Multi-source Inductive Knowledge Graph Transfer

Junheng Hao[1]✉, Lu-An Tang[2], Yizhou Sun[1], Zhengzhang Chen[2], Haifeng Chen[2], Junghwan Rhee[3], Zhichuan Li[4], and Wei Wang[1]
{jhao, yzsun, weiwang}@cs.ucla.edu, {ltang, zchen, haifeng}@nec-labs.com, jrhee2@uco.edu, zhichun.li@gmail.com

[1] University of California Los Angeles (UCLA), Los Angeles CA 90095, USA
[2] NEC Laboratories America, Inc. (NEC Labs), Princeton NJ 08540, USA
[3] University of Central Oklahoma, Edmond OK 73034, USA
[4] Stellar Cyber, Santa Clara CA 95054, USA

Abstract. Large-scale information systems, such as knowledge graphs (KGs), enterprise system networks, often exhibit dynamic and complex activities. Recent research has shown that formalizing these information systems as graphs can effectively characterize the entities (nodes) and their relationships (edges). Transferring knowledge from existing well-curated source graphs can help construct the target graph of newly-deployed systems faster and better which no doubt will benefit downstream tasks such as link prediction and anomaly detection for new systems. However, current graph transferring methods are either based on a single source, which does not sufficiently consider multiple available sources, or not selectively learns from these sources. In this paper, we propose `MSGT-GNN`, a graph knowledge transfer model for efficient graph link prediction from multiple source graphs. `MSGT-GNN` consists of two components: the *Intra-Graph Encoder*, which embeds latent graph features of system entities into vectors; and the graph transferor, which utilizes graph attention mechanism to learn and optimize the embeddings of corresponding entities from multiple source graphs, in both node level and graph level. Experimental results on multiple real-world datasets from various domains show that `MSGT-GNN` outperforms other baseline approaches in the link prediction and demonstrate the merit of attentive graph knowledge transfer and the effectiveness of `MSGT-GNN`.

Keywords: Knowledge graphs · graph neural network · transfer learning

1 Introduction

Various large-scale information systems, such as knowledge bases (KBs), enterprise security systems, IoT computing systems and social networks [4], exhibit comprehensive interactions and complex relationships among entities from multiple different and interrelated domains. For example, knowledge bases, such as DBpedia [1], contain rich information of real-world entities (people, geographic

locations, etc), normally from multiple domains and languages; and IoT systems contain thousands of mobile interrelated computing devices, mechanical and digital machines with various functions that constantly record surrounding physical environments and interact with each other. These systems can be formulated as heterogeneous graphs with nodes as system entities and edges as activities. Considering an enterprise security system as one example shown in Figure 1 (right), processes, internet sockets, and files can be treated as different types of nodes. Activities between entities, such as a process accessing a destination port or importing system libraries, are treated as edges in the graph. They can be utilized for many downstream tasks including identifying active entities or groups in social networks, inferring new knowledge in KBs and detecting abnormal behaviors [3].

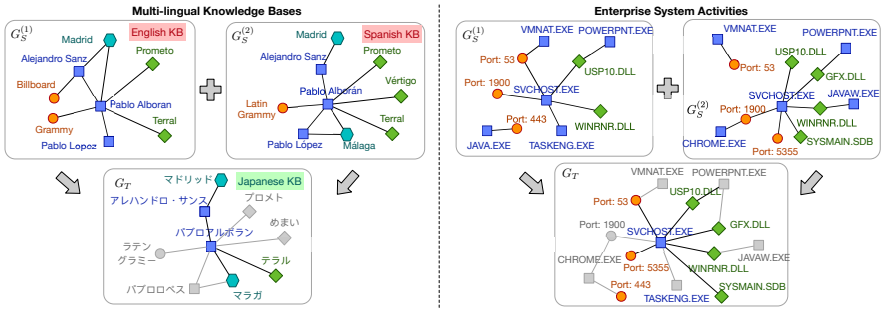


Fig. 1: Two examples of multi-source graph transfer in knowledge bases (left) and enterprise systems (right). By leveraging the entities and relations from sources $G_{S^{(1)}}$ and $G_{S^{(2)}}$, we can estimate the target graph $\hat{G}_T$ based on the current observation G_T. Grey nodes/links in $\hat{G}_T$ denote new predictions from graph knowledge transfer. (Best viewed in color)

Due to the complex nature of real-world systems, it normally takes a long time, sometimes even months for newly-deployed information systems to construct a reliable graph "profile" to identify featured entities and activities. Therefore, there is a need to transfer and migrate knowledge (potential entities with corresponding high-confidence interactions) from other available sources provided by existing multiple well-developed systems. However, directly transferring existing nodes and links by copying is not reasonable and reliable enough since the source and target systems are not necessary for the exact same domains (e.g., transferring knowledge from existing departments to a new department in a corporation). It may transfer irrelevant or even incorrect entities and activities to the target graph. Existing research work [15] mostly focuses on design learning frameworks for effective graph knowledge transfer between one source system and one target system and shows promising results on graph knowledge transfer. But in reality, it is quite common that multiple system sources are available.

Simply using single-source graph knowledge transfer has its own limits: (1) the information from a single source is not sufficient in most cases; and (2) using only one source may lack generalization ability especially when the source and target are largely different, which leads to potential transfer failure. Learning graphs for newly-deployed systems through multi-source graphs will no doubt provide more comprehensive coverage of system entities and activities in multiple domains, and it will be more robust for downstream applications relying on learned target graph after selectively adapting knowledge from source graphs[5]. Two application scenarios are shown in Fig. 1. In the case of multi-lingual KBs, low-resource KB (such as Japanese) can be enriched and improved with other KBs, and especially in the case of `Pablo Alboran` (Spanish pop singer), Spanish KBs can provide better and more accurate knowledge facts than others. Similarly in the example of enterprise systems, after the observation that the system has similar patterns of `.dll` connections of `SVCHOST.EXE`, a reasonable interpretation is that the target graph G_T will more likely grow more closely related patterns shown in source graphs.

However, the aforementioned selective multi-source transfer faces several challenges: (i) *How to represent multiple source graphs and target graphs effectively i.e. set up connections to leverage the graph knowledge in source graphs to the target graphs.* Not all sources are equally related to the target and it is required to differentiate multiple input source graphs in the transferring process, which is a difficult but important task to handle and will significantly affect the transfer performance. (ii) *How to handle potential conflicts on entities and interactions observed in multiple graphs.* The same interactions may be observed in some sources, but are not in others. In other words, there are potentially conflicting observations that cannot be easily tackled by simple transfer. In other words, if all sources are credited equally (for example, using one combined graph to include all the nodes and edges) and other methods that concatenate multiple graphs, one inductive bias is incorrectly assumed that nodes and/or edges are transferred and learned without selectivity and the approaches are subject to noise and misinformation on part of the sources.

To address the aforementioned tasks and corresponding challenges, we proposed a novel type of graph neural network designed for Multi-Source Graph Knowledge Transfer named `MSGT-GNN` which contains two model components: *Intra-Graph Encoder* and *Attention-based Cross-graph Transfer*. The high-level idea is that the knowledge transfer between the source and target graphs is done in a controllable manner where they are selectively learned. We employ self graph encoder model to a variety of state-of-the-art graph neural networks (GNNs) to obtain the node representations, that is, node embeddings learned from the node features itself and neighborhood in the context of the same source/target graph. On top of the encoder model, the Cross-graph Transfer module adopts a novel

[5] In this paper, we use the *source graph* as the graph profiles for existing well-observed systems and *target graph* as the graph profile for new systems, which is relatively smaller than source graphs in graph size (e.g. number of nodes/edges). We assume that the number of source graphs is at least 2 and that of the target graph is 1.

attention mechanism based on both node level and graph level. This module can better learn the representations by attentively aggregating nodes in the broader context, which later applies in the graph decoder for link prediction. As a result, not only can we accelerate the process of graph enlargement to fast characterize the target graph, but we can also selectively and effectively leverage multiple sources in the information systems to estimate more reliable and accurate target graphs. Experimental results on target graph link prediction confirm that the effectiveness of `MSGT-GNN` and the performance of knowledge transfer significantly outperforms other state-of-the-art models including TINET.

2 Problem Statement

Given n multiple source domains $\mathcal{D}_S^{(i)}$ $(i = 1, 2, \ldots, m)$ and one target domain $\mathcal{D}_T$ as input graphs have been on source domain for and these source graphs $G_S^{(i)}$ are stable already. Meanwhile, the system in $\mathcal{D}_T$ is possibly newly deployed and therefore the target graph G_T incomplete and of relatively small size. Our goal is to transfer the graph knowledge (entity and edges) from $G_S^{(i)}$ $(i = 1, 2, \ldots, n)$ to G_T, and then help quickly enlarge and estimate an estimated complete graph $\hat{G}_T$ to fit the domain of $\mathcal{D}_T$, which should be as close to the ground truth $\bar{G}_T$ as possible. Note that in this paper, we assume that alignments of the same entity among source and target graphs are well established, though such alignments are not fully feasible especially in knowledge bases. Under such formulation, we also point out that our proposed problem focuses on the graph enhancement from its incomplete status, different from temporal graph modeling where graphs are dynamically changed with multiple timestamps. Notations of all symbols used in this paper are summarized in Table 1. Scalars, vectors and matrices are denoted with lowercase unbolded letters, lowercase bolded letters and uppercase bolded letters, if not explicitly specified.

We acknowledge that entity alignment may not be flawlessly given in many real-world applications and there are many existing research works lying on the direction of entity disambiguation, etc. As mentioned in Section 2, we point out that in this paper we do not cover the scope of the entity alignments [24,22] (or entity resolution, entity conflation), which essentially predicts the correspondences of the same entity among different graphs. For example, in enterprise graphs, entities are generally identifiable with their IDs; in encyclopedic KGs, some labeled-property graphs are equipped with UID (universal identifier), which significantly reduces the alignment challenge. However, we believe such assumption can be relaxed, that is, MSGT-GNN can be further adapted to partially-given alignment or cross-graph alignment can be jointly learned, corrected, and/or enhanced, which is left as one direction of our future work.

3 Methodology

In this section, we formally propose `MSGT-GNN` to tackle multi-source graph knowledge transfer problem inspired by multi-task learning. As the model ar-

Table 1: Summary of important notations.

Notation	Description
$\mathcal{D}_S^{(i)}$	i-th source domain
$\mathcal{D}_T$	Target domain
$G_S^{(i)}$	The graph of the i-th source from $\mathcal{D}_S^{(i)}$
$\bar{G}_T$, G_T, $\hat{G}_T$	The ground-truth complete / incomplete / estimated complete graph of the target system from $\mathcal{D}_T$
$A_S^{(i)}, A_T$	The adjacency matrix of the i-th source graph $G_S^{(i)}$ / the target graph G_T
$\mathbf{Z}$, $\mathbf{Z}_S^{(i)}$	Embedding table for all N entities, or for $N_S^{(i)}$ entities from the i-th source graph (as output of graph encoders)
$\mathbf{h}^l_{S^{(m)}{}_i}, \mathbf{h}^l_{T_i}$	Embedding of the i-th node in the m-th source graph (or target graph) at the l-th layer of GNN (node embeddings, with node index)
$\mathbf{h}^l_{S^{(m)}}, \mathbf{h}^l_T$	Embedding of the m-th source graph (or target graph), at the l-th layer of GNN (graph embeddings, without node index)

chitecture of `MSGT-GNN` shown in Figure 2, it breaks down into two components: *Intra-graph encoder* and *Cross-Graph transfer*, which are explained in Section 3.1 and Section 3.2 respectively.

3.1 Intra-Graph Encoder

Generally, a graph encoder serves a function to represent nodes by their embeddings, from the original node features (categorical attributes, textual descriptions, etc), based on the graph features. Our proposed Intra-Graph Encoder, as the first component in `MSGT-GNN`, aims to learn the node features in the context of its own graph (source or target), i.e. the graph to which it originally belongs. As discussed in Section 5, graph neural networks (GNNs), deep learning based approaches that operate on graph-structured data, have recently shown effective for various applications such as node classification, link prediction and community detection. A generalized framework of GNNs consists of such a graph encoder, taking as input an adjacency matrix A, as well as original (optional) node features $X = \{X_N\}$. A typical graph encoder parameterized by $\mathbf{\Theta}_{\text{enc}}$ combines the graph structure with node features to produce node embeddings as, $Z = \text{ENC}(A, X, \mathbf{\Theta}_{\text{enc}})$, where Z is the learned comprehensive representation from GNNs and is used for downstream tasks with designated graph decoders.

More specifically, in `MSGT-GNN`, for homogeneous graphs, we choose the Intra-Graph Encoder as standard GCN [12], which can be described as,

$$\mathbf{H}_i^{(l+1)} = \sigma\left(\hat{D}^{-\frac{1}{2}}\hat{A}_G\hat{D}^{-\frac{1}{2}}\mathbf{H}_i^{(l)}W^{(l)}\right), \tag{1}$$

where $\mathbf{H}_i^{(l)} \in \mathbb{R}^{n\times d}$ are embedding of after l-th GCN layers and $\hat{A}_G = A_G + I$ where I is the identity matrix, A_G is adjacency matrix of given graph G, $\hat{D}$ is the

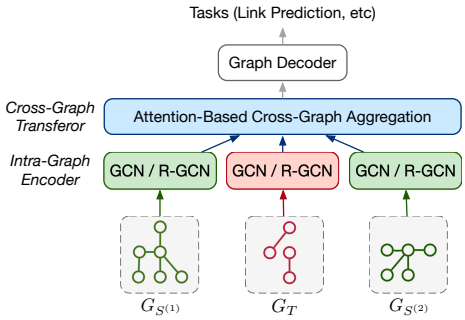


Fig. 2: Model architecture overview for `MSGT-GNN` (two source graphs are shown). Node embeddings across multiple graphs are learned through two-module framework, i.e. *Intra-graph encoder*, which learns node embeddings of its own graph context from initial node features; and *Cross-Graph transfer*, which enables learning through mulitple graphs and node embeddings are updated by its corresponding nodes in other source as well as the graph-level information.

diagonal node degree matrix of $\hat{A}$, as defined in [12].Note that G can be either any source graph $G_{S^{(i)}}$ or target graph G_T. For multi-relational heterogeneous graphs such as knowledge graphs and enterprise systems, we adopt R-GCN [18], which utilizes relation-wise weight matrix,

$$\mathbf{h}_i^{(l+1)} = \sigma\left(\mathbf{W}_0^l \mathbf{h}_i^{(l)} + \sum_{r\in\mathcal{R}} \sum_{j\in\mathcal{N}_i^r} \frac{1}{c_{i,r}} \mathbf{W}_r^l \mathbf{h}_j^{(l)}\right), \tag{2}$$

where $\mathbf{W}_0^l$ is the weight matrix for the node itself and $\mathbf{W}_r^l$ is used specifically for the neighbors having relation r, i.e., $\mathcal{N}_i^r$, $\mathcal{R}$ is the relation set and $c_{i,r}$ is for normalization. Similarly, R-GCN applies both in the source graphs and the target graph. In both cases, the number of GNN layers L is one hyperparameter[6].

3.2 Attention-based Cross-graph Transfer

The goal of our proposed *Cross-graph Transfer* is to provide a valid transfer mechanism in the entity embedding space for multi-source graphs. It is built on top of the Intra-Graph Encoder to enable the node embeddings selectively updated by the cross-graph "neighborhood" in both node level and graph level attention mechanism. Details of *Cross-graph Transfer* are shown in Figure 3.

To prepare for cross-graph transfer, one necessary module is Graph-level Aggregator, which takes the set of node representations and compute graph level representation, as $\mathbf{h}_G = f_G(\{\mathbf{h}_i^G\})$ where $\mathbf{h}_G \in \mathbb{R}^d$, for both source and target

[6] In this work, the performance is relatively insensitive to L where we fix $L = 2$ for GNN modules including baselines

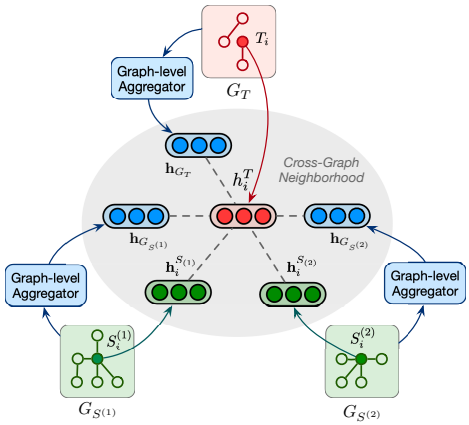


Fig. 3: Details about Cross-Graph Transfer Layer operating on the Node T_i, updated by itself and its corresponding cross-graph neighbors (node-level embeddings), attentively learned from graph-level embeddings (Best viewed in color)

graphs[7]. We use the MLP aggregator following the implementation in [13]. The aggregation function operating on a node i of the target graph is defined as,

$$\mathbf{h}_{T_i}^{l+1} = \sigma\left(\mathbf{W}_0^l \mathbf{h}_{T_i}^l + \sum_m \alpha_m \mathbf{W}_n^l \mathbf{h}_{S^{(m)}{}_i}^l\right), \tag{3}$$

where $\mathbf{W}_0^l$ is the weight matrix for the node itself and W_n^l is used specifically for the cross-graph neighbors (from the given alignments), of the l-th layer. $h_{S^{(m)}{}_i}^l$ denotes the l-th layer's hidden representation of node i in $G_{S^{(m)}}$. α_m is attention weight computed over all m cross-graph neighbors. as,

$$\alpha_m = \text{softmax}\left(\left[\mathbf{h}_i^{S_{(m)}}; \mathbf{h}_{G_{S_{(m)}}}\right]^T \cdot \mathbf{W}_{\text{att}} \cdot \mathbf{h}_{T_i}^l\right), \tag{4}$$

where $\mathbf{W}_{\text{att}} \in \mathbb{R}^{2d \times d}$ and $\left[\mathbf{h}_i^{S_{(m)}}; \mathbf{h}_{G_{S(m)}}\right]$ is the concatenation of node-level cross-graph neighbor embedding and the graph-level embedding. By such cross-graph transfer, the node in one graph will be consequently updated and optimized attentively by nodes from other associated graphs. It is noteworthy to point out that our proposed `MSGT-GNN` does not explicitly differentiate source graphs and target graphs, which means the learned embeddings are not limited to make predictions over the target graph.

[7] Theoretically the embedding dimension of graph-level representation can be different from that of the node-level. For simplicity, we choose both dimensions are the same, that is, $\dim(\mathbf{h}_G) = \dim(\mathbf{h}_{G_i}^l)$, where G refers to either source or target graph.

3.3 Graph Decoder

Graph Decoder and training objective The graph decoder use the learned representation from `MSGT-GNN` for link prediction during the inference stage. For homogeneous graph, we apply inner product to represent the edge plausibility, which is $\text{DEC}(\mathbf{Z}) = \mathbf{h}_i^T \mathbf{h}_j$ where $\mathbf{h}_i, \mathbf{h}_j \in \mathbf{Z}$ ($\mathbf{h}$ is the learned embedding table for all nodes). For multi-relational graph, we apply DistMult score function [32] to represent the edge plausibility, which is $\text{DEC}(\mathbf{Z}) = \mathbf{h}_i^T D \mathbf{h}_j$ where $\mathbf{h}_i, \mathbf{h}_j \in \mathbf{Z}$ and $\mathbf{D}_r$ is a diagonal matrix for relation r. Therefore, the training objective is,

$$\mathcal{L}_G(\mathbf{Z_G}) = \left(\mathbf{Z_G} \mathbf{D}_r {\mathbf{Z_G}}^T - A_G\right)^\theta + \Omega(\mathbf{Z_G}), \tag{5}$$

where $\theta = 2$ in practice and $\Omega\left(\mathbf{Z}_G, \mathbf{w}\right) = \lambda \left|\left|\mathbf{Z_G}\right|\right|_F$ is regularization term. $\mathbf{D_r} = I$ for homogeneous graph.

3.4 Training, Inference and Complexity

Joint training on source and target graphs Considering all the source and target graphs, `MSGT-GNN` minimizes the joint loss with meta-path similarity matrices for multiple graphs, $\mathcal{L} = \mu \sum_i \mathcal{L}_{S^{(i)}} + (1-\mu)\mathcal{L}_T$, where $\mu \in (0, 1)$ is a hyperparameter that explicitly balances the importance of source and target graphs. We use the Adam [11] to optimize the joint loss.

Inference During the inference stage, similar to other graph neural networks with downstream link prediction task, two steps of graph encoders (intra-graph and cross-graph) encodes pairs of nodes (from the target graph only for valid testing) into their representations through the trained GNN with the neighbor nodes (both inside its own graph and other sources) weighted by the graph-level representations. Later such embeddings are forwarded to graph decoder for link prediction which outputs plausibility scores of the given potential edges, as link prediction results.

Complexity Analysis For `MSGT-GNN` with the direct encoder, the overall runtime complexity is $\mathcal{O}(tnd|E|)$, which is linear to the size of total edges in multiple source graphs ($|E|$ is the total number of links in source/target graph). As for model parameter complexity, including all embeddings and transformation functions, the result is $\mathcal{O}(|V|d + nd^2)$ ($|V|$ is the total number of nodes in source/target graphs).

4 Experiments

4.1 Datasets

Three datasets on the knowledge bases, enterprise security and academic scholar community are used in the experiments. Data from a real-world enterprise system are collected from 145 machines from 4 departments (3 used as sources and 1 used as a target) in a period of 30 days, with a size of 3.45GB after integration

Table 2: Dataset statistics.

Dataset	Scholar	Enterprise		DBpedia
		Windows	Linux	
# Graphs	3	5	5	5
# Rel. Types	1	3	3	96
# Nodes	2.1k	10.7k	8.9k	12.5k
# Edges	9.0k	87.9k	62.5k	278.1k

and filtering. The entire enterprise security system contains both Windows and Linux machines and we consider they are disjoint graphs as datasets (named as **Windows and Linux Dataset**). Similar to the example in Figure 1, the entities (nodes) in all graphs are processes, internet sockets and libraries (mostly .dll files) and interactions (edges) between the process to file, process to process and process to internet sockets are observed as links in the dataset.

We also consider alternative datasets that are publicly available and from diverse domains are, (i) encyclopedia knowledge bases i.e. **DBpedia** [1] [8], extracted from five languages (en, es, de, fr, ja) of variant graph sizes and completeness; and (ii) **Aminer**, as one academic scholar community dataset [23][9] from Aminer on five data mining/machine learning related research communities in the past years. The nodes are authors and links are simply co-author relationships, which is essentially a homogeneous graph. More specifically dataset, we consider different languages as different domains in the context of `MSGT-GNN`, and given the graph size of these languages, we adopt two disjoint settings: {en,fr,de}→ja[10] and {en,fr,de}→es. This results in a total of 5 datasets from 3 domains in our experiments. More details are listed in Table 2.

4.2 Baseline Methods

We compare our proposed model `MSGT-GNN` with the following baseline methods:
No Transfer (NT) directly uses the original observed incomplete target graph without any knowledge transfer, that is, $\hat{G}_T = G_T$.
Direct Union Transfer (DUT) directly combines all source graphs and the incomplete target graph, as prediction ("union" graph). That is, DUT outputs a union set on entities and links from all observed graphs without any selection, which means, $\hat{G}_T = G_T + \left(\bigcup_i G_S^{(i)}\right)$.
TINET applies the single graph knowledge transfer framework [15]. To fit the multi-source setting, we choose three variations about TINET models: (i) to

[8] Processed DBpedia dataset are downloadable at: Link.
[9] We use a subset of the co-author networks, which is available at `https://aminer.org/data#Topic-coauthor`.
[10] {en,fr,de}→es means the source graphs are from DBpedia English, French and German KBs and the target is Spanish KB.

Table 3: Results of target graph completion task on 5 different transfer settings from 3 different domains (scholar, enterprise and encyclopedia). The best scores are **bolded**.

Dataset	Scholar	Enterprise		Encyclopedia	
		Windows	Linux	{en, fr, de}→ja	{en, fr, de}→es
NT	0.526 ± 0.000	0.664 ± 0.000	0.656 ± 0.000	0.475 ± 0.000	0.545 ± 0.000
DT	0.398 ± 0.000	0.480 ± 0.000	0.578 ± 0.000	0.299 ± 0.000	0.408 ± 0.000
C-TINET	0.635 ± 0.009	0.727 ± 0.008	0.759 ± 0.009	0.596 ± 0.010	0.764 ± 0.013
U-TINET	0.618 ± 0.015	0.718 ± 0.012	0.733 ± 0.008	0.617 ± 0.014	0.750 ± 0.012
W-TINET	0.644 ± 0.017	0.739 ± 0.011	**0.772 ± 0.017**	0.645 ± 0.022	0.779 ± 0.018
O-TINET	0.622 ± 0.014	0.715 ± 0.012	0.740 ± 0.010	0.620 ± 0.009	0.766 ± 0.011
UT-GCN/RGCN	0.606 ± 0.025	0.700 ± 0.030	0.722 ± 0.019	0.576 ± 0.022	0.756 ± 0.026
UT-GAT/KGAT	0.635 ± 0.018	0.744 ± 0.023	0.750 ± 0.015	0.559 ± 0.012	0.710 ± 0.014
Insta-Only GCN/RGCN	0.597 ± 0.014	0.745 ± 0.012	0.734 ± 0.014	0.661 ± 0.015	0.739 ± 0.021
Insta-Only GAT/KGAT	0.624 ± 0.020	0.742 ± 0.018	0.738 ± 0.021	0.656 ± 0.016	0.724 ± 0.016
UDA-GCN	0.652 ± 0.017	0.735 ± 0.013	0.727 ± 0.016	0.610 ± 0.024	0.688 ± 0.022
`MSGT-GNN`	**0.668 ± 0.016**	**0.776 ± 0.021**	0.768 ± 0.018	**0.685 ± 0.018**	**0.801 ± 0.028**

use the closest[11] source graph as the transfer source, named **C-TINET**; (ii) to use the union graph as defined in DUT, as the single transfer source, named **U-TINET**; iii to use TINET iteratively on multiple sources, i.e. transferring one source once in an order, named **O-TINET**. Best performance is reported among all transfer orders.

W-TINET This method uses the weighted version of TINET for source and target graphs. Extending the single-source graph knowledge transfer model to multi-source, we adopt the same sub-model components (EEM, DCM) but adjust the objective function to be the sum of all source graphs.

Intra-Only GNN only uses *Intra-Graph Encoder* component in `MSGT-GNN` and discards the *Cross-Graph Transfer*. That is, standard GCN [12] is applied for homogeneous graphs and R-GCN [18] is applied for multi-relational graphs which preceded the graph decoder. Alternatively, we also consider existing attention-based graph neural networks (applied on a single graph) i.e. GAT [26]/KGAT [28] as replacement of GCN/R-GCN. (Denoted as "Intra-Only GCN/RGCN" and "Intra-Only GAT/KGAT" respectively).

UT-GNN Similar to Intra-Only GNN, this method applies *Intra-Graph Encoder* component only on the "union graph" from the DUT method which forms one combined graph instead of multiple sources and target graphs. Two options (GCN/RGCN, GAT/KGAT) are still considered except the different graph inputs. (Denoted as "UT-GCN/RGCN" and "UT-GAT/KGAT" respectively)

UDA-GCN It develops a dual attention-based graph convolutional network component and domain adaptive learning module, which jointly exploits local and global consistency for feature aggregation to produce unified representation

[11] Default similarity between the source and target graph is based on the Jaccard index.

for nodes. We replace the decoder module[12] for link prediction instead of node classification in the original paper [30].

4.3 Experiment Setup

Evaluation Protocol Similar to [15] , we adopt F1 score to evaluate the accuracy of the graph completion task on the target system instead of Hit@K or MRR score in knowledge graph completion [13]. In our experiment for multi-graph knowledge transfer, the main result is reported as the average and standard deviation of link prediction (edge) F1 score. As F1 score generally is the harmonic mean of precision and recall, we hereby define the *precision* and *recall* by comparing the estimated links between entities with the ground truth. The precision and recall are defined as: $Precision = N_C/N_E$ and $Recall = N_C/N_T$, where N_C is the number of correctly estimated links, N_E is the number of estimated links in total, and N_T is the number of the ground-truth links. For training, as mentioned in Section 2, we choose one incomplete target graph as the "new" system and complete source graphs from the rest as "old" systems and for training. In addition, e use $m = l/l_{\text{full}}$ as an index of "*graph maturity*", which is defined as the observed number of edges (in training set) l of the target graph and the total number of edges l_{full} recorded in the ground truth target graph.
Hyperparameters In the experiment, we set $m = 0.4$ and $d = 128$ if not specified. The number of GCN/R-GCN layers in Intra-Graph Encoder is set as 2 and The number of Cross-Graph Transfer layers is set as 1. Default node embeddings are initialized by either node categorical features (scholar and enterprise dataset) or BERT sentence embeddings from entity descriptions (KB datasets). Hyperparameters are discussed in Section 4.5 and the supplementary material.

4.4 Results

In this section, we investigate the sensitivity of target graph input maturity m, embedding dimension d and balance weight μ between the source and target graphs, as three key hyperparameters of `MSGT-GNN`, compared with some of the strongest baseline methods. Results on the target graph completion task are shown in Table 3. We observe that `MSGT-GNN` outperforms other baselines in terms of average F-1 score. Especially compared with non-transfer, `MSGT-GNN` achieves an average increase of 0.05 on F1 score among all datasets, which proves that `MSGT-GNN` transfers useful graph knowledge to the target. Also, `MSGT-GNN`

[12] Original code implementation: https://github.com/GRAND-Lab/UDAGCN

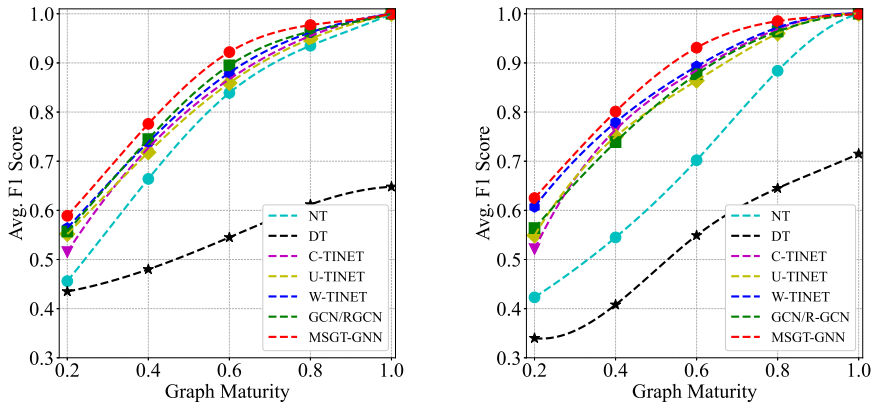
[13] We point out the thread of KG embedding in Section 5, including TransE and recent variants [27]. The limitation of such methods is that they are transductive methods. This is generally not applicable to our inductive learning and its downstream link prediction. However, as for evaluation metrics, we follow the metrics adopted in previous work [15] for target-adapted edge prediction instead of MRR or Hit score for a different triple completion task.

outperforms all the TINET variants in the average F1 score especially on U-TINET and W-TINET which indicates that `MSGT-GNN` adopts a more effective strategy to use multiple sources and learn better latent feature representations of entities with the process graph encoding and domain transferring. Since TINET follows a two-stage (entity selection and edge prediction), the performances significantly decrease when wrong or incomplete entity set is selected for subsequent link prediction. Unlike TINET and its variants, `MSGT-GNN` adopt end-to-end model architecture without explicit steps of entity/node selection. Comparing `MSGT-GNN` and standard GCN/R-GCN or GAT/KGAT, we also observe that `MSGT-GNN` achieves better link prediction performance with a relative gain of 4.9%, which shows the benefit of Cross-Graph Attention Transfer, which can better characterize node latent representations from actively and selectively aggregating useful information from the cross-graph neighborhood. It is noteworthy that NT directly uses the currently observed target graph (incomplete) as output; DT means the union set of all G_S and G_T without any selection. Typically DT includes much more noise and unwanted information into the target graph compared "beneficial section of transfer", i.e., lots of links/edges are falsely predicted as positive. A similar observation is also reported in one of our baselines, TINET. Furthermore, we observe that GAT/KGAT variants almost have similar performance on the task (sometimes even worse). We hypothesize that the attention mechanism adopted by the original GAT/KGAT cannot best selectively learn the knowledge transfer in the cross-graph setting, although recent research shows that they outperform GCN/RGCN on the intra-graph node classification task. It is also noticed that UT-GNN generally performs worse than the Insta-Only setting which indicates that the union graph which equally combines the source graphs without selection has inductive biases which compromise the knowledge transfer in link prediction on the target graph.

4.5 Hyperparameters

In this section, we primarily investigate the sensitivity of target graph input maturity m. Other hyperparameters such as embedding dimension d and balance weight μ between the source and target graphs are discussed in the supplementary material.

Graph maturity m We vary the target graph input by controlling the graph maturity m (let $m = \{0.2, 0.4, 0.6, 0.8, 1.0\}$). From Figure 4, we observe that, for both Windows and DBpedia: {en,fr,en}→es graph, the performance of all models increases when the graph maturity m increases. As other approaches achieve F1 score of 1 when m gets close to 1, direct transfer only achieves around 0.60 as F-1 score, which seems not effective because all the irrelevant entities and links are adopted in the output target graph prediction. On the other hand, given the same level of graph maturity, `MSGT-GNN` achieves the best performance among all other methods on all datasets.



(a) Windows (b) DBpedia: {en,fr,en}→es

Fig. 4: Performances with graph maturity. Most models achieve average F1 score close to 1 as the maturity of input observed target graph grows, while `MSGT-GNN` outperforms other baselines.

5 Related Work

Transfer Learning, Graph Transfer and Multi-source Adaption Transfer learning, domain adaption, and translation [29] have been widely studied in the past decade and played an important role in real-life applications [19] especially on deep transfer learning [14]. Existing transfer learning research is mostly done on the numeric, grid and sequential data, especially image (specific domain classification, style transfer) and text (translation), but research on graphs, networks, or structured data, whose format are relatively less ordered. Some representative work includes *TrGraph* [5], which leverages information via common signature subgraphs. [15] is state-of-the-art and most related research aligned with this direction with two-step learning on entity estimation and dependency reconstruction. The aforementioned methods are mostly based on single-graph knowledge transfer. Note that there is some related work on multi-source adaption that has the same goal of reliable knowledge transfer from multiple sources [16]. However, they are still limited within the domain of images and text rather than graphs. Thus their frameworks cannot be directly applied on graph knowledge transfer. Despite the usage of an attention-based model in transfer, one related work [30] focuses on the node classification task and substantial changes are necessary to make for link prediction in target graphs. We clarify the term of "graph transfer" in Section 2 and distinguish it from other research on the concept of "knowledge transfer" to avoid confusion.

Representation Learning on Knowledge Graphs Graph link prediction is a basic research topic on network analysis. For transfer purposes, [33] presented a transfer learning algorithm to address the edge sign prediction problem using latent topological features from the target and sources. Collective ma-

trix factorization [20] is another major technique. However, these methods are not suitable for dynamics among multiple different domains and the target domain. Another important branch of research related to graph link prediction is network embedding (network representation learning) and similarity search. By representing high-dimensional structured data with embedding vectors, link prediction can be easily performed by node similarity search. These methods can be categorized as meta-path based [21], random walk based [6], matrix factorization based [17] and graph neural networks based methods [7,9]. Similar techniques are applied in multi-relational heterogeneous graphs, i.e. knowledge graphs [25] and their applications [8,10,9]. These embedding based methods (for example [25]) provide insights for representing node features by gathering neighborhood (multi-relational) connections and/or meta-paths and designing graph encoders and decoders. It is worth noted that the most common task over knowledge graphs is triple completion, different from link prediction where focuses on the existence of relations over pairs of nodes in the graph. Another recent research thread along this direction increasingly focuses more on temporal/dynamic graph representation learning [31], which specifically models the graph evolving patterns over time. However, we emphasize that in this work, though it is assumed that the target graphs are relatively incomplete and sparse, we temporarily do not incorporate the time information, as one of the future directions.

Multitask Learning Multitask learning [34] is one emerging active research topic with the rise of artificial intelligence. With the goal of "one model for all tasks", it is widely applied in the area of computer vision and natural language processing. One of the most common approaches in multitask learning is parameter sharing [2]. `MSGT-GNN`is inspired by the similar multi-task learning mechanism considering each graph as one "task", however these frameworks themselves in multitask learning is not applicable for our settings.

6 Conclusion and Future work

In this paper, we formulate a challenging problem on the necessity and benefits of transferring from multi-source graphs into the target graph and then propose `MSGT-GNN`, with the intra-graph Encoder and attention-based cross-graph transfer as major model components. `MSGT-GNN` addresses the challenges and accelerates high-quality knowledge transfer and graph enhancement in the target newly-observed system. Experiments show that `MSGT-GNN` can successfully transfer useful graph knowledge from multiple sources and enable fast target graph construction. For future improvements, one important extension is to temporal graph modeling where we can dive deep into how target graphs grow on newly-deployed systems can grow with the development from multiple sources, which significantly improves explainability on the graph knowledge transfer.

Acknowledgements This work was primarily done and supported during the internship at NEC Laboratories America, Inc (NEC Labs). We thank Dr. Zong

Bo for research discussions. We also would like to thank the anonymous reviewers for their insightful and constructive comments.

References

1. Auer, S., Bizer, C., Kobilarov, G., Lehmann, J., Cyganiak, R., Ives, Z.: Dbpedia: A nucleus for a web of open data. In: The semantic web, pp. 722–735. Springer (2007)
2. Caruana, R.: Multitask learning. Machine learning **28**(1), 41–75 (1997)
3. Cheng, W., Zhang, K., Chen, H., Jiang, G., Chen, Z., Wang, W.: Ranking causal anomalies via temporal and dynamical analysis on vanishing correlations. In: Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. pp. 805–814 (2016)
4. Dong, B., Chen, Z., Wang, H., Tang, L.A., Zhang, K., Lin, Y., Li, Z., Chen, H.: Efficient discovery of abnormal event sequences in enterprise security systems. In: Proceedings of the 2017 ACM on Conference on Information and Knowledge Management. pp. 707–715 (2017)
5. Fang, M., Yin, J., Zhu, X., Zhang, C.: Trgraph: Cross-network transfer learning via common signature subgraphs. IEEE Transactions on Knowledge and Data Engineering **27**(9), 2536–2549 (2015)
6. Grover, A., Leskovec, J.: node2vec: Scalable feature learning for networks. In: Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining. pp. 855–864 (2016)
7. Hamilton, W., Ying, Z., Leskovec, J.: Inductive representation learning on large graphs. Advances in neural information processing systems **30** (2017)
8. Hao, J., Ju, C.J.T., Chen, M., Sun, Y., Zaniolo, C., Wang, W.: Bio-joie: Joint representation learning of biological knowledge bases. In: Proceedings of the 11th ACM International Conference on Bioinformatics, Computational Biology and Health Informatics. pp. 1–10 (2020)
9. Hao, J., Lei, C., Efthymiou, V., Quamar, A., Özcan, F., Sun, Y., Wang, W.: Medto: Medical data to ontology matching using hybrid graph neural networks. In: Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining. pp. 2946–2954 (2021)
10. Hao, J., Zhao, T., Li, J., Dong, X.L., Faloutsos, C., Sun, Y., Wang, W.: P-companion: A principled framework for diversified complementary product recommendation. In: Proceedings of the 29th ACM International Conference on Information & Knowledge Management. pp. 2517–2524 (2020)
11. Kingma, D.P., Ba, J.: Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980 (2014)
12. Kipf, T.N., Welling, M.: Semi-supervised classification with graph convolutional networks. In: International Conference on Learning Representations (2017)
13. Li, Y., Gu, C., Dullien, T., Vinyals, O., Kohli, P.: Graph matching networks for learning the similarity of graph structured objects. In: International conference on machine learning. pp. 3835–3845. PMLR (2019)
14. Long, M., Zhu, H., Wang, J., Jordan, M.I.: Deep transfer learning with joint adaptation networks. In: International conference on machine learning. pp. 2208–2217. PMLR (2017)
15. Luo, C., Chen, Z., Tang, L.A., Shrivastava, A., Li, Z., Chen, H., Ye, J.: Tinet: Learning invariant networks via knowledge transfer. In: Proceedings of the 24th

ACM SIGKDD International Conference on Knowledge Discovery & Data Mining. pp. 1890–1899 (2018)
16. Mansour, Y., Mohri, M., Rostamizadeh, A.: Domain adaptation with multiple sources. Advances in neural information processing systems **21** (2008)
17. Qiu, J., Dong, Y., Ma, H., Li, J., Wang, K., Tang, J.: Network embedding as matrix factorization: Unifying deepwalk, line, pte, and node2vec. In: Proceedings of the eleventh ACM international conference on web search and data mining. pp. 459–467 (2018)
18. Schlichtkrull, M., Kipf, T.N., Bloem, P., Berg, R.v.d., Titov, I., Welling, M.: Modeling relational data with graph convolutional networks. In: European semantic web conference. pp. 593–607. Springer (2018)
19. Shin, H.C., Roth, H.R., Gao, M., Lu, L., Xu, Z., Nogues, I., Yao, J., Mollura, D., Summers, R.M.: Deep convolutional neural networks for computer-aided detection: Cnn architectures, dataset characteristics and transfer learning. IEEE transactions on medical imaging **35**(5), 1285–1298 (2016)
20. Singh, A.P., Gordon, G.J.: Relational learning via collective matrix factorization. In: Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining. pp. 650–658 (2008)
21. Sun, Y., Han, J., Yan, X., Yu, P.S., Wu, T.: Pathsim: Meta path-based top-k similarity search in heterogeneous information networks. Proceedings of the VLDB Endowment **4**(11), 992–1003 (2011)
22. Sun, Z., Zhang, Q., Hu, W., Wang, C., Chen, M., Akrami, F., Li, C.: A benchmarking study of embedding-based entity alignment for knowledge graphs. Proceedings of the VLDB Endowment **13**(11) (2020)
23. Tang, J., Sun, J., Wang, C., Yang, Z.: Social influence analysis in large-scale networks. In: Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining. pp. 807–816 (2009)
24. Trivedi, R., Sisman, B., Dong, X.L., Faloutsos, C., Ma, J., Zha, H.: Linknbed: Multi-graph representation learning with entity linkage. In: Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers). pp. 252–262 (2018)
25. Vashishth, S., Sanyal, S., Nitin, V., Talukdar, P.: Composition-based multi-relational graph convolutional networks. In: ICLR (2020)
26. Veličković, P., Cucurull, G., Casanova, A., Romero, A., Liò, P., Bengio, Y.: Graph Attention Networks. International Conference on Learning Representations (2018)
27. Wang, Q., Mao, Z., Wang, B., Guo, L.: Knowledge graph embedding: A survey of approaches and applications. IEEE Transactions on Knowledge and Data Engineering **29**(12), 2724–2743 (2017)
28. Wang, X., He, X., Cao, Y., Liu, M., Chua, T.S.: Kgat: Knowledge graph attention network for recommendation. In: Proceedings of the 25th ACM SIGKDD international conference on knowledge discovery & data mining. pp. 950–958 (2019)
29. Weiss, K., Khoshgoftaar, T.M., Wang, D.: A survey of transfer learning. Journal of Big data **3**(1), 1–40 (2016)
30. Wu, M., Pan, S., Zhou, C., Chang, X., Zhu, X.: Unsupervised domain adaptive graph convolutional networks. In: Proceedings of The Web Conference 2020. pp. 1457–1467 (2020)
31. Wu, Z., Pan, S., Chen, F., Long, G., Zhang, C., Philip, S.Y.: A comprehensive survey on graph neural networks. IEEE transactions on neural networks and learning systems **32**(1), 4–24 (2020)

32. Yang, B., Yih, S.W.t., He, X., Gao, J., Deng, L.: Embedding entities and relations for learning and inference in knowledge bases. In: International Conference on Learning Representations (2015)
33. Ye, J., Cheng, H., Zhu, Z., Chen, M.: Predicting positive and negative links in signed social networks by transfer learning. In: Proceedings of the 22nd international conference on World Wide Web. pp. 1477–1488 (2013)
34. Zhang, Y., Yang, Q.: A survey on multi-task learning. IEEE Transactions on Knowledge and Data Engineering (2021)