Transfer Learning for Deep Learning on Graph-Structured Data

Jaekoo Lee, Hyunjae Kim, Jongsun Lee and Sungroh Yoon[†]

Department of Electrical Engineering and Computer Science
Seoul National University
Seoul 08826, Korea
[†]sryoon@snu.ac.kr

Abstract

Graphs provide a powerful means for representing complex interactions between entities. Recently, deep learning approaches are emerging for representing and modeling graph-structured data, although the conventional deep learning methods (such as convolutional neural networks and recurrent neural networks) have mainly focused on grid-structured inputs (image and audio). Leveraged by the capability of representation learning, deep learning based techniques are reporting promising results for graph applications by detecting structural characteristics of graphs in an automated fashion. In this paper, we attempt to advance deep learning for graph-structured data by incorporating another component, transfer learning. By transferring the intrinsic geometric information learned in the source domain, our approach can help us to construct a model for a new but related task in the target domain without collecting new data and without training a new model from scratch. We thoroughly test our approach with large-scale real corpora and confirm the effectiveness of the proposed transfer learning framework for deep learning on graphs. According to our experiments, transfer learning is most effective when the source and target domains bear a high level of structural similarity in their graph representations.

Introduction

Recently, many deep neural network models have been adopted successfully in various fields (LeCun, Bengio, and Hinton 2015; Schmidhuber 2015). In particular, convolutional neural networks (CNN) (Krizhevsky, Sutskever, and Hinton 2012) for image/video recognition and recurrent neural networks (RNN) (Sutskever, Vinyals, and Le 2014) for speech and natural language processing (NLP) often deliver unprecedented level of performance. Deep learning has also triggered advances in implementing human-level intelligence, for example, in the game of Go (Silver et al. 2016).

CNN and RNN extract data-driven features from input data (such as image, video, and audio data) structured in (typically low-dimensional) regular grids (see Fig. 1, top). Such grid structure is often assumed to have statistical characteristics (such as stationarity and locality) to facilitate the modeling process. Learning algorithms then take advantage of the assumption and boost performance by reducing the complexity of parameters (Schmidhuber 2015; Bruna et al. 2013; Henaff, Bruna, and LeCun 2015).

In reality, there exist a wide variety of data types in which we need more general, non-grid structure to represent and model complex interactions among entities. Examples include social media mining and protein interaction studies. For such applications, a graph can provide a natural way of representing entities and their interactions by nodes and edges (Deo 2016). For graph-structured input, it is known to be more challenging to find the statistical characteristics we can assume for grid-structured input (Bruna et al. 2013; Henaff, Bruna, and LeCun 2015).

Theoretical challenges including the above and practical limitations (such as data quantity/quality and training efficiency) make it difficult to apply conventional deep learning approaches directly, igniting research on adapting deep learning to graph-structured data (Bruna et al. 2013; Henaff, Bruna, and LeCun 2015; Jain et al. 2015; Li et al. 2015). In many graph analysis methods, structural properties derived from input graphs play a crucial role in uncovering hidden patterns (Koutra, Vogelstein, and Faloutsos 2013; Lee, Kim, and Yoon 2015). The representation learning capability of deep networks can help to detect data-driven structural features in an automated fashion, and deep learning approaches have reported promising results.

In this paper, we attempt to advance deep learning for graph-structured data by incorporating another key component, transfer learning (Pan and Yang 2010). By overcoming the common assumption in machine learning that training and test data should be drawn from the same feature space and distribution, transfer learning between task domains can alleviate the burden of collecting data and training models for a new task, which is similar to an existing task. Given the importance of structural characteristics in graph analysis, the core of our proposal is to transfer the data-driven structural features learned by deep neural networks from a source domain (graph) to a target domain, as informally shown in Fig. 1 (bottom). In the context of graphs, we call the transferred information the *intrinsic geometric* information.

Starting from this intuitive baseline, we need to fill up many details to implement transfer learning for deep learning on graph-structured data. In particular, we need to answer two important questions: (**Q1**) under what condition we can expect a successful knowledge transfer between task

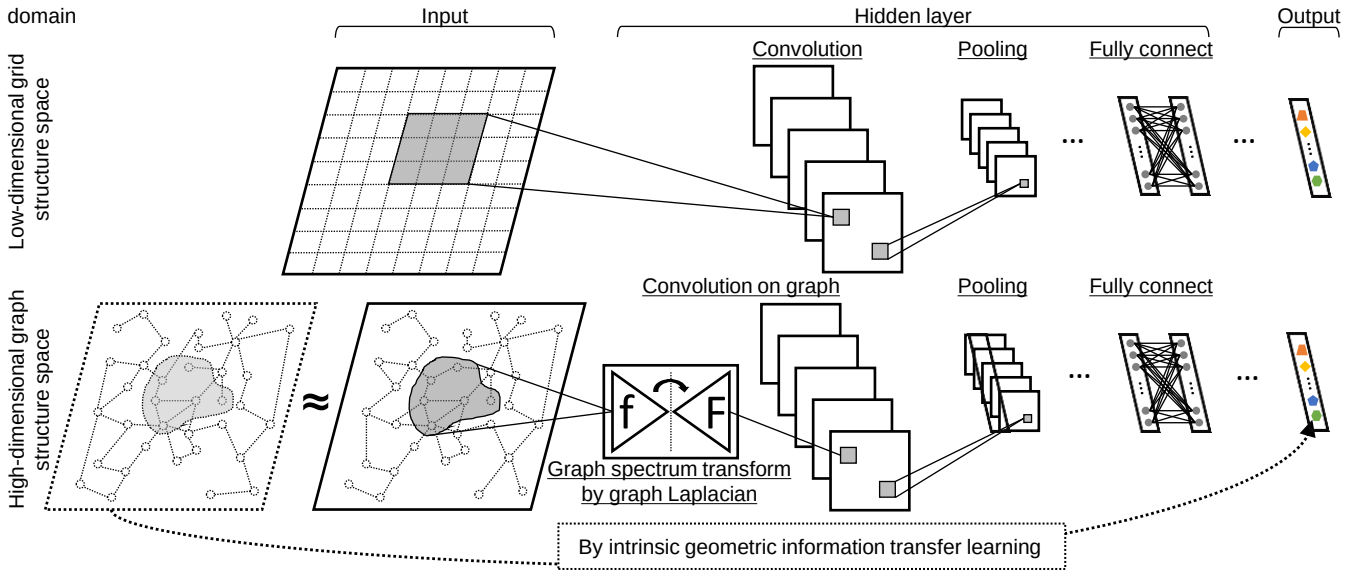

Figure 1: Conventional CNN works on a regular grid domain (top); proposed transfer learning framework for CNN, which can transfer intrinsic geometric information obtained from a source graph domain to a target graph domain (bottom).

domains and (**Q2**) how to actually perform the transfer most effectively. This paper tries to address these questions.

To demonstrate the effectiveness of our approach, we test it with large-scale public NLP datasets for text classification (Zhang, Zhao, and LeCun 2015). Each dataset contains a corpus of news articles, Internet reviews, or ontology entries. We represent a dataset (e.g., Amazon reviews) by a graph to capture the interactions among the words in the dataset. We then use the spectral CNN (SCNN) (Bruna et al. 2013; Henaff, Bruna, and LeCun 2015) to model the graph using neural nets. The learned model can be used for classifying (unseen) texts from the same data source (Amazon). Furthermore, our experimental results confirm that our transfer learning methodology allows us to implicitly derive a model for classifying texts from another source (e.g., Yelp reviews) without collecting new data and without repeating all the learning procedures from scratch.

Our contributions can be summarized as follows:

- We propose a new transfer learning framework for deep learning on input data in non-grid structure such as graphs. To the best of the authors' knowledge, this work is the first attempt in its kind. Adopting our approach will relive the burden of re-collecting data and re-training models for related tasks.
- To address **Q1**, we investigate the conditions for successful knowledge transfers between graph domains. We conjecture that two graphs with similar structural characteristics will give better results and confirm it by comparing graph similarity and transfer learning accuracy.
- To answer **Q2**, we test diverse alternatives to the components of the proposed framework: graph generation, input representation, and deep network construction. In particular, to improve the SCNN model for extracting data-driven structural features from graphs, we analyze and optimize the key factors that affect the performance of SCNN (e.g., the method to quantify spectral features of a graph).
- We perform an extensive set of experiments, using not only synthetic but also real data, to show the effectiveness of our approach.

Related Work

A graph provides a general way of representing diverse interactions of entities and has been studied extensively in various fields and applications (Sonawane and Kulkarni 2014). In addition to studies on representation and quantification of the relations and similarities (Koutra, Vogelstein, and Faloutsos 2013), various studies focus on large-scale graph data and use of structural information. Recently, deep learning methods to automatically extract structural characteristics from graphs have been proposed (Duvenaud et al. 2015; Li et al. 2015).

As for deep learning applied to non-grid, non-Euclidean space, examples include a study on creating graph wavelets by applying deep auto-encoders to graphs and using the properties of automatically extracted features (Rustamov and Guibas 2013). There exists a dee learning approach to analyzing the molecular fingerprints of proteins saved as graphs (Duvenaud et al. 2015). CNN-based model exists for handling tree structures in the context of programming language processing (Mou et al. 2016). The localized SCNN model (Boscaini et al. 2015) is a deep learning approach that can extract the properties of the deformable shape.

The (general) SCNN model (Bruna et al. 2013; Henaff, Bruna, and LeCun 2015), a key component in our framework, borrowed the Fourier transform concept from the signal process field in order to apply the CNN model of a grid domain to a graph-structured domain. In this model, the convolutional operation was re-defined for graphs.

Proposed method

Fig. 2 presents a diagram illustrating the overall flow of the proposed method, which consists of five steps A–E. The first three steps are to produce a graph from input and to identify unique structural features from the graph. The last two steps are to apply transfer learning based on the learned features and graph similarity to carry out inference.

Step A: Producing Graph

We represent data elements of input data and their interactions/relations as nodes and edges in a graph, respectively. More formally, from an input dataset, we construct an undirected, connected, weighted graph $G = (V, E, A)$, where V and E represent the sets of vertices and edges, respectively, and A denotes the weighted adjacency matrix. Assume that $|V| = N$ and $|E| = M$.

We utilize two recent techniques to derive a graph (more specifically, the edge set E) from input data: co-occurrence graph estimation (CoGE) (Sonawane and Kulkarni 2014) and supervised graph estimation (SGE) (Henaff, Bruna, and LeCun 2015). CoGE directly quantifies the closeness of data elements based on the frequency of co-occurrence, while SGE automatically learns a similarity features among elements through a fully connected network model.

B: Representation of Graphs in Spectral Domain

We extract the intrinsic geometric characteristics of the entire graph by deriving (non-)normalized Laplacian matrix L of the graph constructed in step A. For a graph domain, L provides the values for graph spectral bases in the convolution operation of SCNN (Mohar 1997; Koutra, Vogelstein, and Faloutsos 2013).

We consider three types of L: the non-normalized Laplacian (L^{basic}), the random walk-based normalized Laplacian (L^{rw}), and the random walk with restart based normalized Laplacian (L^{rwr}) given by (Tong, Faloutsos, and Pan 2006):

$$L^{\text{basic}} = D - A \tag{1}$$

$$L^{\text{rw}} = D^{-1}(D - A) = I - D^{-1}A \tag{2}$$

$$L^{\text{rwr}} = [I + \epsilon^2 D - \epsilon A]^{-1} \approx [I - \epsilon A]^{-1} \tag{3}$$

$$\approx I + \epsilon A + \epsilon^2 A^2 + \cdots \tag{4}$$

where D represents the degree matrix of the graph and ϵ represents the probability of restart. Note that the approximation in Eq. (3) is by attenuating neighboring influence, while the approximation in Eq. (4) is by belief propagation and its fast approximation.

L is a symmetric matrix that can be decomposed through the diagonalization by combining eigenvalues λ_l and the corresponding orthogonal eigenvectors $u_l(n)$, where l is the order of an eigenvalue, and $n \in [1, N]$ is the index of a node (Mohar 1997). Recall that a function $f : V \mapsto \mathbb{R}$ defined on the nodes of graph G can be represented by a vector $f \in \mathbb{R}^N$ with the n-th dimension of f indicating the value at the n-th vertex in V (Shuman et al. 2013; Shuman, Ricaud, and Vandergheynst 2016). As in the Fourier transform, the eigenfunctions of L represent a function f defined by the nodes in a graph: $f_G(n) = \sum_{l=0}^{N-1} \hat{f}_G(\lambda_l) u_l(n) \leftrightarrow \hat{f}_G(\lambda_l) = \sum_{n=1}^{N} f_G(n) u_l(n)$, where $\hat{f}$, the transformed function of f, is represented by a set of basis eigenvectors. The Parseval theorem also holds, and $\langle f, g \rangle = \langle \hat{f}, \hat{g} \rangle$ for two functions f and g, verifying the consistency between the two dimensions (Chung 1997; Shuman, Ricaud, and Vandergheynst 2016).

This indicates that an input function defined on the vertex domain of a graph can be converted into the corresponding graph spectral domain by using the concept of Fourier analysis on graphs as above. The generalized convolutional operation (denoted by $*_G$) of functions f and g can be defined by the diagonalized linear multiplication in the spectral domain as follows (Bruna et al. 2013; Henaff, Bruna, and LeCun 2015; Shuman, Ricaud, and Vandergheynst 2016):

$$(f *_G g)(n) = \sum_{l=0}^{N-1} \hat{f}(\lambda_l) \hat{g}(\lambda_l) u_l(n)$$

which can also be expressed as

$$f *_G g = \hat{g}(L) f = U \begin{bmatrix} \hat{g}(\lambda_0) & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & \hat{g}(\lambda_{N-1}) \end{bmatrix} U^T f \tag{5}$$

where U is a matrix having the eigenvectors of the graph Laplacian in its columns that quantify the intrinsic structural geometry of the entire graph domain and serve as the spectral bases of a graph. This matrix functions as the Fourier transform into the graph spectral domain. In this regard, a receptive filter learnt through the convolution operation on the convolution layer of a CNN model for a regular grid domain is regarded as a matrix on g, which is diagonalized by $\hat{g}(\lambda_i), 0 \leq i \leq N-1$ elements on input f defined in a graph domain provided by Eq. (5).

For SCNN, the transform of input x_k of size $n \times f_{k-1}$ into output x_{k+1} of size $n \times f_k$ is given by

$$x_{k+1,j} = h\left(U \sum_{i=1}^{f_{k-1}} F_{k,i,j} U^T x_{k,i} \right), \; j = 1, \ldots, f_k \tag{6}$$

where h is a nonlinear function and $F_{k,i,j}$ is a diagonal matrix. This implies that training the weights of learnable filters are the same as training the multipliers on the eigenvalues of the Laplacian (Bruna et al. 2013; Henaff, Bruna, and LeCun 2015). This characterizes SCNN, a generalized CNN model that has several filter banks through generalized convolutional operations in a graph.

We augment the SCNN model so that it supports spatial locality, which is made independent of input size by using windowed smoothing filters. They are defined as $\hat{P}_k(l) = \sum_{k=0}^{K} a_k \lambda_l^k$ for $K < N$, based on the polynomial kernel a_k with degree K (Shuman, Ricaud, and Vandergheynst 2016). This is based on the fact (in signal processing) that the smoothness in the spectral domain can have spatial decay or local features in the original domain. This idea can be implemented by using the eigenvectors of the subsampled Laplacian as the low-frequency eigenvectors of the Laplacian (Boscaini et al. 2015).

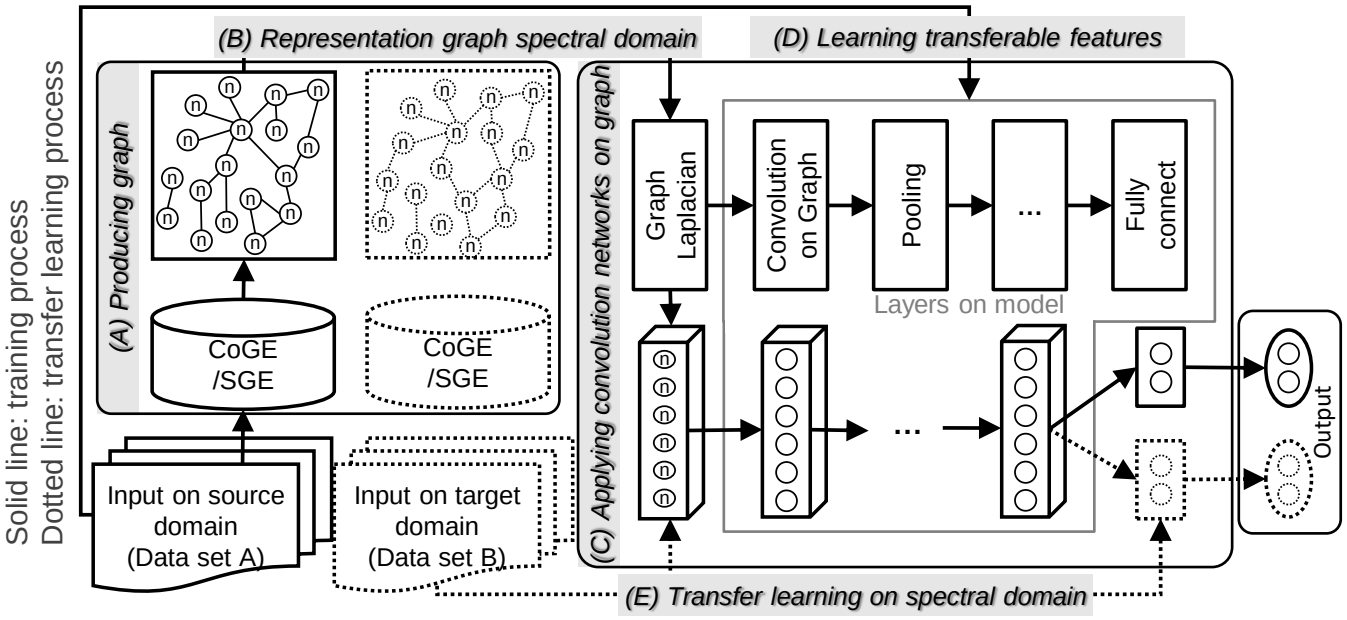

Figure 2: Overview of the proposed method

C: Applying Convolution Networks on Graph

We train the SCNN model by using the information obtained through the previous steps to represent the geometric information of local behaviors from the surface of a structural graph domain. The model has hierarchical structure consisting of layers for convolutional and pooling, and a fully connected layer as shown in Fig. 2. The training determines the weights of each layer by minimizing the task-specific cost (loss) function. The model can learn various data-driven features by re-defining the convolution operation with the spectral information of the structural graph domain (Bruna et al. 2013; Henaff, Bruna, and LeCun 2015).

D: Learning Transferable Features

Once the model training is completed, it contains data-driven features for the graph-structured data derived from the input through steps A and B. As stated in Introduction, the core of our proposal is to transfer the information on structural characteristics of a graph learned by deep learning. The features learned in step C provide such information.

E: Transfer Learning on Spectral Domain

For a new task in the target domain, we transfer the intrinsic geometric information learned from the source domain through steps A–D. We skip the steps to generate data for the new task and to extract the structural characteristics from the graph representation. We directly build the model for the new task by by (1) copying the convolutional and pooling layers, which contain trained features from the source domain, and (2) training the fully connected layer for fine tuning weights for classification task in the target domain.

This way of transfer learning provides efficiency in learning and also helps to minimize the problems occurring from lack of data and imperfect structural information for the new task. Note that the proposed method guarantees the spectral homogeneity of graphs by using the union of node sets on the heterogeneous source and target datasets. In our method, it is possible to utilize the spectral features of graphs from heterogeneous datasets.

Results and Discussion

We tested the proposed method by performing topic classification of text documents. Text data carry information on not only individual words but also their relationships, and graph-based methods are widely used for text mining. As stated in Introduction, we utilized large-scale public NLP data (Zhang, Zhao, and LeCun 2015), which contain multiple corpora of news articles, Internet reviews, or ontology entries (Table 1). For controlled experiments, we also generated two pairs of synthetic datasets by random sampling of the real corpora. One pair consists of two corpora with high similarity, and the other pair consists of two corpora with low similarity. For measuring the structural similarity between graphs (as shown in Table 1 and Fig. 3), we used the methods reported by existing previous work (Koutra, Vogelstein, and Faloutsos 2013; Lee, Kim, and Yoon 2015); refer to the footnote below Table 1 for more details. Note that YELP and AMAZ bear the highest similarity in terms of the used metrics. We implemented the deep networks with Torch and Cuda. We used AdaGrad as the optimizer and ReLU as the activation. We used 10-fold cross validation.The proposed method can offer an efficient training scheme which shows relatively low computation cost of $-O(n^{2.376})$ by leaving out the eigenvalue decomposition on SCNN and reusing the model trained by the data in source domain.

We first carried out comprehensive experiments to de-

Table 1: Details of the real-world datasets used

	AG	DBP	YELP	AMAZ
train	120,000	560,000	580,000	3,600,000
test	7,600	70,000	38,000	400,000
class	4	14	2	2
Name	$sim(G_1, G_2)^*$ [$corr(wv_1, wv_2)^\dagger$]			
AG		0.37 [0.45]	0.28 [0.36]	0.35 [0.42]
DBP	0.37 [0.45]		0.23 [0.29]	0.33 [0.40]
YELP	0.28 [0.36]	0.23 [0.29]		0.50 [0.58]
AMAZ	0.35 [0.42]	0.33 [0.40]	0.50 [0.58]	

AG: a corpus of news articles on the web, DBP: ontology data from DBpedia, YELP: reviews from Yelp, AMAZ: reviews form Amazon.

* $sim(G_1, G_2) = 0$ indicates that two graphs G_1 and G_2 are structurally complementary, whereas the value of 1 means that they are identical.

† $corr(wv_1, wv_2)$ represents the correlation between the log-normalized bag of words extracted from each of the text corpora.

Table 2: Performance of SCNN model with various hyperparameters for text topic classification task

Model architecture*	Graph generation	L type	Classification accuracy AG	DBP	YELP	AMAZ
GC8-FC500	CoGE	L^{basic}	0.891	0.947	0.906	0.883
GC8-FC500	CoGE	L^{rw}	0.893	0.948	0.907	0.883
GC8-FC500	CoGE	L^{rwr}	0.891	0.934	0.903	0.878
GC8-FC500	SGE	L^{basic}	0.906	0.969	0.912	0.885
GC8-FC500	SGE	L^{rw}	0.906	0.971	0.913	0.884
GC8-FC500	SGE	L^{rwr}	0.890	0.951	0.909	0.885
GC8-FC1000	CoGE	L^{basic}	0.895	0.950	0.926	0.884
GC8-FC1000	CoGE	L^{rw}	0.895	0.970	0.920	0.885
GC8-FC1000	CoGE	L^{rwr}	0.893	0.962	0.915	0.878
GC8-FC1000	SGE	L^{basic}	0.906	0.971	0.915	0.877
GC8-FC1000	SGE	L^{rw}	0.907	0.969	0.920	0.883
GC8-FC1000	SGE	L^{rwr}	0.890	0.958	0.908	0.875
GC8-GC8-FC1000	CoGE	L^{basic}	0.890	0.957	0.915	0.889
GC8-GC8-FC1000	CoGE	L^{rw}	0.891	0.972	0.916	0.888
GC8-GC8-FC1000	CoGE	L^{rwr}	0.890	0.968	0.916	0.879
GC8-GC8-FC1000	SGE	L^{basic}	0.905	0.971	0.915	0.883
GC8-GC8-FC1000	SGE	L^{rw}	0.908	0.971	0.916	0.878
GC8-GC8-FC1000	SGE	L^{rwr}	0.894	0.973	0.915	0.890

* For training, we set the kernel degree $K = 60$, learning rate to 0.01 and used cross-entropy cost function with AdaGrad optimizer. GC# means the use of graph convolutional layers with # feature maps, and FC# means the use of fully connected layer with # hidden units.

termine what factors affect the performance of the SCNN model for graph modeling. Table 2 lists a part of the results we obtained by varying the net architecture, the method to generate graphs, and the type of Laplacian matrix along with the resulting classification accuracy for each combination. We can observe from Table 2 that the Laplacian methods do not significantly affect the performance, but L^{rwr} has the benefit in terms of computational complexity. SGE tended to give more accurate results than CoGE, which implies that the initial graph generation affects the model training more critically than structural feature extraction. For the experiments shown in Table 2, the GC8-GC8-FC1000 model (refer to the footnote for notation) gave the best results, and we used this model as our main learning model in the following experiments.

We then performed experiments to see the effectiveness of transfer learning using the synthetic datasets. The results are shown in Fig. 3, in which the plots in the top row are from the pair of synthetic corpora with high similarity [$sim(G_1, G_2) = 0.75$ and $corr(wv_1, wv_2) = 0.95$] for varying quantities of fine-tuning data for training the transferred model in the target domain (1%, 3%, 5%, and 10% of the entire target data). The plots in the top of Figure. 3 correspond to the results from the pair of synthetic corpora with low similarity [$sim(G_1, G_2) = 0.30$ and $corr(wv_1, wv_2) = 0.50$]. We can observe that transfer learning is more effective for the higher similarity case (top), in which the test accuracy of the transferred model increased significantly faster than that of the source domain model. Using only 1% of the target domain data was sufficient for training, and using more data did not give noticeable difference. For the lower similarity case (bottom), the training in the target domain was limited and could not deliver the same level of accuracy in the source domain, due to the discrepancies in the underlying structure between the source and target domains.

Finally, we tested our approach with four corpora (AG, DBP, YELP, and AMAZ) as shown in Fig. 4. The plots in the top row represent the test accuracy of the model trained with the original data (solid line) and those of the transferred model trained with each of the other data (dotted line). The bottom plots represent the test loss. For the two corpora with the highest level of similarity (YELP and AMAZ), the effect of transfer learning was most salient. The test accuracy of the transferred model was comparable to that of the source model (for YELP) or was only 5–8% lower (for AMAZ). For the other cases with lower similarity than YELP and AMAZ, transfer learning was less effective. This results again confirm our observation that the knowledge transfer is most successful when the source and target domains have high level of structural similarity between underlying graph representations.

Conclusion

We have proposed a new transfer learning framework for deep learning on graph-structured data. Our approach can transfer the intrinsic geometric information learned from the graph representation of the source domain to the target domain. We observed that the knowledge transfer between tasks domains is most effective when the source and target domains possess high similarity in their graph representations. We anticipate that adopting our methodology will help extend the territory of deep learning to data in non-grid structure as well as to the cases with limited quantity and quality of data. To prove this, we are planning to apply our approach to diverse datasets in different domains.

References

Boscaini, D.; Masci, J.; Melzi, S.; Bronstein, M. M.; Castellani, U.; and Vandergheynst, P. 2015. Learning class-specific descriptors for deformable shapes using localized spectral

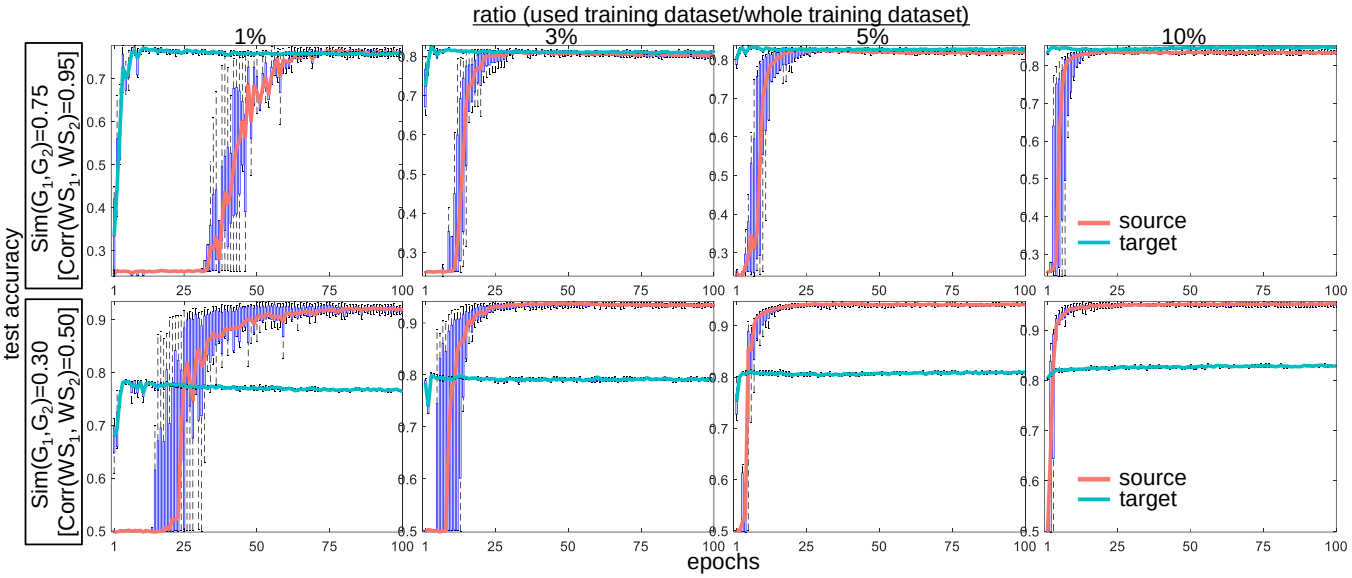

Figure 3: Results of intrinsic geometric information transfer learning for synthesized datasets (best viewed in color). Top: source and target datasets have high similarity in graph representations; bottom: source and target datasets have low similarity. Each column: the percentage (1%, 3%, 5% and 10%) of the target dataset used for training the transferred model (fine tuning the fully connected layer). We repeated every experiment 10 times, and each data point shows a boxplot; red (source domain) and blue (target domain) lines connect the median locations of the boxplots.

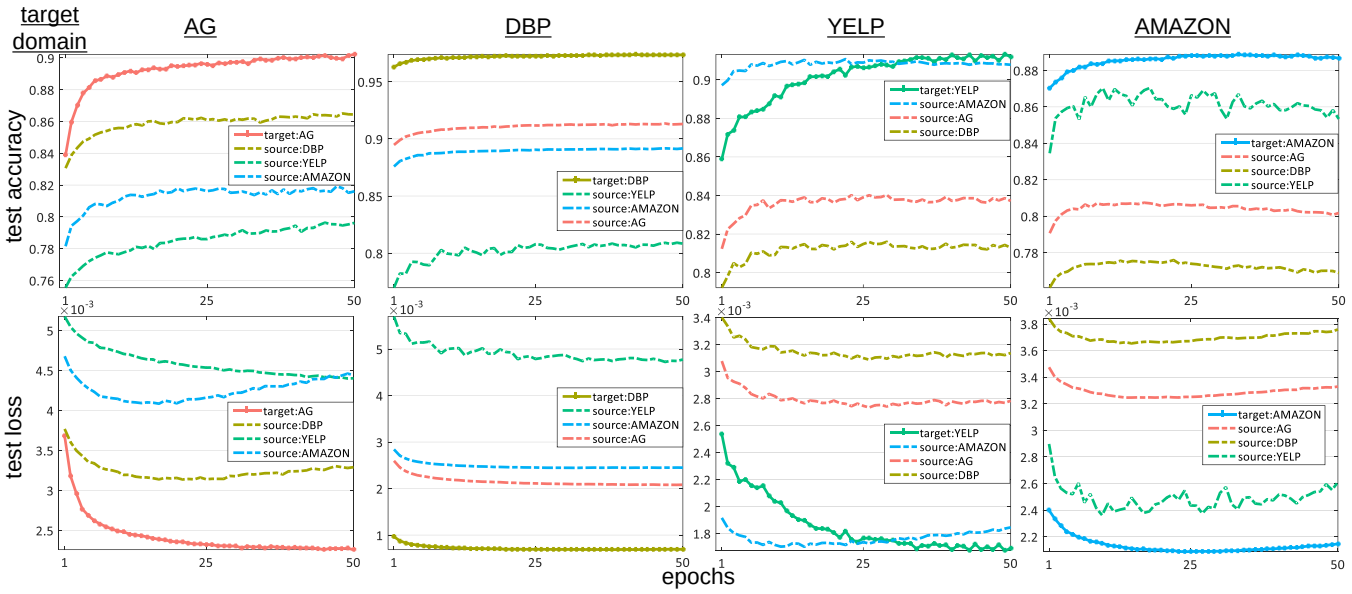

Figure 4: Results of the proposed method on real-world datasets (best viewed in color). Each plot in the top row: test accuracy of the model trained with the original data (solid lines) and those of the models trained with the other data sources and transferred (dotted lines). Each plot in the bottom row: test loss. For YELP and AMAZ, transfer learning was most effective, given that they have the highest level of structural similarity than the other cases (see Table 1).

convolutional networks. In *Computer Graphics Forum*, volume 34, 13–23. Wiley Online Library.

Bruna, J.; Zaremba, W.; Szlam, A.; and LeCun, Y. 2013. Spectral networks and locally connected networks on graphs. *arXiv preprint arXiv:1312.6203*.

Chung, F. R. 1997. *Spectral graph theory*, volume 92. American Mathematical Soc.

Deo, N. 2016. *Graph theory with applications to engineering and computer science*. Courier Dover Publications.

Duvenaud, D. K.; Maclaurin, D.; Iparraguirre, J.; Bombarell, R.; Hirzel, T.; Aspuru-Guzik, A.; and Adams, R. P. 2015. Convolutional networks on graphs for learning molecular fingerprints. In *Advances in Neural Information Processing Systems*, 2215–2223.

Henaff, M.; Bruna, J.; and LeCun, Y. 2015. Deep convolutional networks on graph-structured data. *arXiv preprint arXiv:1506.05163*.

Jain, A.; Zamir, A. R.; Savarese, S.; and Saxena, A. 2015. Structural-rnn: Deep learning on spatio-temporal graphs. *arXiv preprint arXiv:1511.05298*.

Koutra, D.; Vogelstein, J. T.; and Faloutsos, C. 2013. Deltacon: A principled massive-graph similarity function. SIAM.

Krizhevsky, A.; Sutskever, I.; and Hinton, G. E. 2012. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, 1097–1105.

LeCun, Y.; Bengio, Y.; and Hinton, G. 2015. Deep learning. *Nature* 521(7553):436–444.

Lee, J.; Kim, G.; and Yoon, S. 2015. Measuring large-scale dynamic graph similarity by ricom: Rwr with inter-graph compression. In *Data Mining (ICDM), 2015 IEEE International Conference on*, 829–834.

Li, Y.; Tarlow, D.; Brockschmidt, M.; and Zemel, R. 2015. Gated graph sequence neural networks. *arXiv preprint arXiv:1511.05493*.

Mohar, B. 1997. *Some applications of Laplace eigenvalues of graphs*. Springer.

Mou, L.; Li, G.; Zhang, L.; Wang, T.; and Jin, Z. 2016. Convolutional neural networks over tree structures for programming language processing. In *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence*.

Pan, S. J., and Yang, Q. 2010. A survey on transfer learning. *IEEE Transactions on knowledge and data engineering* 22(10):1345–1359.

Rustamov, R., and Guibas, L. J. 2013. Wavelets on graphs via deep learning. In *Advances in Neural Information Processing Systems*, 998–1006.

Schmidhuber, J. 2015. Deep learning in neural networks: An overview. *Neural Networks* 61:85–117.

Shuman, D. I.; Narang, S. K.; Frossard, P.; Ortega, A.; and Vandergheynst, P. 2013. The emerging field of signal processing on graphs: Extending high-dimensional data analysis to networks and other irregular domains. *Signal Processing Magazine, IEEE* 30(3):83–98.

Shuman, D. I.; Ricaud, B.; and Vandergheynst, P. 2016. Vertex-frequency analysis on graphs. *Applied and Computational Harmonic Analysis* 40(2):260–291.

Silver, D.; Huang, A.; Maddison, C. J.; Guez, A.; Sifre, L.; Van Den Driessche, G.; Schrittwieser, J.; Antonoglou, I.; Panneershelvam, V.; Lanctot, M.; et al. 2016. Mastering the game of go with deep neural networks and tree search. *Nature* 529(7587):484–489.

Sonawane, S., and Kulkarni, P. 2014. Graph based representation and analysis of text document: A survey of techniques. *International Journal of Computer Applications* 96(19).

Sutskever, I.; Vinyals, O.; and Le, Q. V. 2014. Sequence to sequence learning with neural networks. In *Advances in neural information processing systems*, 3104–3112.

Tong, H.; Faloutsos, C.; and Pan, J.-Y. 2006. Fast random walk with restart and its applications.

Zhang, X.; Zhao, J.; and LeCun, Y. 2015. Character-level convolutional networks for text classification. In *Advances in Neural Information Processing Systems*, 649–657.