Isomorphic Neural Network for Graph Representation Learning and Classification

Lin Meng
IFM Lab
Florida State University
lin@ifmlab.org

Jiawei Zhang
IFM Lab
Florida State University
jiawei@ifmlab.org

Abstract

Deep learning models have achieved huge success in numerous fields, such as computer vision and natural language processing. However, unlike such fields, it is hard to apply traditional deep learning models on the graph data due to the 'node-orderless' property. Normally, adjacent matrices will cast an artificial and random node-order on the graphs, which renders the performance of deep models extremely erratic, and the representations learned by such models lack clear interpretability. To eliminate the unnecessary node-order constraint, we propose a novel model named **Iso**morphic **N**eural **N**etwork (ISONN), which learns the graph representation by extracting its isomorphic features via the graph matching between the input graph and the learned subgraph templates. We test the proposed model with experiments on three real-world datasets. The results show the effectiveness of ISONN. A full version of this paper is available at [13]

1 Introduction

The graph structure is attracting increasing interests because of its great representation power on various types of data. Researchers have done many analyses based on different types of graphs, such as social networks, brain networks and biological networks. In this paper, we will focus on the binary graph classification problem, which has extensive applications in the real world. For example, one may wish to identify the social community categories according to the users' social interactions [4], distinguish the brain states of patients via their brain networks [16], and classify the functions of proteins in a biological interaction network [7].

However, we should notice that when we deal with the graph-structured data, different node-orders will result in very different adjacent matrix representations for most existing deep models (*i.e.*, SDBN [16], CNN [10]) which take the adjacent matrices as input. The different graph matrix representations brought by the node-order differences may render the learning performance of the existing models to be extremely erratic and not robust. Moreover, although the other models like MPNN [5] and GCN [8] learn implicit structural features, the explict structural information cannot be maintained for further research. Meanwhile, the subgraph mining methods based on BFS or DFS require high cost [17, 18]. Therefore, how to explicitly learn useful subgraph patterns automatically need to be solved. Lastly, representing graphs in the vector space is an important task since it facilitates the storage, parallelism and the usage of machine learning models for the graph data. Extensive works have been done on node representations [6, 12, 11, 7], whereas learning the representation of the whole graph with clear interpretability is still an open problem requiring more explorations.

In this paper, we propose a novel model, namely **Iso**morphic **N**eural **N**etwork (ISONN), to address the aforementioned challenges in the graph representation learning and classification problem. ISONN is composed of two components: the graph isomorphic feature extraction component and the classification component, aiming at learning isomorphic features and classifying graph instances, respectively. In the graph isomorphic feature extraction component, ISONN automatically learns a group of subgraph templates of useful patterns from the input graph. ISONN makes use of a set of permutation matrices, which act as the node isomorphism mappings between the templates and the input graph. With the potential isomorphic features learned by all the permutation matrices and the templates,

33rd Conference on Neural Information Processing Systems (NeurIPS 2019), Vancouver, Canada.

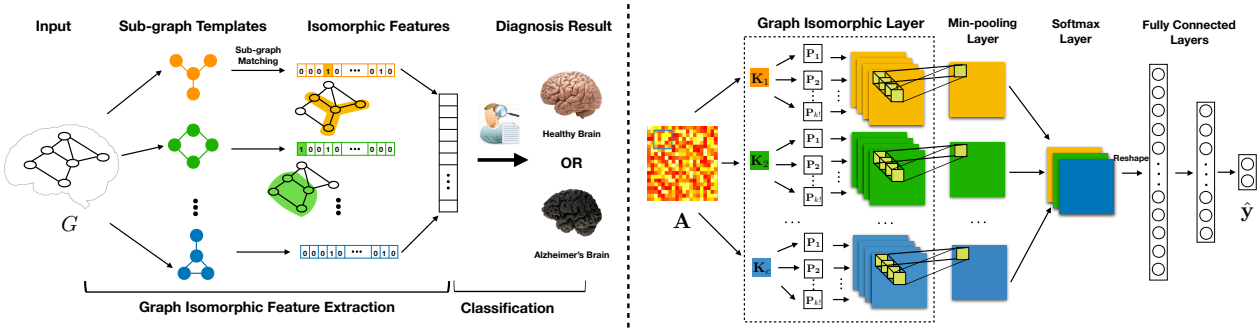


Figure 1: IsoNN Framework Architecture. (The left subplot provides the outline of the proposed framework, including the *graph isomorphic feature extraction* component and the *classification* component respectively. The right subplot illustrates the detailed architecture of the proposed framework, where the *graph isomorphic features* are extracted with the *graph isomorphic layer*, *min-pooling layer* and *softmax layer*, and the graphs are classified with three fully-connected layers.)

IsoNN adopts one min-pooling layer to find the best node permutation for each template and one softmax layer to differentiate all subgraph features learned by different kernels, respectively. Such features learned by different kernels will be fused together and fed as the input for the classification component. IsoNN further adopts three fully-connected layers as the classification component to project the graph instances to their labels.

2 Proposed Method

The overall architecture of IsoNN is shown in Figure 1. The IsoNN framework includes two main components: graph isomorphic feature extraction component and classification component. The graph isomorphic feature extraction component includes a graph isomorphic layer, a min-pooling layer as well as a softmax layer and the classification component is composed by three fully-connected layers. They will be discussed in detail in the following subsections.

2.1 Graph Isomorphic Feature Extraction Component

Graph isomorphic feature extraction component targets at learning the graph features. To achieve that objective, IsoNN adopts an automatic feature extraction strategy for graph representation learning. In IsoNN, one graph isomorphic feature extraction component involves three layers: the graph isomorphic layer, the min-pooling layer and the softmax layer. In addition, we can further construct a deep graph isomorphic neural network by stacking multiple such components on top of each other.

2.1.1 Graph Isomorphic Layer

Graph isomorphic layer is the first effective layer in deep learning that handles the node-order restriction in graph representations. Assume we have a graph $G = \{\mathcal{V}, \mathcal{E}\}$, and its adjacency matrix to be $\mathbf{A} \in \mathbb{R}^{|\mathcal{V}|\times|\mathcal{V}|}$. In order to find the existence of specific subgraph patterns in the input graph, IsoNN matches the input graph with a set of subgraph templates. Each template is denoted as a kernel variable $\mathbf{K}_i \in \mathbb{R}^{k\times k}, \forall i \in \{1, 2, \cdots, c\}$. Here, k denotes the node number in subgraphs and c is the channel number (i.e., total template count). Meanwhile, to match one template with regions in the input graph (i.e., sub-matrices in $\mathbf{A}$), we use a set of permutation matrices, which map both rows and columns of the kernel matrix to the subgraphs effectively. The permutation matrix can be represented as $\mathbf{P} \in \{0, 1\}^{k\times k}$ that shares the same dimension with the kernel matrix. Therefore, given a kernel matrix $\mathbf{K}_i$ and a sub-matrix $\mathbf{M}_{(s,t)} \in \mathbb{R}^{k\times k}$ in $\mathbf{A}$ (i.e., a region in the input graph G and $s, t \in \{1, 2, \cdots, (|\mathcal{V}| - k + 1)\}$ denotes a starting index pair in $\mathbf{A}$), there may exist $k!$ different such permutation matrices. The optimal should be the matrix $\mathbf{P}^*$ that minimizes the following term.

$$\mathbf{P}^* = \arg\min_{\mathbf{P}\in\mathcal{P}} \left\|\mathbf{P}\mathbf{K}_i\mathbf{P}^\top - \mathbf{M}_{(s,t)}\right\|_F^2, \tag{1}$$

where $\mathcal{P} = \{\mathbf{P}_1, \mathbf{P}_2, \cdots, \mathbf{P}_{k!}\}$ covers all the potential permutation matrices. Formally, the isomorphic feature extracted based on the kernel $\mathbf{K}_i$ for the regional sub-matrix $\mathbf{M}_{(s,t)}$ in $\mathbf{A}$ can be represented as

$$z_{i,(s,t)} = \left\|\mathbf{P}^*\mathbf{K}_i(\mathbf{P}^*)^\top - \mathbf{M}_{(s,t)}\right\|_F^2 = \min\{\left\|\mathbf{P}\mathbf{K}_i\mathbf{P}^\top - \mathbf{M}_{(s,t)}\right\|_F^2\}_{\mathbf{P}\in\mathcal{P}} \tag{2}$$

$$= \min(\bar{\mathbf{z}}_{i,(s,t)}(1:k!)), \tag{3}$$

where vector $\bar{\mathbf{z}}_{i,(s,t)} \in \mathbb{R}^{k!}$ contains entry $\bar{\mathbf{z}}_{i,(s,t)}(j) = \left\|\mathbf{P}_j\mathbf{K}_i\mathbf{P}_j^\top - \mathbf{M}_{(s,t)}\right\|_F^2, \forall j \in \{1, 2, \cdots, k!\}$ denoting the isomorphic features computed by the j-th permutation matrix $\mathbf{P}_j \in \mathcal{P}$.

As indicated by the Figure 1, IsoNN computes the final isomorphic features for the kernel $\mathbf{K}_i$ via two steps: (1) computing all the potential isomorphic features via different permutation matrices with

the graph isomorphic layer, and (2) identifying and fusing the optimal features with the min-pooling layer and softmax layer to be introduced as follows. By shifting one kernel matrix $\mathbf{K}_i$ on regional sub-matrices, IsoNN extracts the isomorphic features on the matrix $\mathbf{A}$, which can be denoted as a 3-way tensor $\bar{\mathcal{Z}}_i \in \mathbb{R}^{k! \times (|\mathcal{V}|-k+1) \times (|\mathcal{V}|-k+1)}$, where $\bar{\mathcal{Z}}_i(1:k!, s, t) = \bar{\mathbf{z}}_{i,(s,t)}(1:k!)$. In a similar way, we can also compute the isomorphic feature tensors based on the other kernels, which can be denoted as $\bar{\mathcal{Z}}_1, \bar{\mathcal{Z}}_2, \cdots, \bar{\mathcal{Z}}_c$ respectively.

2.1.2 Min-pooling Layer

Given the tensor $\bar{\mathcal{Z}}_i$ computed by $\mathbf{K}_i$ in the graph isomorphic layer, IsoNN will identify the optimal permutation matrices via the min-pooling layer. Formally, we can represent results of the optimal permutation selection with $\bar{\mathcal{Z}}_i$ as matrix $\mathbf{Z}_i$:

$$\mathbf{Z}_i(s,t) = \min\{\bar{\mathcal{Z}}_i(1:k!, s, t)\}. \tag{4}$$

The min-pooling layer learns the optimal matrix $\mathbf{Z}_i$ for kernel $\mathbf{K}_i$ along the first dimension (i.e., the dimension indexed by different permutation matrices), which can effectively identify the isomorphic features created by the optimal permutation matrices. For the remaining kernel matrices, we can also achieve their corresponding graph isomorphic feature matrices as $\mathbf{Z}_1, \mathbf{Z}_2, \cdots, \mathbf{Z}_c$ respectively.

2.1.3 Softmax Layer

Based on the above descriptions, a perfect matching between the subgraph templates with the input graph will lead to a very small isomorphic feature, e.g., a value approaching to 0. If we feed the small features into the classification component, the useful information will vanish and the relative useless information (*i.e.*, features learned by the subgraphs dismatch the kernels) dominates the learning feature vector in the end. Meanwhile, the feature values computed in Equation (4) can also be in different scales for different kernels. To effectively normalize these features, we propose to apply the softmax function to matrices $\mathbf{Z}_1, \mathbf{Z}_2, \cdots, \mathbf{Z}_c$ across all c kernels. Compared with the raw features, e.g., $\mathbf{Z}_i$, softmax as a non-linear mapping can also effectively highlight the useful features in $\mathbf{Z}_i$ by rescaling them to relatively larger values especially compared with the useless ones. Formally, we can represent the fused graph isomorphic features after rescaling by all the kernels as a 3-way tensor $\mathcal{Q}$, where slices along first dimension can be denoted as:

$$\mathcal{Q}(i,:,:) = \hat{\mathbf{Z}}_i \text{ , where } \hat{\mathbf{Z}}_i = softmax(-\mathbf{Z}_i), \ \ \forall i \in \{1, \ldots, c\}. \tag{5}$$

2.2 Classification Component

After the isomorphic feature tensor $\mathcal{Q}$ is obtained, we feed it into a classification component. Let $\mathbf{q}$ denote the flattened vector representation of feature tensor $\mathcal{Q}$, and we pass it to three fully-connected layers to get the predicted label vector $\hat{\mathbf{y}}$. For the graph binary classification, suppose we have the ground truth $\mathbf{y} = (y_1^g, y_2^g)$ and the predicted label vector $\hat{\mathbf{y}}^g = (\hat{y}_1^g, \hat{y}_2^g)$ for the sample g from the training batch set $\mathcal{B}$. We use cross-entropy as the loss function in IsoNN. Formally, the fully-connected (FC) layers and the objective function can be represented as follows respectively:

$$\text{FC Layers: } \begin{cases} \mathbf{d}_1 = \sigma(\mathbf{W}_1\mathbf{q} + \mathbf{b}_1), \\ \mathbf{d}_2 = \sigma(\mathbf{W}_2\mathbf{d}_1 + \mathbf{b}_2), \\ \hat{\mathbf{y}} = \sigma(\mathbf{W}_3\mathbf{d}_2 + \mathbf{b}_3), \end{cases} \quad \text{Objective Function: } \mathcal{L} = -\sum_{g \in \mathcal{B}} \sum_{j=1}^{2} y_j^g \log \hat{y}_j^g, \tag{6}$$

where $\mathbf{W}_i$ and $\mathbf{b}_i$ represent the weights and biases in i-th layer respectively for $i \in \{1, 2, 3\}$. The σ denotes the adopted the relu activation function. To train the proposed model, we adopt the back propagation algorithm to learn both the subgraph templates and the other involved variables.

3 Experiments

To evaluate the performance of IsoNN, we use accuracy and F1 as the evaluation metrics and discuss the experimental results for all comparison methods. The datasets used in the experiments are all the benchmark [1, 3, 14, 19, 2] for the brain graph classification for patient brain disease early diagnosis.

Reproducibility: Both the datasets and source code used in this paper can be accessed via link[1].

3.1 Comparison Methods

- **IsoNN**: The proposed method IsoNN uses a set of template variables as well as the permutation matrices to extract the isomorphic features and feed these features to the classification component.

[1] https://github.com/linmengsysu/IsoNN

Table 1: Classification Results of the Comparison Methods.

Dataset	Metric	Methods								
		Freq	Conf	Ratio	Gtest	HSIC	AE	CNN	SDBN	IsoNN
HIV-fMRI-77	Accuracy	54.3	58.6	54.3	50.0	58.7	46.9	59.3	66.5	**73.4**
	F1	58.2	64.2	62.0	52.5	59.5	35.5	66.3	66.7	**72.2**
HIV-DTI-77	Accuracy	64.6	52.4	59.3	59.3	49.8	62.4	54.3	65.9	**67.5**
	F1	63.9	46.1	57.9	58.5	58.3	0.0	55.7	65.6	**68.3**
BP-fMRI-97	Accuracy	56.8	50.8	54.2	55.2	54.9	53.6	54.6	64.8	**64.9**
	F1	57.6	49.1	53.7	53.9	55.8	69.5	52.8	63.7	**69.7**

- **Freq**: The method uses the top-k frequent subgraphs as its features. This is also an unsupervised feature selection method based on frequency.
- **Conf, Ratio, Gtest, HISC**: These methods are supervised subgraph selection [9] based on confidence, frequency ratio, G-test score, and HISC respectively. The top-*k* discriminative subgraph features are selected in terms of different discrimination criteria.
- **AE**: We use the autoencoder model [15] to get the features of graphs without label information. It is an unsupervised learning method, which learns the latent representations of all connections in the brain graphs without considering the structural information.
- **CNN**: It is the convolutional model [10] learns the structural information within small regions of the whole graph. We adopt one convolution layer and two fully-connected layers to extract features and one fully-connected layer to be the classification module.
- **SDBN**: A model proposed in [16], which reorders the nodes in the graph first and then feeds the reordered graph into an augmented CNN. In this way, it not only learns the structural information but also tries to minimize the effect of the order constraint.

3.2 Experimental Results

In this section, we investigate the effectiveness of the learned subgraph-based graph feature representations for brain graphs. We adopt one isomorphic layer where the kernel size $k = 2$ and channel number $c = 3$ for HIV-fMRI, one isomorphic layer with $k = 4, c = 2$ and $k = 3, c = 1$ for the HIV-DTI and BP-fMRI, respectively. The results are shown in Table 1. From that table, we can observe that IsoNN outperforms all other baseline methods on these three datasets. Compared with the time-consuming and resource-consuming subgraph mining based methods that search in a potentially exponential space, the proposed method achieves a better performance without searching for all possible subgraphs manually. In addition, the AE has the worst performance among all comparison methods. This is because the features learned from AE do not contain any structural information. For HIV-DTI, AE gets 0 in F1. This is because the dataset contains too many zeros, which makes the AE learns trivial features. CNN performs better than AE but worse than those subgraph mining methods in most cases. The reason can be that it learns some structural information but fails to learn representative structural patterns. Comparing IsoNN with AE, IsoNN achieves better results. This means the structural information is more important than only connectivity information for the classification problem. If compared with CNN, the results also show the contribution of breaking the node-order in learning the subgraph templates. Similar to SDBN, IsoNN also finds the features from subgraphs, but IsoNN gets better performance with more concise architecture. Due to the limited pages, more detailed information about the experimental setups, analyses about the model convergence, parameter sensitivity are also provided in the supplementary material submitted along with this paper.

4 Conclusion

In this paper, we proposed a novel graph neural network named IsoNN to solve the graph classification problem. IsoNN consists of two components: (1) isomorphic component, where a set of permutation matrices is used to break the randomness order posed by matrix representation for a bunch of templates and one min-pooling layer and one softmax layer are used to get the best isomorphic features, and (2) classification component, which contains three fully-connected layers. The experimental results on real-world datasets show the proposed method outperforms all comparison methods, which demonstrate the superiority of our proposed method.

5 Acknowledgment

This work is partially supported by NSF through grant IIS-1763365 and by FSU.

References

[1] B. Cao, X. Kong, J. Zhang, S. Y. Philip, and A. B. Ragin. Identifying hiv-induced subgraph patterns in brain networks with side information. *Brain informatics*, 2(4):211–223, 2015.

[2] B. Cao, X. Kong, J. Zhang, S. Y. Philip, and A. B. Ragin. Mining brain networks using multiple side views for neurological disorder identification. In *2015 IEEE International Conference on Data Mining*, pages 709–714. IEEE, 2015.

[3] B. Cao, L. Zhan, X. Kong, S. Y. Philip, N. Vizueta, L. L. Altshuler, and A. D. Leow. Identification of discriminative subgraph patterns in fmri brain networks in bipolar affective disorder. In *International Conference on Brain Informatics and Health*, pages 105–114. Springer, 2015.

[4] Q. Gao, F. Zhou, K. Zhang, G. Trajcevski, X. Luo, and F. Zhang. Identifying human mobility via trajectory embeddings. In *International Joint Conferences on Artificial Intelligence*, volume 17, pages 1689–1695, 2017.

[5] J. Gilmer, S. S. Schoenholz, P. F. Riley, O. Vinyals, and G. E. Dahl. Neural message passing for quantum chemistry. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pages 1263–1272. JMLR. org, 2017.

[6] A. Grover and J. Leskovec. node2vec: Scalable feature learning for networks. In *Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 855–864. ACM, 2016.

[7] W. Hamilton, Z. Ying, and J. Leskovec. Inductive representation learning on large graphs. In *Advances in Neural Information Processing Systems*, pages 1024–1034, 2017.

[8] T. N. Kipf and M. Welling. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907*, 2016.

[9] X. Kong, P. S. Yu, X. Wang, and A. B. Ragin. Discriminative feature selection for uncertain graph classification. In *Proceedings of the 2013 SIAM International Conference on Data Mining*, pages 82–93. SIAM, 2013.

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

[11] Y.-A. Lai, C.-C. Hsu, W. H. Chen, M.-Y. Yeh, and S.-D. Lin. Prune: Preserving proximity and global ranking for network embedding. In *Advances in neural information processing systems*, pages 5257–5266, 2017.

[12] Y. Lin, Z. Liu, M. Sun, Y. Liu, and X. Zhu. Learning entity and relation embeddings for knowledge graph completion. In *Twenty-ninth AAAI conference on artificial intelligence*, 2015.

[13] L. Meng and J. Zhang. Isonn: Isomorphic neural network for graph representation learning and classification. *arXiv preprint arXiv:1907.09495*, 2019.

[14] A. B. Ragin, H. Du, R. Ochs, Y. Wu, C. L. Sammet, A. Shoukry, and L. G. Epstein. Structural brain alterations can be detected early in hiv infection. *Neurology*, 79(24):2328–2334, 2012.

[15] P. Vincent, H. Larochelle, I. Lajoie, Y. Bengio, and P.-A. Manzagol. Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion. *Journal of machine learning research*, 11(Dec):3371–3408, 2010.

[16] S. Wang, L. He, B. Cao, C.-T. Lu, P. S. Yu, and A. B. Ragin. Structural deep brain network mining. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 475–484. ACM, 2017.

[17] X. Yan, H. Cheng, J. Han, and P. S. Yu. Mining significant graph patterns by leap search. In *Proceedings of the 2008 ACM SIGMOD international conference on Management of data*, pages 433–444. ACM, 2008.

[18] X. Yan and J. Han. gspan: Graph-based substructure pattern mining. In *2002 IEEE International Conference on Data Mining, 2002. Proceedings.*, pages 721–724. IEEE, 2002.

[19] J. Zhang, B. Cao, S. Xie, C.-T. Lu, P. S. Yu, and A. B. Ragin. Identifying connectivity patterns for brain diseases via multi-side-view guided deep architectures. In *Proceedings of the 2016 SIAM International Conference on Data Mining*, pages 36–44. SIAM, 2016.

6 Supplementary Material

6.1 Experimental Setup

In our experiments, to make the results more reliable, we partition the datasets into 3 folds and then set the ratio of train/test according to $2:1$, where two folds are treated as the training data and the remaining one is the testing data. To evaluate the quality of the learned brain network features, we feed the learned features to a softmax classifier for the subgraph mining methods. For the subgraph mining methods, the thresholds for three datasets are 0.9, 0.3, 0.5, respectively. We select top-100 features for classification as in [16]. For Auto-encoder, we apply the two-layer encoder and two-layer decoder. For the CNN, we apply the one convolutional layer with the size $5 \times 5 \times 50$, a max-pooling layer with kernel size 2×2, one gating relu layer as activation layer and two fully-connected layers which contain 1024 and 128 neurons, respectively. For the SDBN, we set the architecture as follows: we use two layers of "convolution layer + max pooling layer + activation layer " and concatenate a fully connected layer with 100 neurons as well as an activation layer, where the parameters are the same as those in CNN. We also set the dropout rate in SDBN being 0.5 to avoid overfitting. In the experiments, we set the kernel size k in the isomorphic layer for three datasets as 2, 4, 3, respectively, and then set the neuron number of each fully-connected layer as 1024, 128 and 2, respectively. In this experiment, we adopt Adam optimizer and the set the learning rate $\eta = 0.001$, and then we report the average results on balanced datasets.

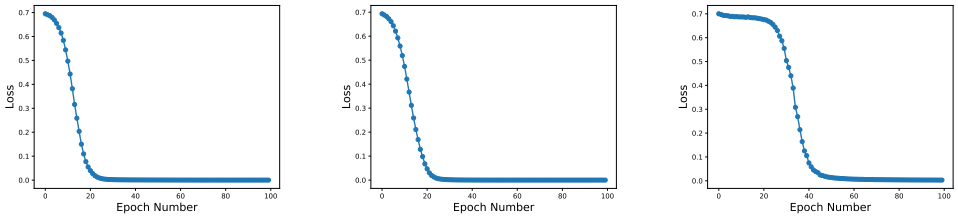
6.2 Convergence Analysis and Parameter Analysis

To further study the proposed method, we will discuss the model convergence and the effects of different kernel size and channel number in IsoNN.

- **Convergence Analysis**: The Figure 2 shows the convergence trend of IsoNN on three datasets, where the x-axis denotes the epoch number and the y-axis is the training loss, respectively. From these three sub-figures, we can know that the proposed method can achieve a stable optimal solution within 50 iterations, which also illustrates our method would converge relatively fast.
- **Kernel Size**: We show the effectiveness of different k in Figure 3. Based on the previous statement, parameter k can affect the final results since it controls the size of learned subgraph templates. To investigate the best kernel size for each dataset, we fix the channel number $c = 1$. As Figure 3 shows, different datasets have different appropriate kernel sizes. The best kernel sizes are 2, 4, 3 for the three datasets respectively.
- **Channel Number**: We also study the effectiveness of multiple channels (i.e., multiple templates in one layer). To discuss how the channel number influences the results, we choose the best kernel size for each dataset (i.e., 2, 4, 3, respectively). From all sub-figures in Figure 4, we can see that the differences among the different channel numbers by using only one isomorphic layer. As shown in Figure 4, IsoNN achieves the best results by $c = 3, 2, 1$, respectively, which means the increase of the channel number can improve the performance, but more channels does not necessarily lead to better results. The reason could be the more templates we use, the more complex our model would be. With such a complex model, it is easy to learn an overfitting model on train data, especially when the dataset is quite small. Thus, increasing the channel number can improve the performance but the effectiveness will still depend on the quality and the quantity of the dataset.

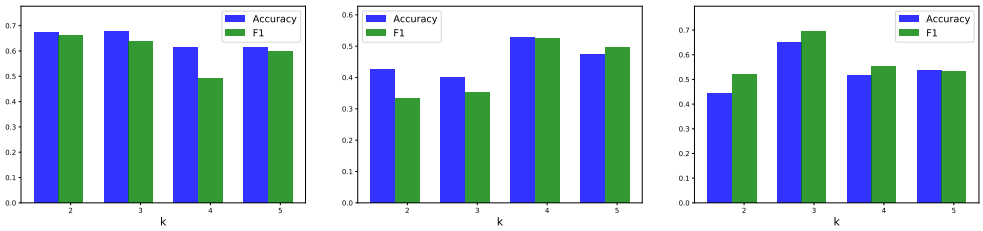
6.3 Time Complexity Study

To study the efficiency of IsoNN, we collect the actual running time with a fixed epoch number (i.e., 50), which is shown on Figure 5. In both Figures 5(a) and 5(b), the x-axis denotes its value for k or c and the y-axis denotes the time cost with different parameters. From Figure 5(a), three lines show the same pattern. When the k increases, the time cost grows exponentially. This pattern can be directly explained by the size of the permutation matrix set. When we increase the kernel size by one, the number of corresponding permutation matrices grows exponentially. While changing c, shown in Figure 5(b), it is easy to observe that those three curves are basically linear with different slopes. This is also natural since whenever we add one channel, we only need to add a constant number of the permutation matrices.



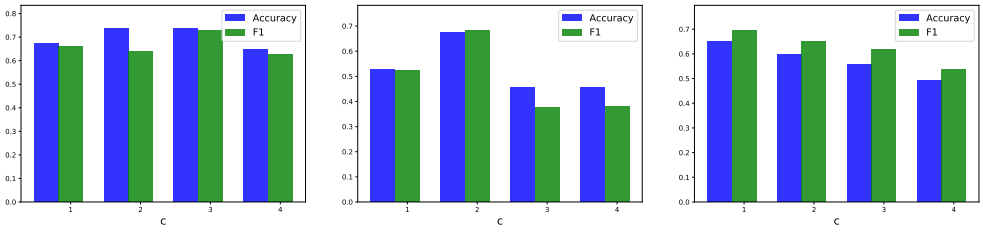
(a) HIV-fMRI (b) HIV-DTI (c) BP-fMRI

Figure 2: Convergence Analysis



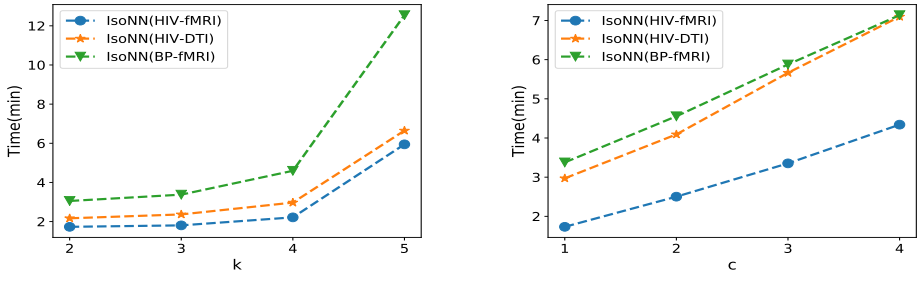
(a) HIV-fMRI (b) HIV-DTI (c) BP-fMRI

Figure 3: Effectiveness of Different k



(a) HIV-fMRI (b) HIV-DTI (c) BP-fMRI

Figure 4: Effectiveness of Different c



(a) Different k (b) Different c

Figure 5: Time Complexity Study