Graph Neural Networks in PyTorch for Link Prediction in Industry 4.0 Process Graphs

Eleanna Kafeza[1], Georgios Drakpopoulos[2](✉), and Phivos Mylonas[3]

[1] College of Interdisciplinary Research, Zayed University, Dubai, UAE
eleana.kafeza@zu.ac.ae

[2] Department of Informatics, Ionian University, Hellas, Greece
c16drak@ionio.gr

[3] Informatics and Computer Engineering Department, University of West Attica, Hellas, Greece
mylonasf@uniwa.gr

Abstract. Process mining constitutes an integral part of enterprise infrastructure as its adaptability and evolution potential enhance the digital awareness of stakeholders. In the context of Industry 4.0 a mainstay of process mining is the integrity verification of process graphs. Since manufacturing typically consists of numerous operations, it follows that process mining techniques, including link prediction, must possess learning capabilities powerful enough to accurately evaluate the deviation degree from the respective template using a wide array of structural and functional attributes, including semantics in the form of labels denoting operations such as data request or human operator notification. In turn, this relies heavily on discerning higher order patterns because of the distributed nature of industrial processes. Graph neural networks (GNNs) are ideally suited for performing link prediction since they offer scalability, versatility, and geometric intuition. Two attribute sets were tested, one containing only structural patterns and one combining them with functional ones. Results with synthetic benchmark process graphs of varying complexity show that GNNs exploit the extra functional information in the form of labels to recover missing edges, themselves part of the graph structure, even when the functional attributes are noisy.

Keywords: Graph neural networks · Link prediction · Industry 4.0 · Geometric analytics · Process graphs · Higher order patterns · PyTorch

1 Introduction

The current theory and practice of Industry 4.0 relies heavily on process mining for production quality, manufacturing progress, and smooth supply chain and logistics flow under a number of possibly conflicting up to a degree constraints. Industry 4.0 is a major turning point since historically production technology was limited to its manufacturing core, whereas tasks such as quality assurance, job flow monitoring, and procedure compliance were the exclusive domain of human professionals. However, the ongoing digital transformation of enterprise environments in conjunction with the recent advent of data driven methodologies require a higher degree of digital awareness, both from humans and machines. One major aspect of the latter is the considerable complexity frequently arising in human-to-machine and machine-to-machine interaction.

Table 1. Notation synopsis.

Symbol	Meaning	First in
$\triangleq$	Equality by definition	Eq. (1)
$\tau\left(\cdot,\cdot\right)$	Tanimoto set similarity coefficient	Eq. (5)
$\nu\left(\cdot,\cdot\right)$	Asymmetric Tversky set similarity index	Eq. (15)
$S_1 \setminus S_2$	Asymmetric set difference	Eq. (15)
$\{s_1,\ldots,s_n\}$	Set with elements $s_1,\ldots,s_n$	Eq. (2)
$(t_1,\ldots,t_n)$	Tuple with elements $t_1,\ldots,t_n$	Eq. (1)
$\lvert S\rvert$	Set or tuple cardinality functional	Eq. (2)
$u \sim v$	Vertex v is adjacent to u	Eq. (9)

In this context and given that process design flaws or execution errors not discovered and promptly corrected may well result in severe production degradation, machine learning (ML) strategies should be employed in order to perform graph integrity operations such as link prediction. This is especially critical with the increased requirements for system interoperability in general and the excessive complexity of process graph mining systems in particular. The major driving forces behind the latter are on one hand the plethora of data facilitating the discovery of latent links each complete with its own semantics and parameters [27], often in the form of labels, including system logs of heterogeneous formats across platforms, checkpoints, minor low level artificial intelligence (AI) decisions, data exchange requests, special flags or triggers, notification to human operators, local branch conditions, and subprocess spawn or termination, and on the other hand unforeseen conditions [1] such as driver misconfigurations, control software exceptions, power outages, random equipment failures, and spurious activations.

In such dynamic enterprise environments it makes perfect sense to utilize ML techniques which not only understand, so to speak, their complexity, but they also natively handle the distributed and higher order nature of process graphs. The latter implies that there should be a fine balance between computational cost and the information harnessed from the neighborhood of each vertex in the form of ground truth attribute vectors. Graph neural networks (GNNs) with attention mechanisms are tailored for tasks such as node classification and link prediction, while offering scalability and exploiting hardware parallelism.

The primary research objective of this conference paper is the development of a GNN implemented in PyTorch[1] for link prediction in synthetic Industry 4.0 benchmark process graphs with properties determined by the relevant scientific blibliography [20] [25]. The GNN is trained with either only structural attributes or in conjunction with functional ones [12]. Moreover, the effect of noisy functional attributes to link prediction is examined. The principal motivation, which also differentiates this work from many previous approaches, is to determine whether the addition of functional information results in improved recovery of purely structural attributes. The main underlying assumption is that, whereas edges may be removed from the template graphs, which express the ground truth, to simulate real process graphs, no edges can be added. Thus, the error is always one sided. This is not trivial as the lack of edges in a template process graph may well reflect physical or manufacturing constraints. The proposed methodology is versatile enough to be applied to any labeled graph from other domains.

The remaining of this work is structured as follows. In Sect. 2 the recent scientific literature about process mining and GNNs is briefly reviewed. The proposed methodology is presented in Sect. 3, while in Sect. 4 are given the results and the intuition behind them. Future research directions are explored in Sect. 5. Each technical acronym is explained the first time is encountered in text. The terms *attribute* and *feature* are used interchangeably. In function definitions parameters are placed inside parenthesis after formal arguments following a semicolon. Finally, in Table 1 the notation is explained.

2 Previous Work

Industry 4.0 is a milestone in manufacturing [6] as it aims to the full digitization of industrial production [20] through a wide array of sensors installed in machinery and in wearable electronics for human operators as well as through the delegation of minor, mundane, or dangerous tasks to AI [1]. Cyber-physical systems are critical for Industry 4.0 environments [18] with major aspects theoreof being operational efficiency [29], interoperability over a broad spectrum of operational requirements [25], and device technologies [4]. Industry 4.0 can benefit from circular economy [22] and reinforce industrial sustainability [19].

In process mining automatically generated process logs are mined for patterns [26], latent dependencies [27], and persistent anomalies [20]. The IEEE extensive event stream (XES) or IEEE standard 1849–2016 is a log file format designed for the explicit purpose of process mining [1] currently supported in a wide array of commercial software [17]. Algorithms for process pattern discovery include A and A^{+} [7] [14]. Research topics include dealing with malformed or otherwise imperfect process logs [23], extracting abstract events with unsupervised learning [24], optimizing software engineering [15], context-aware process mining with advanced graph mining [3], and the connection between process mining and sequence complexity [2]. Process mining relies on graph operations such as graph approximation [11] and heuristic community discovery [13].

[1] https://pytorch.org.

GNNs are widely employed among others in recommender systems [16], graph clustering [21], combinatorial optimization [5], and social network analysis [10]. Other neural network architectures include self organizing maps (SOMs) [8], probabilistc convolutional networks [28], and tensor stack networks (TSNs) [9].

3 Proposed Attributes

This section describes how the two attribute sets for the ground truth vector in each vertex are generated, shown in Table 2 along with the equation where each functional feature is defined in. Please note that the structural attributes are taken from the graph mining literature. Process graphs are distributed representations of higher order phenomena with vertices and edges storing values, indicating process results, conditional dependencies, and input determining subsequent manufacturing actions. Moreover, edge labels denoting states, conditions, or flags lead to complex global interaction patterns stemming from simpler local ones. In fact, each distinct vertex pair can be connected with multiple edges as long as their labels are distinct as shown in Fig. 1. Therefore, any extension of Metcalfe's law to the class of multilayer process graphs, whose formal definition is given in Definition 1, should take into account the above factors.

Definition 1 (Multilayer process graph). *A multilayer graph Q modelling an industrial process is formally defined as the ordered quadruple of equation (1):*

$$Q \triangleq (V, E, \Sigma, h) \tag{1}$$

- *The vertex set V contains states such as the beginning or the end of a subprocess or important intermediate operations.*
- *The directed edge set E contains the dependencies or the connections between either process states or subprocesses.*
- *The label set Σ consists of each possible label. The latter depend heavily on the semantics of the underlying business process.*
- *The functional $h\colon E \rightarrow 2^{\Sigma} \setminus \emptyset$ maps an edge to a not necessarily unique label subset as shown in equation (2).*

In these graphs a *layer* is formed by the edges of a single label and their endpoints with the total activity depending on a number of interacting factors. The first is the activity in each separate layer. Thus, the denser a layer is, the higher its activity. The second is the activity across layers occurring at the vertices connecting different layers. Therefore, the more vertices two layers are connected with, the higher their interaction is. The final factor is the labels in each edge with higher edge variability resulting in higher activity on the process graph.

In an Industry 4.0 setting the transition from one process step to the next one requires various elements, each represented with a single labeled edge resulting in vertex pairs connected with multiple edges. Instead of examining each such edge separately, they are collected in a single edge with a label set L as explained in definition 2. This lays the groundwork for higher order functional features.

Table 2. Ground truth attributes.

Structural attributes	Functional attributes	Definition
Inbound degree	Label variability of inbound edges	Eq. (3)
Outbound degree	Label variability of outbound edges	Eq. (3)
Input or output vertex?	Vertex fan out	Eq. (4)
Number of triangles	Similarity of directed labeled triangles	Eq. (5)
Clustering coefficient	Similarity of undirected labeled triangles	Eq. (5)
Number of squares	Similarity of directed labeled squares	Eq. (5)
Number of squares	Similarity of undirected labeled squares	Eq. (5)
Adamic-Adar score	Labeled Adamic-Adar score	Eq. (9)
Resource allocation	Labeled reource allocation	Eq. (10)
Shortest paths through the vertex	Label variability of paths through the vertex	Eq. (13)

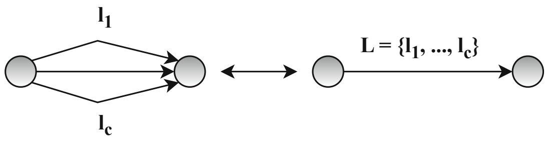

Fig. 1. Edge sets.

Definition 2 (Process graph edge). *For a given vertex pair connected by c edges with distinct labels let L be the set consisting of these edges as shown in (2). These individual edges can be replaced by a single edge with L as its property.*

$$L \triangleq \{l_1, \ldots, l_c\} \in 2^{\Sigma} \setminus \emptyset, \quad c \triangleq |L| \tag{2}$$

At this point it should be highlighted that throughout this work a *template graph* is the blueprint graph or the ground truth, which is considered to be flawless, and a *real* or *actual process graph* is the graph derived by removing edges from the template graph in order to simulate errors during manufacturing. Each attribute has been normalized in order to keep them at the same scale.

Based on definition 2 a number of features can be derived for each label set. The first is the inbound variability $\beta_{\text{in}}(v)$ of a vertex v which is defined as the cardinality of the inbound label set as shown in (3). The outbound variability $\beta_{\text{out}}(v)$ is similarly defined. Intuitively the two variability flavors correspond to the number of conditions attached to a vertex like reading sensor input, sending or waiting for a control signal, notifying a human operator, moving an item across the plant floor, or running a full cycle of diagnostics.

$$\beta_{\text{in}}(v) \triangleq \frac{1}{\max_{s \in Q}\left[\beta_{\text{in}}(s)\right]} |L_{\text{in}}| \quad \text{and} \quad \beta_{\text{out}}(v) \triangleq \frac{1}{\max_{s \in Q}\left[\beta_{\text{out}}(s)\right]} |L_{\text{out}}| \tag{3}$$

Along a similar line of reasoning the fan-out $\gamma(v)$ shown in equation (4) can be also defined on the grounds that it is a nonlinear transformation of the features of (3) expressing how the graph branches at v. This is not trivial to be computed by the GNN of Sect. 3 in contrast to, for instance, the sum of or the difference between these attributes. This attribute is zero only for input or output vertices.

$$\gamma(v) \triangleq \begin{cases} \frac{1}{\max_{s\in Q}\left[\gamma(s)\right]}\frac{\beta_{\text{out}}(v)}{\beta_{\text{in}}(v)}, & \beta_{\text{in}}(v) \neq 0 \\ 0, & \text{otherwise} \end{cases} \tag{4}$$

Triangles, namely closed paths of length three, play a significant role in both the overall connectivity and the small world phenomenon in power law graphs. When label sets are available, then there is a plethora of ways to measure labeled triangle similarity. The one shown in (5) relies on the Tanimoto set similarity coefficient of (6) for the label sets L_u, L_v, and L_s of the vertices u, v, and s. Since its algebraic expression ignores edge direction, it can be applied to both directed and undirected triangles. Formula (5) can be naturally extended to squares with a similar rationale. Moreover, the harmonic mean $\mathcal{H}(\cdot)$ is insensitive to outliers, in contrast to the arithmetic mean, or to values close to zero, unlike the geometric mean. Therefore, it has better numerical and algorithmic properties compared to the other two Pythagorean means. Its numerical stability can be explained by (7) and (8). Additionally, its intuitive interpretation as work rate or velocity can be used to assess trajectories in feature spaces across domains like physics and computer science, paving thus the way for geometric analytics.

$$\delta(v) \triangleq \frac{1}{\max_{u'\in Q}\left[\delta(u')\right]}\frac{3}{\frac{1}{\tau\left(L_u,L_v\right)}+\frac{1}{\tau\left(L_v,L_s\right)}+\frac{1}{\tau\left(L_s,L_u\right)}} \tag{5}$$

The Tanimoto set similarity coefficient of (6) relies on rudimentary set operations and it is a metric of the size of the overlap of two sets. The right hand size can be derived by the left hand one through Venn diagrams and it is computationally more efficient, especially if set cardinality estimators are used.

$$\tau\left(S,S'\right) \triangleq \frac{|S\cap S'|}{|S\cup S'|} = \frac{|S\cap S'|}{|S|+|S'|-|S\cap S'|} \tag{6}$$

The partial derivative of the harmonic mean with respect to x_k of the argument vector is computed as in (7) using the rule of inverse function derivative. Therein it can be seen that this derivative is roughly proportional to the square of the harmonic mean value at this point. Given that in the attributes of Table 2 lie between zero and one, it follows that the partial derivative remains bounded.

$$\frac{\partial\mathcal{H}\left(\mathbf{x}\right)}{\partial x_k} = \frac{\partial}{\partial x_k}\left[\frac{n}{\sum_{k=1}^{n}\frac{1}{x_k}}\right] = -n\frac{\frac{\partial}{\partial x_k}\left[\frac{1}{\sum_{k=1}^{n}\frac{1}{x_k}}\right]}{\left(\sum_{k=1}^{n}\frac{1}{x_k}\right)^2} = \frac{x_k}{n}\mathcal{H}\left(\mathbf{x}\right)^2 \tag{7}$$

The harmonic mean gradient vector with respect to the input vector is given by (8). Again, given that the actucal harmonic mean value lies between zero and one, it follows that the gradient vector norm not only is easy to compute based on known quantities but it also remains bounded in the region of interest.

$$\left\|\nabla_{\mathbf{x}}\mathcal{H}\left(\mathbf{x}\right)\right\|_2 = \frac{\mathcal{H}\left(\mathbf{x}\right)^2}{n}\left\|\underbrace{\left[x_1\ \ldots\ x_n\right]^T}_{\mathbf{x}}\right\|_2 = \frac{\mathcal{H}\left(\mathbf{x}\right)^2}{n}\left\|\mathbf{x}\right\|_2 \tag{8}$$

The labeled Adamic-Adar score shown in (9) is a variant of the Adamic-Adar score, which in turn is roughly based on information theory as it indicates the amount of information between a vertex v and its neighobrs and it is extensively employed in link prediction settings. The variant proposed here is a metric of the label variability between a given vertex v and every adjacent vertex u.

$$\zeta(v) \triangleq \frac{1}{\max_{s\in Q}\left[\zeta(s)\right]}\sum_{u\sim v}\frac{1}{\ln\left|L_u\right|} \tag{9}$$

The labeled resource allocation of (10) has a similar formula with (9) but it comes from a different reasoning, namely the expansion potential of the graph segment formed by v and its neighborhood. This can be adapted to evaluate this potential in terms of the local information content existing around v.

$$\eta(v) \triangleq \frac{1}{\max_{s\in Q}\left[\eta(s)\right]}\sum_{u\sim v}\frac{1}{\left|L_u\right|} \tag{10}$$

Another graph feature which has a geometric aspect is the label variability across a given path π of length p coming through vertex s which also expresses topological proximity. Specifically, the overall path similarity is the harmonic mean of the similarities of the individual edges comprising π as shown in (11).

$$\xi(\pi; s) \triangleq \frac{p}{\sum_{u\sim v}\frac{1}{\tau\left(L_u, L_v\right)}}, \qquad \pi = \{(u,v)\}, s \in \pi \tag{11}$$

Under mild conditions $\xi(\pi)$ can be approximated by (12). The correction factor c_0 and integral bounds τ_1 and τ_2 measure the approximation error. It should be noted that the following conditions should hold: The path length p should be sufficiently large and there is sufficiently high variability in the values of label set similarity. The latter can be assessed by set cardinality estimators.

$$\xi(\pi) \approx p\left(\int_{\tau_1}^{\tau_2}\frac{d\tau}{\tau}\right)^{-1} = p\left(\ln\frac{\tau_2}{\tau_1} + c_0\right)^{-1} \tag{12}$$

The final feature is the arithmetic mean of the similarities as determined by (11) of the shortest paths coming through vertex v. This evaluates the role

v plays in the overall structural coherence of the graph Q, in contrast to the preceding attributes which measure contributions to local structural coherence.

$$\sigma(v) \triangleq \frac{1}{\max_{s \in Q}\left[\sigma(s)\right]} \frac{1}{|\{\pi\}|} \sum_{\pi} \xi(\pi; v) \tag{13}$$

It should be noted that (13) was derived by analogy with the Newman-Girvan vertex centrality metric. Intuitively, it is maximized for articulations or when the graph has a tree-like structure and minimized for peripheral vertices. Since the harmonic mean of (11) has already smoothed out outliers in the various shortest paths through v, the simpler arithmetic mean is used in (13).

4 Results

4.1 Data Synopsis

Table 3 contains the summaries of the benchmark graphs, each a Kronecker synthetic graph, taken from [12]. In order to generate the simulated process graphs, edges were removed at random from the template graph until the average SNR of (16) is met. For each benchmark and for each SNR n_p noisy graph instances were created by removing a percentage of edges m_p and their labels, where n_p and m_p are shown in Table 4 along with other parameters.

Table 3. Dataset synopsis.

Property	Set 1	Set 2	Set 3	Set 4	Property	Set 1	Set 2	Set 3	Set 4
Generator vertices	5	5	7	7	Diameter	11	13	15	17
Generator edges	7	8	13	17	80% diameter	7	9	11	12
Label set size	16	32	48	64	90% diameter	8	11	13	15
Number of triangles	625	3125	7617	21881	Labels per edge	6.53	11.67	28.44	32.33
Number of squares	422	1932	5894	18432	Label variance	4.19	8.32	14.86	16.22

From Table 3 it follows that the four benchmark graphs have an increasing level of structural and functional complexity. In particular, set 1 has few patterns and a small label set. Also there is a low number or triangles and a short diameter, indicating a very local information flow. Set 2 has a moderate number of patterns and label set size. Also it has a short diameter and an average number of triangles, meaning that there is some structural resiliency. Set 3 has a high number of patterns but a moderate label set size. Also it has a longer diameter and a high number of triangles. Therefore, it is more complex than the preceding benchmark graphs but it contains more attributes to be exploited. Finally, set 4 has a high number of triangles and a large label set size.

Table 4. Experiment setup.

Parameter	Value	Parameter	Value	Parameter	Value
n_p	100.000	Tversky index	0.95, 0.9, 0.85	Learning rate	cosine
m_p	0.05, 0.1, 0.15	Discrete SNR	2.9444, 2.1972, 1.7346	Epochs e_p	50

The GNN was implemented in PyTorch using the *geometry* library, which is a very popular tool for this ML task. Additionally, the PyTorch implementation of GNNs supports sparse attribute vectors, further reducing computational cost. The training takes place over e_p epochs using the cosine learning rate. The decay rate of the latter adapts to the number of iterations in the training process. Specifically, during the early iterations the cosine rate is close to one and then decays with a quadratic rate. This is due to the Taylor expansion shown in (14).

$$\cos\vartheta = \sum_{k=0}^{+\infty} (-1)^k \frac{\vartheta^{2k}}{2k} \approx 1 - \frac{\vartheta^2}{2} \tag{14}$$

In order to create to evaluate the effect noisy functional attributes have on link prediction, the distortion degree of the functional features should be evaluated. This can be done through the asymmetric Tversky index of (15). The parameters α_0 and α_1 determine different penalties for the different sets in the denominator. Here α_0 is three times as high as α_1 as missing labels are more important.

$$\nu\left(T, V; \alpha_0, \alpha_1\right) \triangleq \frac{|T \cap V|}{|T \cap V| + \alpha_0 |T \setminus V| + \alpha_1 |V \setminus T|}, \quad \alpha_0 + \alpha_1 = 1 \tag{15}$$

The above evaluates the divergence between sets T and V where the former is a template and the latter a variance thereof. Thus, these two sets are not interchangeable as is the case of many major set similarity metrics. The cardinality of large datasets can be estimated with suitable techniques. The following hold:

- When T and V coincide, their intersection equals their cardinality and both set differences are empty. Thus, the nominator equals the denominator.
- Similarly, when T and V are disjoint, then the nominator is zero. Simultaneously, at least one of the two set difference terms is non-zero.
- For intermediate cases at the denominator the intersection and at least one of the two set difference terms is non-empty and so is the nominator.

With 15 the discrete signal-to-noise ratio (SNR) can be defined as in 16. This definition is different from other settings, for instance in digital telecommunications where the nature of noise is continuous and it is attributed to factors including electron mobility from multiple sources of equal power. In this context noise comes from label omission such as malformed or partially overwritten logs.

$$\mathrm{SNR} \triangleq \ln\left(\frac{\nu}{1-\nu}\right) \Leftrightarrow \nu = \frac{1}{1 + \exp\left(-\mathrm{SNR}\right)}, \quad 0 < \nu < 1 \tag{16}$$

The logarithm in (16) is inspired by the logit function. Also ν is the average SNR taken over all edges of the noiseless and the noisy process graphs. Moreover, the logarithm tends to be easy to be approximated by a piecewise constant function. One way to see this is to consider the average tangent λ in non-overlapping exponential intervals as shown in (17). Note that (16) is an independent noise power assessment metric not used in the GNN training or in its evaluation.

$$\lambda \triangleq \left| \frac{\log_2 2^{\kappa+1} - \log_2 2^{\kappa}}{2^{\kappa+1} - 2^{\kappa}} \right| = 2^{-\kappa} \tag{17}$$

The computation of SNR is genrally numerically stable. However, the closer SNR gets to zero or one, the harder becomes to compute its actual value since the derivative approaches infinity with a quadratic rate as shown in (18):

$$\left| \frac{d\mathrm{SNR}}{d\nu} \right| = \left| \frac{d}{d\nu} \left[\ln \left(\frac{\nu}{1-\nu} \right) \right] \right| = \left| \frac{1}{\nu(1-\nu)} \right| = \frac{1}{\nu(1-\nu)} \tag{18}$$

Table 5. GNN accuracy for both attribute sets.

	SNR	noiseless	noiseless	2.9444	2.9444	2.1972	2.1972	1.7346	1.7346
graph	m_p	S	S+F	S	S+F	S	S+F	S	S+F
set1	0.05	0.9417	0.9483	0.9131	0.9205	0.8715	0.8804	0.8551	0.8633
	0.10	0.9124	0.9284	0.8986	0.9154	0.8631	0.8731	0.8433	0.8591
	0.15	0.8991	0.9108	0.8731	0.8926	0.8554	0.8592	0.8301	0.8441
set2	0.05	0.9433	0.9522	0.9164	0.9199	0.8831	0.8984	0.8589	0.8713
	0.10	0.9401	0.9467	0.8941	0.9036	0.8701	0.8792	0.8389	0.8544
	0.15	0.9208	0.9312	0.8817	0.8970	0.8665	0.8712	0.8345	0.6465
set3	0.05	0.9302	0.9554	0.9217	**0.9344**	0.8851	**0.9024**	0.8642	0.8693
	0.10	0.9286	0.9324	0.9055	0.9186	0.8711	0.8902	0.8559	0.8631
	0.15	0.9143	0.9224	0.8976	0.9067	0.8631	0.8793	0.8411	0.8493
set4	0.05	**0.9590**	**0.9618**	**0.9319**	0.9335	**0.8991**	0.9017	**0.8723**	**0.8887**
	0.10	0.9510	0.9543	0.9158	0.9199	0.8718	0.8841	0.8645	0.8742
	0.15	0.9467	0.9521	0.9044	0.9154	0.8604	0.8734	0.8567	0.8591

In Table 5 the accuracy of the GNN for the structural (S) and the structural and functional (S+F) feature vectors is shown for each synthetic benchmark graph for each value of m_p and discrete SNR. The maximum of each column is marked with boldface for easier comparison. Also in Fig. 2 the accuracy for the first and fourth benchmarks are shown for the lowest m_p value.

From the above results certain conclusions can be drawn. First and foremost, the introduction of functionality always outperforms the purely structural attributes in terms of accuracy. This can be attributed to the fact that functionality follows structure as labels belong to labels. Moreover, the increasing complexity does not necessarily lead to significant accuracy degradation. This can be explained by the distributed nature of GNNs which exploit attribute locality as well as the distributed information typically present on graphs. This is in accordance with the requirement that Industry 4.0 processes are resilient to many minor failures and reconfigurable enough to overcome them through alternative paths. Another observation is that the simplest benchmark graph goes not necessarily yield the best accuracy, as more complex graphs have more variability. Therefore, larger graphs are not merely a higher granularity representation of individual processes. Instead, they offer more flexibility by breaking down processes to elementary ones and paving the way for advanced pattern mining algorithms relying on tensor algebra and higher order statistics. The latter can yield potentially deeper insight to what constitutes normal operation or what causes systematic anomalies. Finally, the higher the discrete SNR is, the more difficult the task of link prediction becomes. This shows the importance of the role of the functional attributes to recovering edges.

4.2 Discussion

Graphs are indispensable tools for representing manufacturing processes for various reasons. First and foremost, properties tied to the underlying industrial process are stored in vertices and edges as appropriate. In this case and depending on the application, it may be significant that past states may be retained and accessed, implying that graph data structures supporting persistency, like splay trees, should be used. Alternatively, process properties can well be stored in specially marked vertices representing subprocesses or composite transitions between the various process states. Although representing transitions with vertices may sound counter-intuitive, it is in fact a valid graph design approach.

Along a similar line of reasoning discrete time steps can be represented by dedicated vertices. Thus using these vertices as anchors temporal queries pertaining to given interval such as when a subprocces was called or terminated can be executed. With this information available, a process execution distribution template can be assembled and tracked in real time as part of an operational profile and checked regularly against discrepancies. This can reveal not only random but also systemic errors. Still that comes at the expense of added complexity.

Moreover, when a certain event, task, or transition is tied to a certain probability, then the latter can be used among others to define the cost of traversing an edge or reaching to or leaving from a vertex of a fuzzy graph, where said cost can depend on the probability of an edge or vertex belonging to a graph. This can form the basis of various graph potential functions. For instance, if p_j is the probability that edge e_j belongs to the graph, then a candidate potential function $V_e(e_j)$ for edges is shown in (19). Also, based on the edges e_j incident to a particular vertex v_i a cost function $V_v(v_i)$ for vertices can also be defined.

$$V_e(e_j) \triangleq \frac{1}{1+p_j^2} \qquad \text{and} \qquad V_v(v_i) \triangleq \frac{1}{|\{e_j\}|} \sum_{e_j} V_e(e_j) \tag{19}$$

Finally, as shown here process graphs abound with patterns shaped by factors such as the graph growth rate, the possible types of interaction, and graph topol-

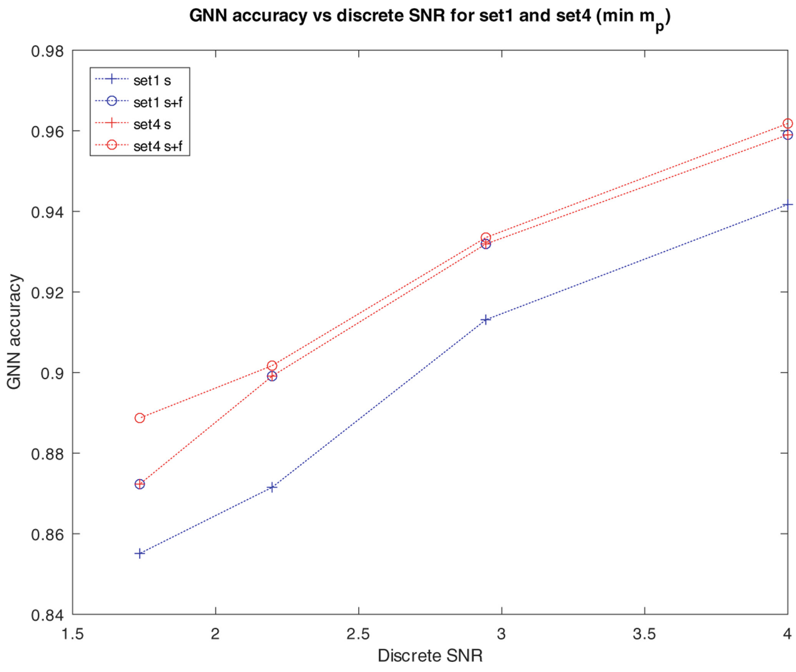

Fig. 2. GNN accuracy for the first and four benchmarks.

ogy. Among them higher order patterns are more difficult to discover, but at the same time they provide greater insight into the structure and the functionality of the graph. For instance, in the case of mixed labeled triangles, many results from classical graph theory like Sperner's lemma or maximal colored cliques can be applied. Because of the inherently distributed nature and self-similarity of graphs, most higher order patterns of interest may be found locally, despite having a global contribution, in the respective adjacency matrix Laplacian.

The importance of the proposed methodology stems from the fact that process graphs are included in nearly all manufacturing levels ranging from simple everyday administrative tasks, such as the timely dispatching maintenance personnel to defunct production units, to complex ones, such as the entire unified control of job flow to a factory floor in order to meet unexpected demand spikes. Industry 4.0 relies heavily on the inclusion of low level AI tools, such as intelligent agents, for automating mundane or repeating tasks, executing dangerous operations, and even automatically resolving most conflicts between two or more low level processes. Additionally, human operators are delegated to supervisory roles, intervening only when the local AI is unable to resolve an issue. To this end they are equipped with a broad array of sensors installed in smart watches, glasses, or cell phones and tablets with wearable electronics on the way, thus essentially becoming mobile sensor platforms constantly on the move on the factory floor. Graph resilience is believed to be a local property, a hypothesis which is yet to be proven for a number of domains but it is nonetheless supported by a substantial volume of empirical data and by intuition.

5 Conclusions and Future Work

This conference paper focuses on developing a graph neural network (GNN) for link prediction in Industry 4.0 graphs. They are trained in two sets of attributes, one containing only structural features and one combining them with functional ones, specifically in the form of labels tied to the underlying industrual process like events in system logs, intelligent agent reports, and automated telemetry. Results with synthetic benchmark graphs of varying complexity indicate that the additional functional information results in increased performance, even in cases of severe discrete noise. The latter as well as the signal-to-noise ratio (SNR) are intuitively defined with elementary set operations. Moreover, the proposed methodology is sufficiently generic to be applied to labeled graphs from other domains such as linguistics, graph signal processing, or social networks.

The results hint at a number of further research directions, therefore laying the groundwork for a utilizing GNNs for process clustering, anomaly discovery, and improved process scheduling based on attributes such as the average degree and the decay rate of the degree distribution. Another possible research direction is to train GNNs in order to generate useful diagnostic messages to human operators, especially to the floor crews. Moreover, when deviations are encountered, they should be explainable to an extent, so that human operators can take swift and appropriate action as needed. Finally, the above should be part of an integrated framework for the theory and practice and process mining where the system design principles can be easily understood in high level terms.

Acknowledgements. This conference paper has been funded by Research Initiative Fund (RIF) Grant R23015, Zayed University, UAE.

References

1. Acampora, G., Vitiello, A., Di Stefano, B., van der Aalst, W., Günther, C., Verbeek, E.: IEEE 1849: the XES standard. IEEE Comput. Intell. Mag. **12**(2), 4–8 (2017)
2. Augusto, A., Mendling, J., Vidgof, M., Wurm, B.: The connection between process complexity of event sequences and models discovered by process mining. Inf. Sci. **598**, 196–215 (2022)
3. Becker, T., Intoyoad, W.: Context aware process mining in logistics. Procedia Cirp **63**, 557–562 (2017)
4. Brettel, M., Friederichsen, N., Keller, M., Rosenberg, M.: How virtualization, decentralization and network building change the manufacturing landscape: an industry 4.0 perspective. Int. J. Inf. Commun. Eng. **8**(1), 37–44 (2014)
5. Cappart, Q., et al.: Combinatorial optimization and reasoning with graph neural networks. J. Mach. Learn. Res. **24**(130), 1–61 (2023)
6. Castelo-Branco, I., Cruz-Jesus, F., Oliveira, T.: Assessing industry 4.0 readiness in manufacturing: evidence for the EU. Comput. Ind. **107**, 22–32 (2019)

7. Choueiri, A.C., Santos, E.A.P.: Discovery of path-attribute dependency in manufacturing environments: a process mining approach. J. Manuf. Syst. **61**, 54–65 (2021)
8. Drakopoulos, G., Giannoukou, I., Sioutas, S., Mylonas, P.: Self organizing maps for cultural content delivery. Neural Comput. Appl. (2022). https://doi.org/10.1007/s00521-022-07376-1
9. Drakopoulos, G., Kafeza, E., Mylonas, P., Iliadis, L.: Transform-based graph topology similarity metrics. Neural Comput. Appl. **33**(23), 16363–16375 (2021). https://doi.org/10.1007/s00521-021-06235-9
10. Drakopoulos, G., Kafeza, E., Mylonas, P., Sioutas, S.: A graph neural network for fuzzy twitter graphs. In: Cong, G., Ramanath, M., (eds.) CIKM companion volume, vol. 3052. CEUR-WS.org (2021)
11. Drakopoulos, G., Kafeza, E., Mylonas, P., Sioutas, S.: Approximate high dimensional graph mining with matrix polar factorization: a twitter application. In: IEEE Big Data, pp. 4441–4449. IEEE (2021). https://doi.org/10.1109/BigData52589.2021.9671926
12. Drakopoulos, G., Kafeza, E., Mylonas, P., Sioutas, S.: Process mining analytics for industry 4.0 with graph signal processing. In: WEBIST SCITEPRESS, pp. 553–560 (2021). https://doi.org/10.5220/0010718300003058
13. Drakopoulos, G., Mylonas, P.: A genetic algorithm for Boolean semiring matrix factorization with applications to graph mining. In: Big Data. IEEE (2022). https://doi.org/10.1109/BigData55660.2022.10020828
14. Fang, N., Fang, X., Lu, K., Asare, E.: Online incremental mining based on trusted behavior interval. IEEE Access **9**, 158562–158573 (2021)
15. Fauzi, R., Andreswari, R.: Business process analysis of programmer job role in software development using process mining. Procedia Comput. Sci. **197**, 701–708 (2022)
16. Gao, C., et al.: A survey of graph neural networks for recommender systems: challenges, methods, and directions. ACM Trans. Recommender Syst. **1**(1), 1–51 (2023)
17. Kerin, M., Pham, D.T.: A review of emerging industry 4.0 technologies in remanufacturing. J. Cleaner Prod. **237**, 117805 (2019)
18. Lee, J., Bagheri, B., Kao, H.A.: A cyber-physical systems architecture for industry 4.0-based manufacturing systems. Manufact. lett. **3**, 18–23 (2015)
19. Machado, C.G., Winroth, M.P., Ribeiro da Silva, E.H.D.: Sustainable manufacturing in industry 4.0: an emerging research agenda. Int. J. Prod. Res. **58**(5), 1462–1484 (2020)
20. Mitsyuk, A.A., Shugurov, I.S., Kalenkova, A.A., van der Aalst, W.M.: Generating event logs for high-level process models. Simul. Model. Pract. Theory **74**, 1–16 (2017)
21. Müller, E.: Graph clustering with graph neural networks. J. Mach. Learn. Res. **24**, 1–21 (2023)
22. Rajput, S., Singh, S.P.: Connecting circular economy and industry 4.0. Int. J. Inf. Manage. **49**, 98–113 (2019)
23. Sadeghianasl, S., Ter Hofstede, A.H., Suriadi, S., Turkay, S.: Collaborative and interactive detection and repair of activity labels in process event logs. In: 2020 2nd International Conference on Process Mining, pp. 41–48. IEEE (2020)
24. Tax, N., Sidorova, N., Haakma, R., van der Aalst, W.M.P.: Event abstraction for process mining using supervised learning techniques. In: Bi, Y., Kapoor, S., Bhatia, R. (eds.) IntelliSys 2016. LNNS, vol. 15, pp. 251–269. Springer, Cham (2018). https://doi.org/10.1007/978-3-319-56994-9_18

25. Van Der Aalst, W.: Process mining. Commun. ACM **55**(8), 76–83 (2012)
26. Van Der Aalst, W.: Process mining: overview and opportunities. ACM Trans. Manage. Inf. Syst. **3**(2), 1–17 (2012)
27. Verenich, I., Dumas, M., Rosa, M.L., Maggi, F.M., Teinemaa, I.: Survey and cross-benchmark comparison of remaining time prediction methods in business process monitoring. ACM Trans. Intell. Syst. Technol. **10**(4), 1–34 (2019)
28. Yang, L., Guo, Y., Gu, J., Jin, D., Yang, B., Cao, X.: Probabilistic graph convolutional network via topology-constrained latent space model. IEEE Trans. Cybern. **52**(4), 2123–2136 (2020)
29. Zhou, K., Liu, T., Zhou, L.: Industry 4.0: towards future industrial opportunities and challenges. In: 2015 12th International Conference on Fuzzy Systems and Knowledge Discovery, pp. 2147–2152. IEEE (2015)