes and 42,604 edges. It was used for link prediction experiments, especially inductive link prediction, since nodes, i.e., crime cases, consisted of embedded descriptions as attributes.

4. Link Prediction Methods

We propose two approaches for link prediction: *transductive link prediction* (Section 4.1), which considers prior knowledge of the offender network to predict emerging links between existing nodes (i.e., offenders), and *inductive link prediction* (Section 4.2), where the links of a new crime to existing crimes can be predicted based on available meta-information (i.e, node attributes).

4.1. Transductive Link Prediction

In principle, all link prediction methods work by assigning a $score(u, v)$ to pairs of nodes (u, v) and then choosing the top-scored pairs as predicted links for u [57]. The score value is computed based on the input network and often measures the proximity or similarity between u and v. Different measures of similarity have been proposed, the most relevant in the literature and applicable for crime prediction being the following:

- **Jaccard similarity** measures the probability that both u and v have past connections to individual f. For this, an individual f is randomly selected from a set of all individuals with past connections to either u or v. The probability is then calculated as the ratio of the number of individuals that both u and v have observed interactions with and the number of individuals with whom either u or v have observed

interactions. In particular:

$$Jaccard(u, v) = \frac{|\Gamma(u) \cap \Gamma(v)|}{|\Gamma(u) \cup \Gamma(v)|}, \tag{1}$$

where Γ indicates the neighborhood function of a node.

- **Adamic Adar similarity** [58] is a variant of the Jaccard ratio, where similarity is determined as a weighted count of the individuals with whom both u and v have had interactions. Each individual is weighted by the inverted number of all other observed individuals:

$$Adamic(u, v) = \Sigma_{z \in \Gamma(u) \cap \Gamma(v)} \frac{1}{\log |\Gamma(z)|}. \tag{2}$$

  In this setting, a popular individual with many connections other than to u and v contributes less to the similarity ratio than an individual f who only has past connections to u and v. It is reasonable since, in this case, f could probably be the driver of the relationship between u and v.

- **Preferential attachment similarity** [59] follows what is called the Rich-Get-Richer phenomenon. This means that the probability that u will connect with v in the future is proportional to the popularity of these two individuals. In its simplest mathematical form, this measure is determined by the product of the number of other individuals with whom u has had interactions and the same number with v:

$$Preferential(u, v) = |\Gamma(u)||\Gamma(v)|. \tag{3}$$

- **Resource allocation similarity** [60] aims to measure resources flowing from u to v through other people with whom u and v have connections. It has a very similar mathematical formulation to the Adamic

Adar similarity, except that the number of other individuals is now weighted by their individual interactions:

$$Resource\_Allocation(u, v) = \Sigma_{z \in \Gamma(u) \cap \Gamma(v)} \frac{1}{|\Gamma(z)|}. \quad (4)$$

- **Soundarajan-Hopcroft similarity** [61] is an improved variant of Adamic Adar similarity that considers information about the community structure of the input network. The rationale is that the more community members the two individuals have in common, the more likely they are to form some sort of connection. Mathematically, the score is determined by the number of individuals both u and v have interacted with, plus the number of communities they both belong to:

$$Soundarajan\_Hopcroft(u, v) = \Sigma_{z \in \Gamma(u) \cap \Gamma(v)} \frac{f(z)}{|\Gamma(z)|}, \quad (5)$$

  where $f(z)$ equals 1 if z belongs to the same community as u and v.

These measures are based on the connections between individuals in the network, so they can be improved by considering the attributes and behavior of individuals when calculating the similarity values. Several works have leveraged this approach. The tensor factorization method, for instance, has gone hand in hand with a high computational cost considering its technical complexity, which is why we have discarded it in this study.

4.1.1. Prediction by Transductive Algorithm

Our approach for similarity-based transductive link prediction is simple yet effective. For each pair of nodes, (v, u), in the node set V, we first calculate their similarity $\text{sim}(v, u)$ using one of the following similarity metrics:

Algorithm 1 Transductive Link Prediction Algorithm

Require: $G = (\mathcal{V}, \mathcal{E})$, k, sim ▷ sim represents the similarity metric function

```
Calculate sim(u, v), ∀u, v ∈ V
pred ← []
for v ∈ V do
    for u ∈ select-top(sim(v, ·), k) do        ▷ select the top k similar nodes
        pred ← pred + (v, u)
    end for
end for
Return pred
```

the Jaccard coefficient, Adamic-Adar index, resource allocation index, preferential attachment or the Soundarajan-Hopcroft coefficient. Then, for each node, the top k similar nodes regarding the computed similarity metric are selected, which indicates the target nodes of the outgoing edges of that particular node. The pseudo-code of our transductive approach is presented in Algorithm 1.

4.2. Inductive Link Prediction

In attributed graphs, both the network structure and attribute information can be used for link prediction. An attributed graph is represented by $G = (\mathcal{V}, \mathcal{E}, \mathcal{X})$, where $\mathcal{V} = \{v_1, ..., v_n\}$ represents the node set, $n = |\mathcal{V}|$, $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$ the edge set, $\mathcal{X} = \{x_1, ..., x_n\}$ the attribute feature matrix, $x_i \in \mathbb{R}^m$ the attribute feature vector of node v_i and m the number of attributes in the graph. Given a graph G and a pair of nodes (v_i, v_j), the goal of link prediction is to estimate the likelihood of a link exists between v_i and

v_j. Most current methods focus on transductive link prediction, where both nodes v_i and v_j are already known. However, in many real-life situations, inductive prediction is also required considering new nodes, where attribute information is available, but one or both of the nodes v_i and v_j have not been observed, at least during the training process.

To overcome this limitation, we implement an attributed graph embedding method called Dual-Encoder graph embedding with ALignment (DEAL) [62], which can be used for both inductive and transductive link prediction. This framework embeds the graph of existing nodes in the vector space and extracts its structure information (i.e., structure-oriented node embedding). It then computes an embedding vector for new query nodes where the only information available is their attributes (i.e., attribute-oriented node embedding), which is ultimately compared to the previously computed structure embedding. In particular, the DEAL approach involves three main components, including two types of node embedding encoders and one alignment mechanism. The first encoder aims to output the attribute-oriented node embedding ($\mathcal{H}_a$), while the second relates to the structure-oriented node embedding ($\mathcal{H}_s$): $\mathcal{H}_a$ computes the embedding vectors with attributes of the new nodes and $\mathcal{H}_s$ computes the node embedding vector that preserves the structure information. Finally, the alignment mechanism is employed to contrast the two embedding vectors and build the connections between the attributes and the links. Both encoders are updated during training, so the generated embeddings are aligned.

4.2.1. Attribute-Oriented Encoder

The attribute-oriented encoder $\mathcal{H}_a$ ingests an attribute vector x_i from node v_i and generates a node embedding $z_i^a = \mathcal{H}_a(x_i)$. We used an MLP with a nonlinear activation layer to learn $\mathcal{H}_a$:

$$\mathcal{H}_a(x_i) = \sigma(\mathbf{W}_a^2(\sigma(\mathbf{W}_a^1 x_i + b_a^1)) + b_a^2), \tag{6}$$

where W_a^1, W_a^2, b_a^1 and b_a^2 are hyperparameters of the model and $\sigma(.)$ is the exponential linear unit. We opt for a simple MLP approach instead of complex neural networks since well-known Graph Neural Networks (GNN) models such as Graph Convolutional Networks (GCN) [63] and Graph Attention Networks (GAT) [64] suffer from scalability limitations (see Section 5.2).

4.2.2. Structure-Oriented Encoder

The structure-oriented encoder $\mathcal{H}_s$ generates node embeddings that only preserve the structural information of the graph G without regard to the attributes of the nodes. We use the one-hot encoding of the nodes $\mathcal{I}_\mathcal{V} = \{I_1, ..., I_n\}$ as the input of the encoder and further map node v_i to its node embedding vector $z_s^i = \mathcal{H}_s(I_i)$. We then employ a linear model to compute the encoder $\mathcal{H}_s$:

$$\mathcal{H}_s(I_i) = g(W_s)I_i, \tag{7}$$

where $g(.)$ is used to re-parameterize W_s and accelerate the convergence of stochastic gradient descent optimization.

4.2.3. Alignment Mechanism

We align the embeddings of the two types of encoders with learning the connections between the node attributes and the graph structure. In doing

so, we apply a ranking-motivated loss function that learns the graph embedding through ranking, which can help capture the relationships between each pair of nodes in the training samples. Inspired by the contrastive loss [65] that maps similar input samples to nearby points in the output vector space and dissimilar samples to distant points, we map the linked graph nodes to close points in the output vector space and the unlinked nodes to points far apart. However, it should be considered that the unlinked nodes (negative pair-wise samples) have different distances in the graph. Consequently, we employ the following loss function for a given mini-batch of node pairs $B = \{(v_{p_1}, v_{q_1}), ..., (v_{p_k}, v_{q_k})\}$, where $p_i \neq q_i$ and $i \in [1, k]$:

$$\mathcal{L}_B(\mathcal{Z}) = \frac{1}{|B|}\Sigma_{(v_{p_i}, v_{q_i})\in B}[(1 - y_i)\alpha(v_{p_i}, v_{q_i})\phi_1(-s(z^{p_i}, z^{q_i})) + y_i\phi_2(s(z^{p_i}, z^{q_i}))], \tag{8}$$

where $s(.,.)$ represents a similarity function that compares two node embeddings (z^{p_i} and z^{q_i}) and y_i the link relation label (i.e., $y_i = 1$ if two nodes are connected). In this paper, we use the so-called cosine similarity function. Further, α represents a weight function to measure the importance of negative samples with different distances. We can define it as $\alpha(v_p, v_q) = \exp \frac{\beta}{d_{sp}(v_p, v_q)}$, where $d_{sp}(.)$ denotes the shortest path between two nodes and $\beta > 0$ is a hyperparameter. If two nodes are unreachable, then $d_{sp}(v_p, v_q) = \infty$. Both ϕ_1 and ϕ_2 are derived from function $\phi(.)$, which considers different hyperparameters to link regularization. We then use the generalized logistic loss function ($\phi(x)$, with $\gamma > 0$ and b as loss margin parameters) to tune regularization:

$$\phi(x) = \frac{1}{\gamma}\log(1 + \exp^{-\gamma x + b}). \tag{9}$$

Instead of optimizing $\mathcal{L}_B(\mathcal{Z}_s)$ and $\mathcal{L}_B(\mathcal{Z}_a)$ independently, we design a

Algorithm 2 Inductive Link Prediction Algorithm (DEAL)

Require: Graph $G = (\mathcal{V}, \mathcal{E}, \mathcal{X})$, a set of mini-batches B, loss weight θ

for each batch in B **do**

$\mathcal{Z}_s \leftarrow \mathcal{H}_s(\mathcal{I}_\mathcal{V})$

$\mathcal{Z}_a \leftarrow \mathcal{H}_a(\mathcal{X})$

$\mathcal{L} \leftarrow \theta.[\mathcal{L}_B(\mathcal{Z}_s), \mathcal{L}_B(\mathcal{Z}_a), \mathcal{L}_{align}(\mathcal{Z}_s, \mathcal{Z}_a)]$

Update $\mathcal{H}_s$ and $\mathcal{H}_a$ with stochastic gradient $\nabla\mathcal{L}$

end for

Update $\mathcal{Z}_s$ and $\mathcal{Z}_a$ via $\mathcal{H}_s$ and $\mathcal{H}_a$

Calculate score$(u, v), \forall u, v \in \mathcal{V}$ via Equation 12

joint alignment method that aims to maximize the similarity between $z_s^{p_i}$ and $z_a^{q_i}$ of two linked nodes v_{p_i} and v_{q_i}:

$$\mathcal{L}(\mathcal{Z}_s, \mathcal{Z}_a) = \frac{1}{|B|}\Sigma_{(v_{p_i}, v_{q_i})\in B}[(1-y_i)\alpha(v_{p_i}, v_{q_i})\phi_1(-s(z_s^{p_i}, z_a^{q_i}))+y_i\phi_2(s(z_s^{p_i}, z_a^{q_i}))]. \tag{10}$$

The overall objective of our inductive model follows:

$$\mathcal{L} = \theta_1\mathcal{L}_B(\mathcal{Z}_s) + \theta_2\mathcal{L}_B(\mathcal{Z}_a) + \theta_3\mathcal{L}_{align}(\mathcal{Z}_s, \mathcal{Z}_a), \tag{11}$$

where $\theta_1, \theta_2, \theta_3$ are hyperparameters to parameterize the weights of different losses further.

4.2.4. Link Prediction

To determine if there is a link between two nodes v_p and v_q exists, we calculate the following score:

$$score(v_p, v_q) = \lambda_1 s(z_s^p, z_s^q) + \lambda_2 s(z_a^p, z_a^q) + \lambda_3 s(z_s^p, z_a^q), \tag{12}$$

where $\lambda_1, \lambda_2, \lambda_3$ are hyperparameters of the similarity score. In inductive link prediction, the link z_a^q to a new node v_q is calculated by setting $\lambda_1 = 0$. Algorithm 2 summarizes the steps in the DEAL framework. DEAL can also perform transductive link prediction by specifying λ_1 to a non-zero value.

5. Experimental Analysis

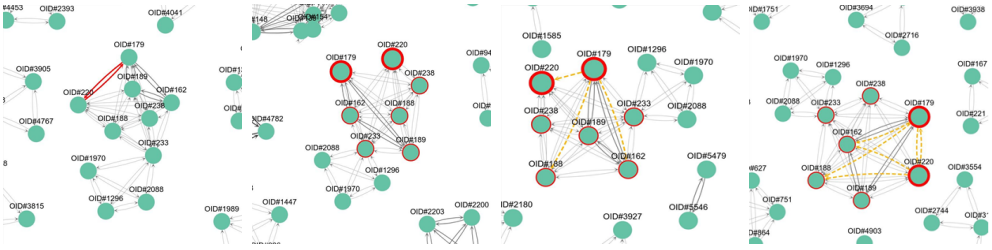
The performance of link prediction methods is often evaluated by their ability to retrieve links hidden on purpose. Accordingly, we first conceal a small portion of the links for a given individual and then use the prediction methods to identify potential hidden links. Section 5.1 and 5.2 discuss our findings on transductive[5] and inductive[6] link prediction experiments, respectively.

5.1. Transductive Link Prediction Results

Figure 4 shows an example of our transductive link prediction using the Jaccard similarity measure. Figure 4a shows the original network where two connected nodes, $OID\#179$ and $OID\#220$, are randomly selected. To evaluate our link prediction algorithm, we remove the existing links using the available options in our visualization platform. Then, we apply the transductive link prediction with Jaccard similarity sequentially on $OID\#179$ and $OID\#220$. The method returns the top k likely emerging edges (here, top three). Figure 4d shows that the method can correctly predict the links between the two nodes.

[5] `https://github.com/erichoang/criminal-network-visualization`

[6] `https://github.com/erichoang/criminal-link-prediction`



(a) (b) (c) (d)

Figure 4: (a) Original offender network and two randomly selected nodes ($OID\#179$ and $OID\#220$), (b) Modified network by removing the links between the two selected nodes, (c) Transductive link prediction on $OID\#179$ in a modified offender network, (d) Transductive link prediction on both $OID\#179$ and $OID\#220$ in the modified network. The link prediction method uses the Jaccard similarity measure, and the predicted links are the yellow-dash lines.

Table 1 reports the results of the transductive link prediction on the offender network by randomly selecting a sample of links to be removed from the original network (1%, 5%, 10%, and 15%). The link prediction scores between the central nodes and their 2-hop neighbors (the so-called *candidate edges*) are computed for each removed link based on the respective similarity measure. For each setting (determined by the specific similarity measure and the percentage of links removed), accuracy will naturally increase if we search for the correct link among more potential links (i.e., larger k). Moreover, for a fixed similarity measure and a fixed k, we can see the accuracy degrades with more links removed as the network structure changes more and we have less knowledge between offenders. The only exception is the Preferential attachment measure when the removed links increase from 1% to 5%, where accuracy improves slightly. It can also be observed that Jaccard similarity achieves the best accuracy on top-1 predicted links for all cases (i.e., offender

Link Removal	Top-k	Jaccard similarity	Adamic Adar	Preferential attachment	Resource allocation	Soundarajan-Hopcroft
1%	1	**61.78 (1)**	61.03 (3)	28.69 (5)	60.84 (4)	61.12 (2)
	3	67.10 (4)	**67.38 (1)**	46.92 (5)	**67.38 (1)**	67.20 (3)
	5	68.32 (2)	68.32 (2)	58.32 (5)	68.32 (2)	**68.60 (1)**
5%	1	**58.69 (1)**	54.84 (3)	28.50 (5)	54.73 (4)	56.64 (2)
	3	65.87 (2)	65.52 (3)	47.1 (5)	65.50 (4)	**66.34 (1)**
	5	67.49 (4)	67.56 (2)	59.38 (5)	67.54 (3)	**68.03 (1)**
10%	1	**53.1 (1)**	48.50 (3)	27.67 (5)	48.37 (4)	50.23 (2)
	3	**62.30 (1)**	60.85 (3)	46.41 (5)	60.81 (4)	62.03 (2)
	5	64.11 (2)	63.70 (3)	57.87 (5)	63.66 (4)	**64.35 (1)**
15%	1	**48.36 (1)**	43.59 (3)	27.02 (5)	43.47 (4)	45.24 (2)
	3	**58.78 (1)**	57.27 (3)	45.54 (5)	57.18 (4)	58.20 (2)
	5	60.96 (2)	60.64 (3)	56.10 (5)	60.59 (4)	**61.02 (1)**

Table 1: Accuracy of transductive link prediction on five different similarity measures and their ranking. The best results are in bold.

network with different ratios of removed links). Finally, the Soundarajan-Hopcroft similarity measure slightly improves the Jaccard similarity results on the top five predicted links in all test cases.

5.2. Inductive Link Prediction Results

In this experiment, we train the DEAL framework on the crime network. We first explain how data is split in Section 5.2.1, and then we describe the comparison methods and their settings in Section 5.2.2 and 5.2.3, respectively. Performance is measured by known metrics such as the Area Under the Receiver Operating Characteristic Curve (ROC-AUC) and average precision at the top-ranked predicted link level [57]. Results are finally discussed in Section 5.2.4.

5.2.1. Data Splitting

We split the input burglary network into three subsets:

1. Training set: Burglary cases before 2019.01.01 (75% with the number of nodes being $17,531$)
2. Validation set: Burglary cases from 2019.01.01 to 2020.01.01 (15% with $3,488$ nodes)
3. Test set: Burglary cases after 2020.01.01 (10% with $2,361$ nodes)

For the data processing and the creation of the subnetworks, we only consider the training set that does not include the nodes and edges of the validation and test sets. The validation set is used for fine-tuning the DEAL hyperparameters, while the test set is left for evaluation purposes. For all subsets, we consider the text embeddings of the burglary case summaries as node attributes. We run the experiments in two versions:

- **Training with transductive validation**, where both training and validation sets are combined in the training process. In other words, the validation set is a subset of the training set, but there are no links between its nodes and the nodes of the training set.
- **Training with inductive validation**, where the network used in the training process is completely independent and different from the validation set network.

5.2.2. Methods of Comparison

We implement several baselines and state-of-the-art methods to compare their performance against our method on the burglary dataset. For the inductive link prediction, the following comparison approaches are considered:

- **Radius Neighbours:** This simple implementation predicts which newly-added nodes will be connected to their nearest neighbor nodes

within the ball in the node embedding space. The radius of the nearest-neighbors ball is defined using the embedding of the training data (i.e., the nodes that are already in the network), so that the prediction accuracy on the training dataset is maximized.

- **Linear Model:** We consider the link prediction as a classification task, where the classifier differentiates the node pairs connected by an edge from the pairs that are not. We first augment the input data by considering any subset of data from Section 5.2.1, selecting *all* node pairs (u, v) where an edge $e = (u, v)$ exists and concatenating their vector representations into a new vector. Such concatenations are mapped to the positive class, i.e., $y = 1$. Then, we *randomly* sample negative pairs from each subset, where there are no edges in between, and set their $y = 0$. Finally, we train different linear classification methods, such as LASSO, Ridge, and SVM, on the augmented training set $< (u, v), y >$ and make inferences on the test set. Note that this approach does not have newly added node w, so only transductive experiments apply.

- **Graph2Gauss:** As mentioned in Section 2, the Graph2Gauss [37] method trains a Gaussian model over the training network and further expands it using the newly added nodes so that the dissimilarity with respect to the Gaussian model is minimized between all connected nodes.

- **DeepWalk:** DeepWalk [66] considers short truncated random walks on the graph as a language corpus and vertices of that graph as vocabulary. It uses a skip-gram language model.

- **Node2Vec:** Similar to DeepWalk, Node2Vec [67] uses a skip-gram language model over a graph to learn its structure. In Node2Vec, preferences for depth-first or breadth-first sampling can be specified.

- **LINE:** The LINE embedding method [68] attempts to represent nodes as low dimensional embeddings that combine first-order (direct connection between nodes with strength weight) and second-order (direct neighborhood overlap between two nodes) proximity. The algorithm employs sampling of the second-order relationships to improve efficiency and make learning on datasets with millions of nodes and billions of edges feasible.

- **Graph Convolutional Network:** GCNs [63] use a message-passing algorithm to propagate information between neighboring nodes. They typically employ multiple graph convolutional layers to learn increasingly complex features. Generally, GCN extends the capabilities of CNN by incorporating the Laplacian matrix as a first-order approximation for the propagation between the layers of spectral graph convolutions.

- **Graph Attention Network:** GATs [64] employ an attention mechanism that assigns different importance to neighbor nodes. This allows the model to aggregate information from different parts of the graph efficiently. The attention mechanism dynamically adjusts to each node's neighborhood, enabling the model to adapt to varying connectivity patterns and handle irregular structures. Their ability to capture local and global contextual information makes them particularly well-suited

to complex graph data tasks.

5.2.3. Parameter Setting

For training our method as well as Graph2Gauss, we augment the training network into multiple triples (u, v, w), where u and v are connected while w is a "negative sample" drawn from those nodes that do not connect to neither u nor v. For validation purposes, we aim to predict whether there is a connection between each new-old pair of nodes from the test set. Further, we configure the parameters for the baseline models as follows:

- **Radius Neighbours:** The radius of the nearest-neighbors ball is 0.74, which is the optimal value that minimizes the prediction error on the training set.

- **Linear Models:** Three linear classification models are implemented, including LASSO, Ridge, and the linear SVM model. We adopt the sklearn[7] implementation of these three algorithms maintaining the default parameters. For Ridge and LASSO, we set $\alpha = 1$. For the SVM classifier, the regularization parameter is set as $C = 1$ and the radial basis function (RBF) as the kernel (see [69]).

- **Graph2Gauss:** In the Graph2Gauss method, the Gaussian model for a single node is built based on the node's embeddings and its k-hop neighbors. We follow the same parameter setting as in [37] and set the count of maximum hop $k = 2$. We implement a two-layer perceptron to represent each node using the mean and covariant of a Gaussian

[7] `https://scikit-learn.org/stable/index.html`

distribution in $L = 64$ dimensions. We then train the model for 500 epochs and use the Adam optimizer with a learning rate of 0.01 and no weight decay.

- **DeepWalk:** The embeddings are trained over 100 epochs with window size 10 and $K = 3$ negative samples. Overall, $n = 10$ random walks are simulated with walk lengths of $l = 80$.

- **Node2Vec:** We use the skip-gram language model with a window size of $win = 10$ and $K = 3$ negative samples over 100 epochs. $n = 10$ random walks of length $l = 40$ with return parameter of $p = 1$ and in-out parameter of $q = 4$ are simulated in order to build the corpora.

- **LINE:** The LINE model is trained for 50 epochs with a batch size of 1024. The Adam optimizer is used for gradient descent with a learning rate of 0.001 and no weight decay. The number of negative samples is set as $K = 3$, and second-order proximity is used.

- **Graph Convolutional Network**: We use the authors' recommended settings in the GCN model. Specifically, the model is trained for 200 epochs and uses the Adam optimizer with a learning rate of 0.01, two hidden layers, a dropout being 0.5, and no weight decay. Since the burglary network is large, we utilized two decomposed layers [70] to solve memory issues in our GCN experiments.

- **Graph Attention Network**: The GAT model, featuring eight attention heads, is trained for 200 epochs. Gradient descent is performed using the Adam optimizer, with a learning rate of 0.01, two hidden

	Methods	Featured Parameters	ROC-AUC	Average Precision
Conventional methods	Radius Neighbours	Maximum radius $r = 0.74$	-	0.5146
	LASSO	Regularization parameter $\alpha = 1$	-	0.5000
	Ridge	Regularization parameter $\alpha = 1$	-	0.6114
	SVC	Regularization parameter $C = 1$	-	0.6065
	Graph2Gauss	$d_{\text{output}} = 64, d_{\text{hidden}} = 64, n_{\text{hidden layer}} = 2$	0.6711	0.6556
Combination of basic graph embeddings and case summary text embeddings	Node2vec + SIF	Pooling function = avg	0.568	0.5631
	DeepWalk + SIF	Pooling function = avg	0.5837	0.5833
	LINE + SIF	Pooling function = avg	0.533	0.5333
	GCN + SIF	Learning rate = 0.01, $d_{\text{hidden}} = 64, n_{\text{decomposed layers}} = 2$	0.6206	0.6207
	GAT + SIF	Learning rate = 0.01, $d_{\text{hidden}} = 64, n_{\text{heads}} = 8$	0.6128	0.6076
Variants of DEAL framework	DEAL-tr (default setting)	$\theta = [0.1, 0.85, 0.05]$, $\lambda = [0.1, 0.85, 0.05]$	0.7580	0.7567
	(increase θ)	$\theta = [0.2, 1.7, 0.1]$, $\lambda = [0.1, 0.85, 0.05]$	0.7582	0.7569
		$\theta = [0.4, 3.4, 0.2]$, $\lambda = [0.1, 0.85, 0.05]$	**0.7583**	**0.7571**
	(change λ)	$\theta = [0.1, 0.85, 0.05]$, $\lambda = [0.2, 1.7, 0.1]$	0.7580	0.7567
	(change both)	$\theta = \lambda = [0.05, 0.425, 0.025]$	0.7576	0.7561
		$\theta = \lambda = [0.2, 1.7, 0.1]$	0.7582	0.7569
		$\theta = \lambda = [0.4, 3.4, 0.2]$	**0.7583**	**0.7571**
	DEAL-ind (default setting)	$\theta = [0.1, 0.85, 0.05]$, $\lambda = [0.1, 0.85, 0.05]$	0.7468	0.7477
	(increase θ)	$\theta = [0.2, 1.7, 0.1]$, $\lambda = [0.1, 0.85, 0.05]$	0.7470	0.7479
		$\theta = [0.4, 3.4, 0.2]$, $\lambda = [0.1, 0.85, 0.05]$	0.7470	**0.7480**
	(change λ)	$\theta = [0.1, 0.85, 0.05]$, $\lambda = [0.2, 1.7, 0.1]$	0.7468	0.7477
	(change both)	$\theta = \lambda = [0.05, 0.425, 0.025]$	0.7465	0.7473
		$\theta = \lambda = [0.2, 1.7, 0.1]$	0.7470	0.7479
		$\theta = \lambda = [0.4, 3.4, 0.2]$	0.7470	**0.7480**
		$\theta = \lambda = [0.8, 6.8, 0.4]$	**0.7471**	**0.7480**

Table 2: Inductive link prediction results on the crime network. The best results in the transductive and inductive settings are highlighted separately. Note that the ROC-AUC is not applicable for Radius Neighbour and linear approaches.

layers, a 0.6 dropout rate, and no weight decay.

5.2.4. Results Discussion

We compare the methods with respect to ROC-AUC and Average Precision (AP). However, simple baselines such as Radius Neighbours and Linear Models can only deliver Average Precision. The results of our experiments are reported in Table 2, while our findings are as follows:

- LASSO and Radius Neighbours achieve the worst results in correctly predicting links in the crime network. This was expected since both use

straightforward strategies in comparing node embeddings. However, more complex regression methods, such as Ridge Regression which gives more importance to the correlation between variables, notably achieve better results. Moreover, an SVM classifier with a linear kernel (SVC) can achieve comparable results to Ridge Regression. Graph2Gauss reaches the best results among the conventional baselines.

- Low-dimensional node embedding methods, including LINE, Node2Vec, and DeepWalk, do not show promising results on the crime network. LINE achieves the worst results among all three tested node embedding methods, unlike the experiments and comparisons in the original paper [68]. LINE is supposed to be more scalable than DeepWalk. Still, since it is an adjacency-based node representation, it does not perform as well as random walk-based embeddings in smaller networks like criminal analysis networks. Furthermore, since these node embedding methods are unsupervised, a simple SVC or Ridge classifier can reach comparable results.

- DEAL achieves considerably better results than conventional linear models and node embedding methods. This indicates that the DEAL framework effectively combines the structure and attributes of the crime network and extracts their augmented relations better than the methods that only use structure or attributes. Therefore, aligning the network topology with the node attributes is an effective strategy.

- Using a transductive validation set marginally improves the DEAL results. The larger the training set, the greater the performance improve-

ment.

- Finally, DEAL is not sensitive to its hyperparameters. Changing the parameters θ and λ does not have a noticeable impact on the performance of the DEAL link prediction framework in both its transductive and inductive tests.

6. Conclusion and Future Outlook

This paper studies the problem of co-offense prediction and crime linkage using real-life burglary data. We propose two network-based learning frameworks to automatically predict the links among offenders or crime cases. The first framework is a simple similarity-based unsupervised approach that does not require any labeled training instances, as it makes predictions solely based on the network topology. The second framework, DEAL, is a supervised approach that aligns the network topology with its node attributes to efficiently predict links, specifically for unseen newly emerged nodes. Since the unsupervised approach does not require training phases, it is better in scalability and has less need for memory requirements. However, it only works for transductive link prediction, which is suitable for predicting upcoming crimes based on the previous network structure. Alternatively, inductive link prediction is beneficial for finding connections between a new unknown case and existing ones. Experimental results indicate that the proposed frameworks are reliable and improve state of the art in criminology. However, there are several enhancements that can be made in the future:

- The proposed frameworks, especially the supervised DEAL framework, deal with an offline setting. As a future direction, we will adapt the

DEAL framework to the online setting, where the model is updated upon receipt of new instances.

- As mentioned in Section 3, we build on a previous study ([19]) and use the SIF embeddings for crime reports. However, one could train and fine-tune more recent embedding methods, like AlephBERT, specifically on police reports. This experiment will be continued in the future.

- In this paper, we only consider homogeneous graphs where the nodes are either offenders or crimes. However, the original graph is bipartite, with nodes representing both offenders and crimes. Therefore, the bipartite graph could be converted into a hypergraph, where the nodes are offenders, and a hyperedge indicates a crime and connects it to all associated offenders. In this way, both high-order semantics and complex relations between nodes can be captured, thus making transductive link prediction a hyperedge prediction problem. There is rich literature on heterogeneous graph embedding [71, 72, 73] and even on bipartite embedding as a special case [74]. Moreover, heterogeneous multi-level frameworks have recently attempted to learn the importance of nodes through hyperedge attention mechanisms [75, 76]. An interesting future direction is to extend the DEAL framework for hyperedge prediction [77] and compare it with heterogeneous graph embedding methods and the homogeneous version.

Acknowledgment

This work was supported by the European Union's Horizon 2020 research and innovation program under grant agreement No. 833635 (project ROXANNE: Real-time network, text, and speaker analytics for combating organized crime, 2019-2022).

References

[1] D. Weisburd, The law of crime concentration and the criminology of place, Criminology 53 (2) (2015) 133–157.

[2] N. N. Martinez, Y. Lee, J. E. Eck, S. O, Ravenous wolves revisited: a systematic review of offending concentration, Crime Science 6 (1) (2017) 10.

[3] Y. Lee, J. E. Eck, S. O, N. N. Martinez, How concentrated is crime at places? A systematic review from 1970 to 2015, Crime Science 6 (1) (2017) 6.

[4] M. A. Tayebi, U. Glässer, Social Network Analysis in Predictive Policing, 2016.

[5] A. G. Ferguson, Policing Predictive Policing, Vol. 94, Washington University Law Review, 2017.

[6] K. J. Bowers, S. D. Johnson, Who commits near repeats? a test of the boost explanation, Western Criminology Review 5 (3) (2004).

[7] S. D. Johnson, W. Bernasco, K. J. Bowers, H. Elffers, J. Ratcliffe, G. Rengert, M. Townsley, Space–time patterns of risk: A cross national assessment of residential burglary victimization, Journal of Quantitative Criminology 23 (3) (2007) 201–219.

[8] O. Kounadi, A. Ristea, A. Araujo, M. Leitner, A systematic review on spatial crime forecasting, Crime Science 9 (1) (2020) 7.

[9] L. Breiman, Random forests, Machine Learning 45 (1) (2001) 5–32.

[10] H. Zhang, The optimality of naive bayes, in: Proceedings of the 17th International FLAIRS Conference (FLAIRS), 2004, pp. 562–567.

[11] C. Cortes, V. Vapnik, Support-vector networks, Machine Learning 20 (3) (1995) 273–297.

[12] C. D. Uchida, Predictive Policing, in: G. Bruinsma, D. Weisburd (Eds.), Encyclopedia of Criminology and Criminal Justice, Springer, 2014, pp. 3871–3880.

[13] A. Meijer, M. Wessels, Predictive Policing: Review of Benefits and Drawbacks, International Journal of Public Administration 42 (12) (2019) 1031–1039.

[14] J. H. Ratcliffe, Advocate: Predictive Policing, in: D. Weisburd, A. A. Braga (Eds.), Police Innovation: Contrasting Perspectives, 2nd Edition, Cambridge University Press, Cambridge, 2019, pp. 347–365.

[15] R. Boba Santos, Critic: Predictive Policing: Where's the Evidence?, in: D. Weisburd, A. A. Braga (Eds.), Police Innovation: Contrasting

Perspectives, 2nd Edition, Cambridge University Press, 2019, pp. 366–398.

[16] Y.-S. Li, M.-L. Qi, An approach for understanding offender modus operandi to detect serial robbery crimes, Journal of Computational Science 36 (2019) 101024.

[17] K. Davies, J. Woodhams, The practice of crime linkage: A review of the literature, Journal of Investigative Psychology and Offender Profiling 16 (3) (2019) 169–200.

[18] J. Woodhams, C. R. Hollin, R. Bull, The psychology of linking crimes: A review of the evidence, Legal and Criminological Psychology 12 (2) (2007) 233–249.

[19] A. Solomon, A. Magen, S. Hanouna, M. Kertis, B. Shapira, L. Rokach, Crime linkage based on textual hebrew police reports utilizing behavioral patterns, in: Proceedings of the 29th ACM International Conference on Information and Knowledge Management (CIKM), 2020, p. 2749–2756.

[20] P. Stalidis, T. Semertzidis, P. Daras, Examining deep learning architectures for crime classification and prediction, Forecasting 3 (4) (2021) 741–762.

[21] R. N. Lichtenwalter, J. T. Lussier, N. V. Chawla, New perspectives and methods in link prediction, in: Proceedings of the 16th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2010, pp. 243–252.

[22] M. A. Hasan, V. Chaoji, S. Salem, M. Zaki, Link prediction using supervised learning, in: Proceedings of the SIAM Data Mining Workshop on Link Analysis, Counterterrorism and Security, 2006.

[23] M. A. Tayebi, M. Ester, U. Glässer, P. L. Brantingham, Spatially embedded co-offence prediction using supervised learning, in: Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2014, p. 1789–1798.

[24] G. M. Campedelli, Machine Learning for Criminology and Crime Research: At the Crossroads, Routledge, London, 2022.

[25] J. Woodhams, K. Toye, An empirical test of the assumptions of case linkage and offender profiling with serial commercial robberies., Psychology, Public Policy, and Law 13 (1) (2007) 59.

[26] C. Bennell, B. Snook, S. Macdonald, J. C. House, P. J. Taylor, Computerized crime linkage systems: A critical review and research agenda, Criminal Justice and Behavior 39 (5) (2012) 620–634.

[27] Y. Li, X. Shao, Thresholds learning of three-way decisions in pairwise crime linkage, Applied Soft Computing 120 (2022) 108638.

[28] Y.-S. Li, H. Chi, X.-Y. Shao, M.-L. Qi, B.-G. Xu, A novel random forest approach for imbalance problem in crime linkage, Knowledge-Based Systems 195 (2020) 105738.

[29] H. Chi, Z. Lin, H. Jin, B. Xu, M. Qi, A decision support system for detecting serial crimes, Knowledge-Based Systems 123 (2017) 88–101.

[30] J. Vimala Devi, K. Kavitha, Adaptive deep q learning network with reinforcement learning for crime prediction, Evolutionary Intelligence (2022) 1–12.

[31] K. He, X. Zhang, S. Ren, J. Sun, Deep residual learning for image recognition, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2016, pp. 770–778.

[32] B. Wang, D. Zhang, D. Zhang, P. Brantingham, A. L. Bertozzi, Deep learning for real time crime forecasting, IEICE Proceeding Series 29 (A3L-E-2-4) (2017) 330–333.

[33] Q. Wang, G. Jin, X. Zhao, Y. Feng, J. Huang, Csan: A neural network benchmark model for crime forecasting in spatio-temporal scale, Knowledge-Based Systems 189 (2020) 105120.

[34] K. Simonyan, A. Zisserman, Two-stream convolutional networks for action recognition in videos, Advances in Neural Information Processing Systems 27 (2014).

[35] Z. Wu, X. Wang, Y.-G. Jiang, H. Ye, X. Xue, Modeling spatial-temporal clues in a hybrid deep learning framework for video classification, in: Proceedings of the 23rd ACM International Conference on Multimedia, 2015, pp. 461–470.

[36] A. Ghazvini, S. N. H. S. Abdullah, M. K. Hasan, D. Z. A. B. Kasim, Crime spatiotemporal prediction with fused objective function in time delay neural network, IEEE Access 8 (2020) 115167–115183.

[37] A. Bojchevski, S. Günnemann, Deep gaussian embedding of graphs: Unsupervised inductive learning via ranking, in: Proceedings of the International Conference on Learning Representations (ICLR), 2018.

[38] A. Grover, J. Leskovec, node2vec: Scalable feature learning for networks, in: Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2016, pp. 855–864.

[39] S. Jha, E. Yang, A. O. Almagrabi, A. K. Bashir, G. P. Joshi, Comparative analysis of time series model and machine testing systems for crime forecasting, Neural Computing and Applications 33 (17) (2021) 10621–10636.

[40] T. Nakaya, K. Yano, Visualising crime clusters in a space-time cube: An exploratory data-analysis approach using space-time kernel density estimation and scan statistics, Transactions in GIS 14 (3) (2010) 223–239.

[41] Y.-L. Lin, M.-F. Yen, L.-C. Yu, Grid-based crime prediction using geographical features, ISPRS International Journal of Geo-Information 7 (8) (2018) 298.

[42] X. Zhang, L. Liu, L. Xiao, J. Ji, Comparison of machine learning algorithms for predicting crime hotspots, IEEE Access 8 (2020) 181302–181310.

[43] A. M. Olligschlaeger, Artificial neural networks and crime mapping, Crime Mapping and Crime Prevention 1 (1997) 313.

[44] W. Gorr, A. Olligschlaeger, Y. Thompson, Short-term forecasting of crime, International Journal of Forecasting 19 (4) (2003) 579–594.

[45] M. Feng, J. Zheng, J. Ren, A. Hussain, X. Li, Y. Xi, Q. Liu, Big data analytics and mining for effective visualization and trends forecasting of crime data, IEEE Access 7 (2019) 106111–106123.

[46] K. Kianmehr, R. Alhajj, Effectiveness of support vector machine for crime hot-spots prediction, Applied Artificial Intelligence 22 (5) (2008) 433–458.

[47] C.-H. Yu, M. W. Ward, M. Morabito, W. Ding, Crime forecasting using data mining techniques, in: Proceedings of the IEEE 11th International Conference on Data Mining Workshops (ICDMW), 2011, pp. 779–786.

[48] E. Eftelioglu, S. Shekhar, J. M. Kang, C. C. Farah, Ring-shaped hotspot detection, IEEE Transactions on Knowledge and Data Engineering 28 (12) (2016) 3367–3381.

[49] G. O. Mohler, M. B. Short, P. J. Brantingham, F. P. Schoenberg, G. E. Tita, Self-exciting point process modeling of crime, Journal of the American Statistical Association 106 (493) (2011) 100–108.

[50] J. H. Ratcliffe, A temporal constraint theory to explain opportunity-based spatial offending patterns, Journal of Research in Crime and Delinquency 43 (3) (2006) 261–291.

[51] M. Rosenblatt, Remarks on some nonparametric estimates of a density function, The Annals of Mathematical Statistics (1956) 832–837.

[52] B. W. Silverman, Density estimation for statistics and data analysis, Routledge, 2018.

[53] J. L. Toole, N. Eagle, J. B. Plotkin, Spatiotemporal correlations in criminal offense records, ACM Transactions on Intelligent Systems and Technology 2 (4) (2011) 1–18.

[54] S. Arora, Y. Liang, T. Ma, A simple but tough-to-beat baseline for sentence embeddings, in: Proceedings of the International Conference on Learning Representations (ICLR), 2017.

[55] J. Devlin, M.-W. Chang, K. Lee, K. Toutanova, Bert: Pre-training of deep bidirectional transformers for language understanding, in: Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers), Association for Computational Linguistics, 2019, pp. 4171–4186.

[56] T. Mikolov, K. Chen, G. Corrado, J. Dean, Efficient estimation of word representations in vector space, arXiv preprint arXiv:1301.3781 (2013).

[57] D. Liben-Nowell, J. Kleinberg, The link-prediction problem for social networks, Journal of the American Society for Information Science and Technology 58 (7) (2007) 1019–1031.

[58] L. A. Adamic, E. Adar, Friends and neighbors on the web, Social Networks 25 (3) (2003) 211–230.

[59] A.-L. Barabâsi, H. Jeong, Z. Néda, E. Ravasz, A. Schubert, T. Vicsek,

Evolution of the social network of scientific collaborations, Physica A: Statistical Mechanics and Its Applications 311 (3-4) (2002) 590–614.

[60] L. Lü, T. Zhou, Link prediction in complex networks: A survey, Physica A: Statistical Mechanics and Its Applications 390 (6) (2011) 1150–1170.

[61] S. Soundarajan, J. Hopcroft, Using community information to improve the precision of link prediction methods, in: Proceedings of the 21st International Conference on World Wide Web (WWW), 2012, pp. 607–608.

[62] Y. Hao, X. Cao, Y. Fang, X. Xie, S. Wang, Inductive link prediction for nodes having only attribute information, in: Proceedings of the 29th International Joint Conferences on Artificial Intelligence (IJCAI), 2021, pp. 1209–1215.

[63] M. Welling, T. N. Kipf, Semi-supervised classification with graph convolutional networks, in: Proceedings of the International Conference on Learning Representations (ICLR), 2017.

[64] P. Velickovic, G. Cucurull, A. Casanova, A. Romero, P. Lio, Y. Bengio, Graph attention networks, in: Proceedings of the International Conference on Learning Representations (ICLR), 2018.

[65] R. Hadsell, S. Chopra, Y. LeCun, Dimensionality reduction by learning an invariant mapping, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Vol. 2, 2006, pp. 1735–1742.

[66] B. Perozzi, R. Al-Rfou, S. Skiena, Deepwalk: Online learning of social representations, in: Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD '14, 2014, pp. 701–710.

[67] A. Grover, J. Leskovec, Node2vec: Scalable feature learning for networks, in: Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2016, p. 855–864.

[68] J. Tang, M. Qu, M. Wang, M. Zhang, J. Yan, Q. Mei, LINE: Large-scale information network embedding, in: Proceedings of the 24th International Conference on World Wide Web (WWW), 2015, pp. 1067–1077.

[69] C. K. Williams, C. E. Rasmussen, Gaussian processes for machine learning, Vol. 2, MIT press Cambridge, MA, 2006.

[70] A. Zhou, J. Yang, Y. Gao, T. Qiao, Y. Qi, X. Wang, Y. Chen, P. Dai, W. Zhao, C. Hu, Brief industry paper: Optimizing memory efficiency of graph neural networks on edge computing platforms, in: 2021 IEEE 27th Real-Time and Embedded Technology and Applications Symposium (RTAS), IEEE, 2021, pp. 445–448.

[71] Y. Dong, N. V. Chawla, A. Swami, metapath2vec: Scalable representation learning for heterogeneous networks, in: Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2017, pp. 135–144.

[72] T.-y. Fu, W.-C. Lee, Z. Lei, Hin2vec: Explore meta-paths in heterogeneous information networks for representation learning, in: Proceedings

of the ACM on Conference on Information and Knowledge Management (CIKM), 2017, pp. 1797–1806.

[73] R. Aponte, R. A. Rossi, S. Guo, J. Hoffswell, N. Lipka, C. Xiao, G. Chan, E. Koh, N. Ahmed, A hypergraph neural network framework for learning hyperedge-dependent node embeddings, arXiv preprint arXiv:2212.14077 (2022).

[74] W. Huang, Y. Li, Y. Fang, J. Fan, H. Yang, Biane: Bipartite attributed network embedding, in: Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval, 2020, pp. 149–158.

[75] H. Fan, F. Zhang, Y. Wei, Z. Li, C. Zou, Y. Gao, Q. Dai, Heterogeneous hypergraph variational autoencoder for link prediction, IEEE Transactions on Pattern Analysis and Machine Intelligence 44 (8) (2021) 4125–4138.

[76] H. Chen, H. Yin, X. Sun, T. Chen, B. Gabrys, K. Musial, Multi-level graph convolutional networks for cross-platform anchor link prediction, in: Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, 2020, pp. 1503–1511.

[77] C. Chen, Y.-Y. Liu, A survey on hyperlink prediction, arXiv preprint arXiv:2207.02911 (2022).

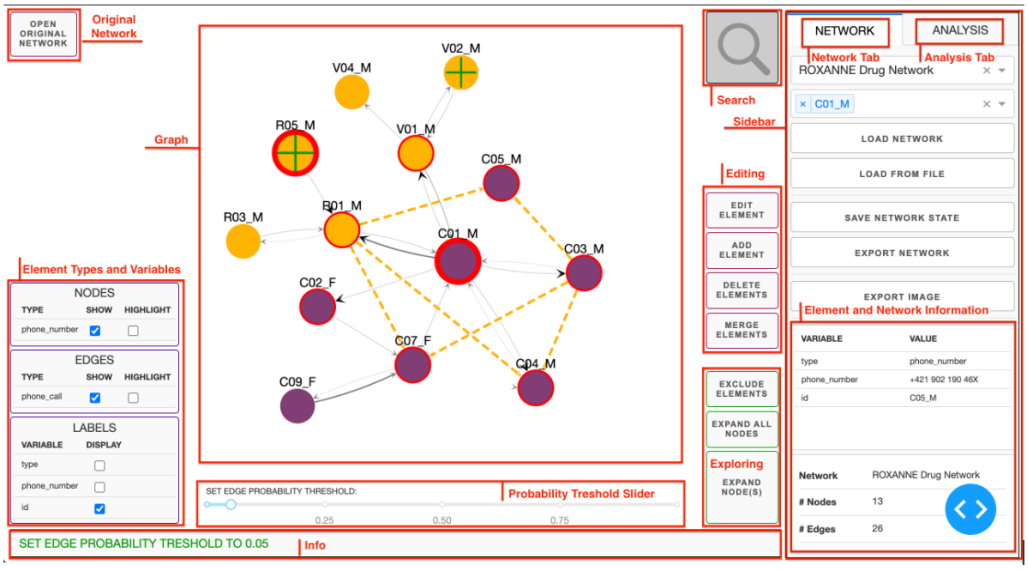


Figure 5: The visualizer user interface and its components. The sidebar enables the user to load, save and export a network. The main frame illustrates the network and provides editing and exploring of it.

7. Appendix

7.1. Network Visualization Platform

We developed a public visualization platform with a user interface for link prediction[8] as illustrated in Figure 5. The platform consists of three main components:

1. The **analyzer** provides the necessary analysis functionalities for the requests received from the *web interfaces & visualizer*. It queries the corresponding data from the *database manager* or external graph data, performs the requested analysis, and returns the analysis results.

[8] `https://anonymous.4open.science/r/criminal-network-visualization-2346`

2. The **web interfaces & visualizer**, which is a web-based component mainly for end-users to import, modify, perform analysis, and visually examine networks. This component receives users' requests from the *database manager* or external graph data and the corresponding analysis from the *analyzer* and visualizes the results on web interfaces.
   Our platform enables users to export the network into a file or an image via the sidebar menu options for later use. The output file is a JSON graph file with the attributes in nodes and links. The network image can be exported in any of the three SVG, PNG, and JPEG formats.
3. The in-memory **database manager** is designed to work smoothly and transparently with other components to load network data. For larger and more complex graphs, we can migrate to any graph database, such as JanusGraph or GraphQL, suitable for querying massive graph structures with only minor changes.
   Besides the built-in networks in *database manager*, our system supports loading external networks using a standard JSON format. The required fields of the nodes for the JSON graph file are "id" and "type", while the required fields of the links are "source", "target", and "type".

Our platform provides a drop-down menu to configure the parameters for network analysis functionalities such as link prediction[9]. Our interface contains different parts:

- The **sidebar** contains options related to loading, saving, and export-

[9]Our visualization platform offers other network analysis functionalities, including community detection and social influence analysis

ing a network in the *"Network"* tab, the network analysis tools in the *"Analysis"* tab, and general information about nodes and edges in *"Element and Network Information"*.

- The **main frame** primarily includes the visualization of the network itself and the means to edit and explore it. Each node is initially displayed in the network interface with a label attached to it. If the underlying dataset contains a label or name field, the content of that field is used. Otherwise, it uses the "id" field. The appearance of nodes is determined by their type property: different types of nodes are visualized using different shapes and colors. Expandable nodes are marked with a green plus sign on top of itself. A node is expandable if it has outgoing edges to nodes not yet displayed by the visualizer. If there is a probability or weight property attached to an edge, the thickness of an edge illustrates the value of that property. Nodes can be moved around by clicking and dragging them to the desired position. It is also possible to drag the whole window. Furthermore, nodes and edges can be clicked, which changes their visual appearance and allows further interaction with the selected element(s). Clicking nodes also highlights neighboring nodes and outgoing edges. If the network contains edge probabilities, a probability threshold slider is presented below the network visualization and can be used to hide edges below a certain threshold. Moreover, the controls at the bottom left allow hiding and highlighting node and edge types and selecting the variables to be displayed above the nodes in the visualization. Element types are defined in the dataset with the "type" field. All node properties in the

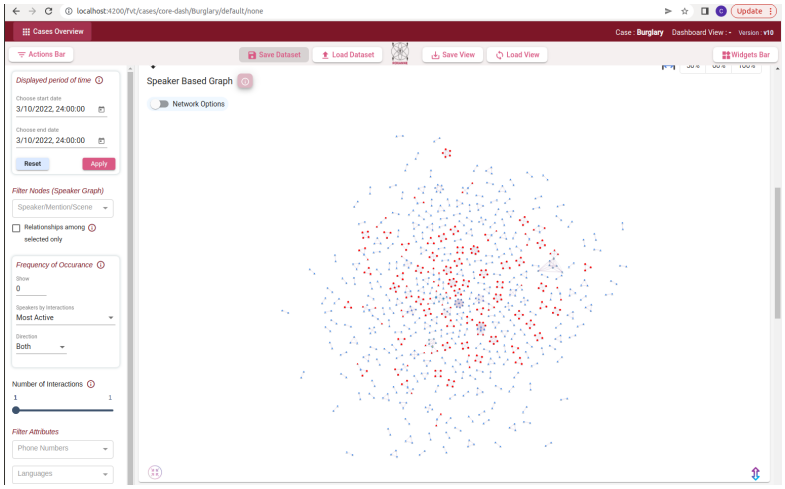
dataset can be displayed as labels using the three boxes.

- The **Editing** buttons, including adding, deleting, and editing a single element (node or edge) or merging two or more elements of the same type, permanently alter the network and are meant for upgrading the original network. However, the **Exploring** buttons, which expand nodes or exclude elements, only affect the current visualization and not the original loaded network.

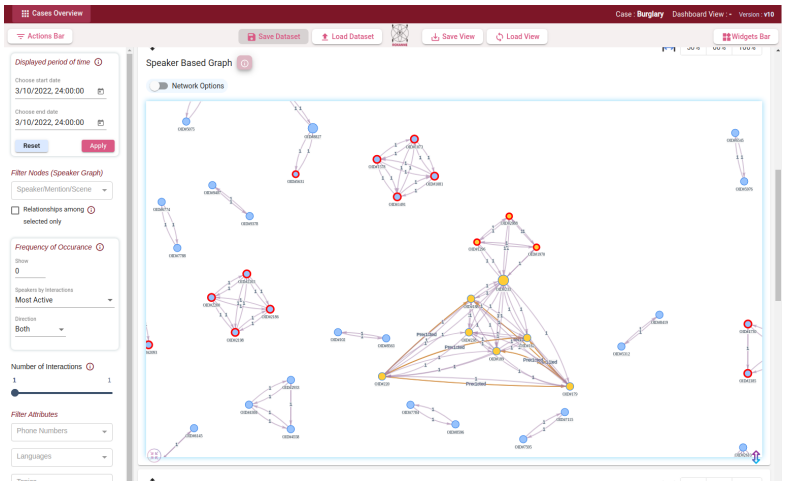
- We also provide the option of loading the original network beside the main modified network via the *OPEN ORIGINAL NETWORK* button, which splits the screen into two. This means that all changes done to the network in the main windows are not adopted in the original network. The original network can be dragged around, and nodes can be repositioned, but no other changes can be made to this network. The *"CLOSE ORIGINAL NETWORK"* button closes the original network view and brings the modified network back to full size.

Besides this network analysis platform, we are developing a powerful crime investigation tool as part of the ROXANNE EU project[10] called Autocrime. The Autocrime platform is not limited to the network analysis functionalities and includes a set of tools for researchers and law enforcement agencies to process intercepted conversations in ongoing investigations. It expects audio files and creates a JSON file which is then fed to the network analysis technologies. Figure 6 shows a snapshot of Autocrime running transductive link prediction on the same nodes as in Figure 4.

[10] `https://www.roxanne-euproject.org/`



(a)



(b)

Figure 6: (a) Visualization of a subnetwork of offender network in the Autocrime platform, (b) Transductive link prediction results on $OID\#179$ and $OID\#220$. The predicted links are shown in yellow arrows.