MoCaNet: Motif Graph Capsule Neural Network

Abstract

In this paper, we propose a novel structure-driven approach, named **Mo**tif graph **Ca**psule neural **Net**work (MoCaNet), for graph embedding learning by exploring structure information from various levels. In specific, given an input graph, MoCaNet first utilizes a set of motifs to generate motif graph. Then the motif graph is fed to a specially designed motif graph convolution network (m-GCN) for low-level structure extraction, which is insufficiently explored by existing methods works directly on the original input graph. Afterwards, we introduce a learnable condensation operation to further mine mid-level structure from the low-level embedding obtained by m-GCN. In addition to learning mid-level structure, another benefit of condensation is to perform size normalization among different inputs. The final high-level structured embedding is captured using a capsule architecture. To effectively train MoCaNet, we propose a novel loss that helps leverage rich useful structure information during learning by focusing on core graph reconstruction. In our extensive experiments on 10 diverse datasets, MoCaNet consistently achieves better or comparable results in comparison to previous arts, evidencing the superiority of our structure exploration in graph embedding learning. Our implementation will be made available.

Introduction

Graph learning is one of the fundamental topics in representation learning and has a long list of applications including graph classification, link prediction, node classification, node clustering and so on. One of the core problems in graph learning is to develop a high-quality graph embedding that effectively captures structure in graph data. Recently, inspired by the success of deep convolutional neural network (CNN) (Krizhevsky, Sutskever, and Hinton 2012; LeCun et al. 1998), many graph convolution network (GCN) approaches (Bruna et al. 2014; Zhang et al. 2018; Xinyi and Chen 2019; Verma and Zhang 2018) have been proposed to learn robust node representation in graph. The basic procedure of these GCN methods is to aggregate neighbors' information for each node in different layers to develop graph embedding. Despite demonstrating promising results, the learned graph embedding by GCN may not suffice to preserve node/graph properties effective (*e.g.*, node position and connection) (Xinyi and Chen 2019). To deal with this problem, recent studies (Verma and Zhang 2018; Xinyi and Chen 2019) propose graph capsule network by combining GCN with capsule architecture (Sabour, Frosst, and Hinton 2017). Similar to GCN methods (Bruna et al. 2014; Zhang et al. 2018), these graph capsule networks start from neighbor node aggregation to learn embedding. The difference is that, an additional capsule network is applied to the learned embedding to capture high-level invariant structure information. As a consequence, better embedding is achieved in graph capsule network.

For embedding learning, it is important to fully exploit structure information from graph data. The aforementioned *node-driven* algorithms mainly leverage node aggregation for information extraction, which may leave rich *underlying* structure in graph data under-explored. In particular, the structure of connection patterns (*e.g.*, hub, triangle, star, square, etc.) is crucial for distinguishing different graph data (Rossi et al. 2018). The aforementioned GCN or graph capsule networks, however, do not pay sufficient attention to such rich pattern structures, which may result in sub-optimal embedding learning.

From the perspective of structure exploration, we introduce a novel *structure-driven* approach, named **Mo**tif graph **Ca**psule neural **Net**work (MoCaNet), for graph embedding learning. MoCaNet is inspired by the observation that motifs[1] can effectively represent the basic connection patterns (*e.g.*, hub, triangle, star and square) in a graph (Tsourakakis, Pachocki, and Mitzenmacher 2017; Rossi et al. 2018; Dareddy, Das, and Yang 2019). These patterns describe various structure relationships among nodes, and provide rich extra structure information for exploration. Thus motivated, MoCaNet utilizes a set of motifs to generate a motif graph for graph embedding, with each motif encoding certain type of local structure pattern. With the motif graph, a specially designed motif graph convolution network (m-GCN) is designed for learning *low-level* structure of a graph. To further mine structure information from low-level embedding obtained by m-GCN, we introduce a condensation operation that extracts *mid-level* structure by clus-

[1] A motif is a sub-graph that is frequently repeated in a graph (Milo et al. 2002).

tering nodes in low-level graph embedding into condensed ones. Meanwhile, condensation also solves the size variation problem of different inputs by fixing the number of nodes in condensed graph embedding. Then, to capture higher-level structure, similar to (Verma and Zhang 2018; Xinyi and Chen 2019), we introduce the capsule network (Sabour, Frosst, and Hinton 2017) to extract *high-level* structured embedding. For effective training of MoCaNet, a novel core graph reconstruction loss is proposed. By integrating it with conventional margin loss, we can not only exploit richer information for high-level embedding learning, but also accelerate the training process for convergence.

Through explicit exploration of various information in different stages, our structure-driven MoCaNet learns better graph embedding than those node-driven methods. In our thorough experiments on 10 diverse datasets, MoCaNet consistently achieves better or comparable results in comparison with state-of-the-arts, including recent GCN and graph capsule network approaches.

We summarize the main contributions of our work as follows:

- We propose the *structure-driven* MoCaNet for high-quality graph embedding learning by fully exploring structure from various levels.
- We introduce a novel core graph reconstruction loss that encourages more information for embedding learning and accelerates training of MoCaNet.
- We conduct extensive experiments on 10 biology and social network datasets where the proposed MoCaNet performs favorably against existing state-of-the-arts.

Related work

Graph representation learning is a fundamental problem and has attracted extensive attention. Here we would like to discuss four lines of works that are most relevant to ours.

Motif in graph learning. Motif has been extensively explored in graph learning owing to its ability to provide rich structure information. The method of (Benson, Gleich, and Leskovec 2016) leverages motif to perform clustering on nodes for understanding complex graph and network. The approach of (Sankar, Zhang, and Chang 2017) combines CNN with motif to model key properties of local connectivity for high-order structure caption in graph. Despite promising result, it requires a symmetric Laplacian matrix for orthogonal eigen-decomposition, limiting its application to directed graph. To solve this problem, the work of (Monti, Otness, and Bronstein 2018) explores local graph motifs that enable representation learning of directed graph. In (Tsourakakis, Pachocki, and Mitzenmacher 2017), the triangle motif is adopted for node clustering. The algorithm of (Rossi et al. 2018) utilizes motif to generate the motif weighted graph for embedding learning. In (Dareddy, Das, and Yang 2019), the motif is employed to learn high-quality node embedding.

Graph neural network (GNN). GNN has demonstrated state-of-the-art performance in many graph tasks. Early GNN approaches (Gori, Monfardini, and Scarselli 2005; Scarselli et al. 2008) are mainly focused on recursive neural network models. Inspired by the success of CNN in vision tasks, the seminal work of (Bruna et al. 2014) proposes several graph convolution operations for spectral analysis. Despite excellent performance, this approach is complex due to the calculation of eigen-decomposition of graph Laplacian matrix. To handle this issue, the methods of (Henaff, Bruna, and LeCun 2015; Kipf and Welling 2017) apply the Chebyshev polynomials to efficiently approximate the graph Laplacian. The algorithm of (Defferrard, Bresson, and Vandergheynst 2016) proposes to optimize GCN using fast localized spectral filters. The method of (Ying et al. 2018) introduces a differentiable pooling layer to learn hierarchical graph representation. In (Zhang et al. 2018), a sort pooling layer is presented to coarsen graphs by performing 1D convolution on sorted node for graph representation learning. In (Simonovsky and Komodakis 2017), the convolution filter weights are conditioned on edge labels and dynamically generated for different input graphs.

Capsule network. Recently capsule network has drawn increasing attention owing to its advantages in capturing high-level structure. The pioneer work (Hinton, Krizhevsky, and Wang 2011) introduces the capsule architecture to solve the pose-invariant problem in vision tasks. In specific, each capsule represents one certain visual entity, and there exists a belonging relationship between higher- and lower-level entities. The pose of a higher-level visual entity can be learned by aggregating lower-level ones belonging to it. Latter, the approach of (Sabour, Frosst, and Hinton 2017) introduces a framework of capsule neural network using vector to represent capsule and dynamic routing mechanism to aggregate lower-level capsules. In (Hinton, Sabour, and Frosst 2018), a capsule is defined by a matrix and a scale, indicating respectively the detailed properties and the probability of the presence.

Graph capsule network. Several recent studies combine GNN and capsule architecture for embedding learning and show promising results. The approach in (Mallea, Meltzer, and Bentley 2019) utilizes a contextual tensor to represent a graph by encoding the adjacent matrix and node features. After that, a convolution layer is adopted to extract features from the contextual tensor for developing the capsule network. In (Verma and Zhang 2018), node capsules are built upon the higher-order statistical moments. Then, graph convolution is utilized to generate higher-level node capsules for graph representation learning. In (Xinyi and Chen 2019), node embedding is first generated from GCN layers. The primary capsule is then built based on node embedding from different GCN layers. Transform matrix sharing and attention module are adopted to address the problem of graph size inequality. Dynamic routing is applied to generate final graph representation.

Our approach. We aim to learn high-quality embedding by exploring rich structure from graph data. Different from existing GCN methods that learn graph embedding in a node-driven way (Bruna et al. 2014; Ying et al. 2018; Defferrard, Bresson, and Vandergheynst 2016), our MoCaNet takes a structure-driven way to explore rich structure information using motifs. MoCaNet is also related to but signifi-

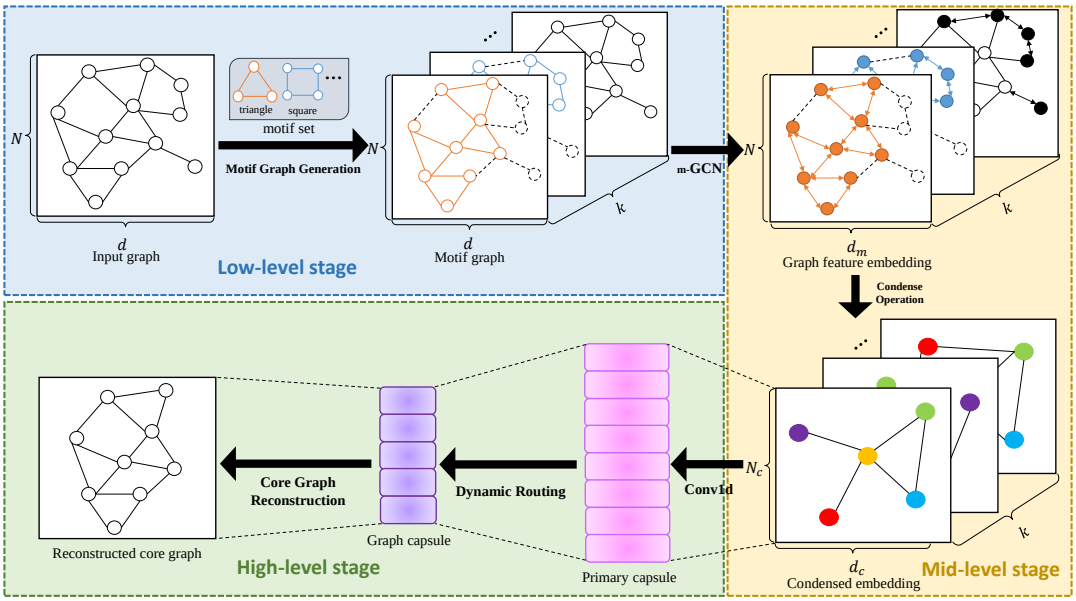


Figure 1: Framework of the proposed structure-driven MoCaNet by full exploration of various structure in a graph. Best viewed in color.

cantly different from previous graph capsule network methods (Verma and Zhang 2018; Xinyi and Chen 2019). In particular, it uses condensation operations and motifs to promote structural learning and avoids drastic information loss. In addition, a novel loss function is introduced for effective training.

Motif graph capsule neural network

Overview

The goal of MoCaNet is to explore structure from various levels in a graph for high-quality embedding learning. In specific, we first utilize a set of motifs to generate motif graph that provide rich additional structure information for exploration. Based on the motif graph, we employ m-GCN to extract the low-level structured graph embedding. After that, we use a learnable condensation operation to exploit mid-level structure from the low-level structure obtained by m-GCN. Meanwhile, the condensation operation helps avoid the problem of different sizes of graphs. To further capture high-level information from the mid-level condensed embedding, a capsule architecture with dynamic routing is adopted. Figure 1 illustrates the overall architecture of MoCaNet.

Motif graph generation

By definition, a graph can be represented by $G = (\mathbb{V}, \mathbb{E}, \boldsymbol{A}, \boldsymbol{F})$, where $\mathbb{V}$ is the node set of size N, $\mathbb{E}$ the edge set, $\boldsymbol{A} \in \{0,1\}^{N\times N}$ the adjacent matrix, and $\boldsymbol{F} \in \mathbb{R}^{N\times d}$ the node feature matrix of feature dimension d. A motif describes a featured sub-graph frequently repeated in the original graph:

Motif: A motif $M = (\mathbb{V}_M, \mathbb{E}_M)$ can be viewed as a featured isomorphic subgraph that consists of several nodes from the original graph, where the motif node set $\mathbb{V}_M \subseteq \mathbb{V}$ is a subset of the original node set, and the motif edge set $\mathbb{E}_M \subseteq \mathbb{E}$ consists of all the edges in $\mathbb{E}$ that have both endpoints in $\mathbb{V}_M$. In this paper, we use three types of motifs, including *triangle*, *square* and *normal edge*.

Then the **motif graph** $\widetilde{G}$ is defined as $\widetilde{G} = (\mathbb{V}, \mathbb{E}, \mathbb{B}, \boldsymbol{F}, \mathbb{M})$, where $\mathbb{M} = \{M_1, ..., M_k\}$ is the set of k motifs, and $\mathbb{B} = \{\boldsymbol{B}^1, ..., \boldsymbol{B}^k\}$ is the set of structural information matrix containing k different motif adjacent matrices. The motif graph is generated by building the motif adjacent matrices. For each node pair (m, n), $m \neq n$, ${\boldsymbol{B}^i}_{mn} = 1$ iff node m and node n participate in the i-th motif, where $\boldsymbol{B}^i \in \{0,1\}^{N\times N}$ is the motif adjacent matrix of the i-th motif. Different motif adjacent matrices contain different structural information because they are generated from different motifs.

Motif graph convolution network (m-GCN)

Given a graph G, GCN conducts graph convolution on each node and its neighbors and generate higher-level embedding. The procedure of basic GCN be written as:

$$\boldsymbol{Z}^{l+1} = \sigma(\boldsymbol{T}\boldsymbol{Z}^l\boldsymbol{W}^l) \tag{1}$$

where $\boldsymbol{Z}^l$ is the node embedding at l-th layer, $\boldsymbol{W}^l$ is a trainable filter, $\boldsymbol{T}$ is a structural information matrix to guide the information flow between different nodes and determine the aggregation in graph convolution. In particular, $\boldsymbol{T}$ is usually calculated by $\boldsymbol{T} = \widetilde{\boldsymbol{D}}^{-\frac{1}{2}}\widetilde{\boldsymbol{A}}\widetilde{\boldsymbol{D}}^{-\frac{1}{2}}$.

Considering the difference between motif graph and normal graph, we extend GCN to a new model named *motif graph convolution network* (m-GCN). The procedure of m-GCN can be written as:

$$\boldsymbol{Z}_t^{l+1} = \sigma(\boldsymbol{T}_t\boldsymbol{Z}_t^l\boldsymbol{W}_t^l) \tag{2}$$

where $\boldsymbol{T}_t = \text{concat}(\boldsymbol{B}^1, \cdots, \boldsymbol{B}^k) \in \{0,1\}^{N\times N\times k}$ is the structural information tensor used to guide the information flow between different nodes, $\boldsymbol{Z}_t^l \in \mathbb{R}^{N\times d_l\times k}$ is the node embedding at l-th layer and $\boldsymbol{Z}_t^0 = \text{concat}(\boldsymbol{F}, \cdots, \boldsymbol{F})$, $\boldsymbol{W}_t^l \in \mathbb{R}^{d_l\times d_{l+1}\times k}$ is a trainable parameter tensor. Graph feature embedding $Z_m \in \mathbb{R}^{N\times d_m\times k}$ is generated by the last layer of m-GCN. Then graph feature embedding can be fed into the condensation operator to generate middle-level structure.

Condensed embedding

As shown in Figure 2, the condensation operation is used to aggregate low-level structure (*i.e.*, graph feature embedding) and generate the middle-level structure (*i.e.*, condensed embedding). Inspired by (Ying et al. 2018), we implement the condensation operation by learning an assignment tensor. By definition, assignment tensor $\boldsymbol{S}_t \in \mathbb{R}^{N\times N_c\times k}$ contains k different assignment matrix corresponding to k different motifs. For each assignment matrix $\boldsymbol{S}_{ti} \in \mathbb{R}^{N\times N_c}$, N is the original graph size, N_c is the size of condensed graph. The assignment matrix describes the aggregation relationship between the low-level structure and the middle-level structure. With the assignment tensor, the condensation operator is performed by:

$$\boldsymbol{Z}_c = \boldsymbol{S}_t{}^T \boldsymbol{Z}_m \tag{3}$$

where $\boldsymbol{Z}_c \in \mathbb{R}^{N_c\times d_c\times k}$ is the condensed embedding, $\boldsymbol{S}$ is the assignment matrix, $\boldsymbol{Z}_m$ is the graph feature embedding. In this paper, we simply adopt a GNN layer to learn the assignment matrix:

$$\boldsymbol{S} = \text{softmax}(\text{GNN}(\boldsymbol{T}_t \boldsymbol{Z}_m \boldsymbol{W}_c)) \tag{4}$$

where $\boldsymbol{W}_c$ is a trainable parameter.

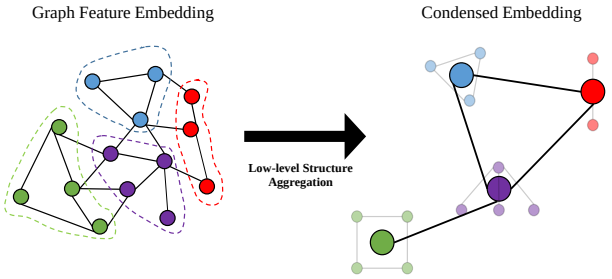


Figure 2: Framework of the proposed structure-driven MoCaNet by full exploration of various structure in a graph. Best viewed in color.

Capsule architecture

After extracting the condensed embedding, the capsule network is built to capture the high-level structure and generate the graph representation. We first construct the primary capsule using 1d convolution (Conv1d) layers, as shown in Figure 3. We use the Conv1d layers to combine different structural information in the condensed embedding and then build the primary capsule $P \in \mathbb{R}^{N_cO_c\times d_p}$ where O_c is the output channel of the Conv1d layers, and d_p is the dimension of primary capsule which is equal to the number of Conv1d layers.

Then, dynamic routing is employed to learn a graph-level representation, *i.e.*, the graph capsule $\boldsymbol{C} \in \mathbb{R}^{K\times d_G}$, where K is the number of classes, d_G is the dimension of graph capsule. In the graph capsule layer, the length of the k-th capsule represents the probability that the input graph belongs to the k-th class.

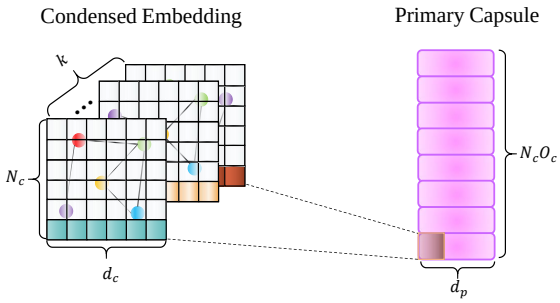


Figure 3: A 1d convolution layer is used to combine different structural information generated from different motifs. There k input channels corresponding to k motifs, and $O_e = 3$ output channels. The kernel size is d_c, same as the dimension of the condensed embedding.

After extracting the condensed embedding, the capsule network is built to capture the high-level structure and generate the graph representation. We first construct the primary capsule using 1d convolution (Conv1d) layers, as shown in Figure 3. We use the Conv1d layers to combine different structural information in the condensed embedding and then build the primary capsule $P \in \mathbb{R}^{N_cO_c\times d_p}$ where O_c is the output channel of the Conv1d layers, and d_p is the dimension of primary capsule which is equal to the number of Conv1d layers.

Then, dynamic routing is employed to learn a graph-level representation, *i.e.*, the graph capsule $\boldsymbol{C} \in \mathbb{R}^{K\times d_G}$, where K is the number of classes, d_G is the dimension of graph capsule. In the graph capsule layer, the length of the k-th capsule represents the probability that the input graph belongs to the k-th class.

Loss function

In this work, the loss $\mathcal{L}$ for training MoCaNet consists of two components, including the conventional margin loss $\mathcal{L}_c$ and a novel core graph reconstruction loss $\mathcal{L}_r$. The margin loss is used to measure the classification error, and the our core graph reconstruction loss serves as a regularization for accelerating training of MoCaNet. In addition, it can be used as a guidance to promote MoCaNet to capture additional graph structure information. $\mathcal{L}$ is mathematically computed as:

$$\mathcal{L} = \mathcal{L}_c + \mu\mathcal{L}_r \tag{5}$$

where μ controls the weight of reconstruction loss and is set to 0.0005 in our experiment. The margin loss $\mathcal{L}_c$ and core reconstruction loss $\mathcal{L}_r$ are described later.

Table 1: Detailed description of 10 diverse datasets used for experiments.

	Datasets	Graphs	Classes	Avg. Nodes	Avg. Edges
Biology	ENZYMES	600	6	32	63
	MUTAG	188	2	18	20
	PROTEINS	1113	2	39	73
	D&D	1178	2	284	716
	NCI1	4110	2	30	32
	PTC	344	2	26	29
Social Network	COLLAB	5000	3	74	4915
	IMDB-M	1500	3	13	132
	IMDB-B	1000	2	20	193
	REDDIT-M5K	4999	5	509	1190

Margin loss The graph capsule $\boldsymbol{C}$ is used to calculate the margin loss. We use the margin loss $\mathcal{L}_c$ as in (Sabour, Frosst, and Hinton 2017):

$$\begin{aligned}\mathcal{L}_c = \{&\sum_n T_n \max(0, m^+ - \|c_n\|)^2 \\ &+ \lambda(1 - T_n)\max(0, \|c_n\| - m^-)^2\}\end{aligned} \tag{6}$$

where $m^+ = 0.9$, $m^- = 0.1$, and $T_n = 1$ if the input graph belongs class *n*, otherwise $T_n = 0$, λ is used to stop the initial learning from shrinking the lengths of all graph capsules, and c_n is the *n*-th capsule in the graph capsule layer $\boldsymbol{C}$.

Core graph reconstruction loss Due to the various size of graph, it is hard to reconstruct the whole feature matrix or adjacent matrix. The approach of (Xinyi and Chen 2019) reconstructs the histogram of nodes that is independent of the graph size. However, the node histogram does not contain rich graph information and thus makes it hard for the model to capture structural information through reconstruction.

Addressing this issue, we design a new reconstruction loss called core graph reconstruction loss. Before introducing it in details, we first give the definition of the core graph $G_c = (\mathbb{V}_c, \mathbb{E}_c, \boldsymbol{A}_c)$, where $\mathbb{V}_c$ is the set of the selected most representative node and $\mathbb{E}_c$ denotes the edge set. For each pair of nodes i and j in $\mathbb{V}$, $(i, j) \in \mathbb{E}_c$ iff $(g(i), g(j)) \in \mathbb{E}$ where $g : \mathbb{V}_c \to \mathbb{V}$ is the mapping function that used to select the most representative nodes from the original graph. $\boldsymbol{A}_c$ is the adjacent matrix of core graph, and the node's representative of node i is defined as:

$$R_i = \sum_{m=1}^{k} \sum_{j=1}^{N} {\boldsymbol{B}^k}_{i,j} \tag{7}$$

Where R_i is the node's representative of node i, $\boldsymbol{B}^k$ is the motif adjacent matrix of the *k*-th motif. Node's representative measures this node's structural importance in the original graph. Due to the various graph size, we can not reconstruct the whole graph but a fixed-size reduced informative graph, namely core graph. To avoid drastic information loss, we select the nodes that are rich in structural information. So that the core graph is built based on the most representative nodes. Then we can use the adjacent matrix of core graph to calculate the reconstruction loss. Here we use an MLP to reconstruct the core graph adjacent matrix. The reconstructing loss $\mathcal{L}_r$ can guide the model to learn more graph structure information, which can be written as:

$$\mathcal{L}_r = \sum_{i,j} (\boldsymbol{A}_{ci,j} - \mathrm{MLP}(\boldsymbol{C})_{i,j})^2 \tag{8}$$

When reconstructing the core graph, except the capsule that belongs to the correct class, all other capsules will be masked.

Experiments

Implementation details

In MoCaNet, the number l of layers in m-GCN is chosen from $\{2, 3, 4\}$ according to the size of datasets. Accordingly, the dimension d of hidden layers in m-GCN is selected from $\{10, 15, 20\}$. The size N_s of the condensed embedding is selected from $\{15, 25, 75, 100\}$, and its dimension is chosen form $\{10, 15, 20\}$. As shown in Figure 1, we utilize two capsule layers for all datasets, and the dimension of capsule is chosen from $\{10, 15, 20\}$. In addition, the iteration of dynamic routing procedure is set to 3. The size of the core graph is chosen from $\{15, 25, 75, 100\}$. We utilize ADAM optimizer (Kingma and Ba 2014) for training, and the weight decay is set to $1e - 6$. The training epoch is selected from $\{50, 100, 250, 500\}$.

Datasets and compared methods

We conduct graph classification tasks using our MoCaNet on 10 diverse datasets. In specific, we utilize six biology datasets (*i.e.*, ENZYMES (Schomburg et al. 2004), MUTAG (Debnath et al. 1991), PROTEINS (Borgwardt et al. 2005), D&D (Dobson and Doig 2003), NCI1 (Wale, Watson, and Karypis 2008) and PTC (Toivonen et al. 2003)) and four social network datasets (*i.e.*, COLLAB (Yanardag and Vishwanathan 2015), IMDB-B, IMDB-M and RE-M5K). Table 1 describes the details of these datasets[2].

In order to to demonstrate the performance of MoCaNet, we compare our approaches with 14 algorithms. These

[2]These datasets can be downloaded at https://ls11-www.cs.tu-dortmund.de/staff/morris/graphkerneldatasets.

Table 2: Results and comparisons with state-of-the-art methods on biology datasets. The best two results are shown in red and blue fonts.

	Methods	ENZYMES	MUTAG	PROTEINS	D&D	NCI1	PTC
Graph Kernal	WL	52.22±1.26	82.05±0.36	74.68±0.49	79.78±0.36	82.19±0.18	57.97±0.49
	SP	40.10±1.50	85.79±2.51	75.07±0.54	>1day	73.00±0.24	58.24±2.44
	GK	32.70±1.20	81.58±2.11	71.67±0.55	78.45±0.26	62.49±0.27	57.26±1.41
	RW	24.16±1.64	79.17±2.07	74.22±0.42	>1day	>1day	57.85±1.30
	DGK	53.43±0.91	87.44±2.72	75.68±0.54	73.50±1.01	80.31±0.46	60.08±2.55
	MLG	61.81±0.99	84.21±2.61	76.34±0.72	78.18±2.56	81.75±0.24	63.26±1.48
Deep Learning	Graph2vec	-	83.15±9.25	73.30±2.05	-	73.22±1.81	-
	AWE	35.77±5.93	87.87±9.76	-	71.51±4.02	-	-
	PSCN	-	88.95±4.37	75.00±2.51	76.27±2.64	76.34±1.68	62.29±5.68
	DGCNN	51.00±7.29	85.83±1.66	75.54±0.94	79.37±0.94	74.44±0.47	58.59±2.47
	ECC	45.67	76.11	-	72.54	76.82	-
Graph Capsule	GCAPS-CNN	61.83±5.39	-	76.40±4.17	77.62±4.99	82.72±2.38	66.01±**5.91**
	CapsGNN	54.67±5.67	86.67±6.88	76.28±3.63	75.38±4.17	78.35±1.55	-
	TGCapsNN	27.0±8.45	88.9±5.49	74.10±3.24	77.90±2.49	65.9±1.07	53.37±1.63
	MoCaNet (Ours)	62.50±4.83	90.52±4.21	76.34±4.00	79.32±4.57	78.95±1.34	64.86±3.72

Table 3: Results and comparisons with state-of-the-art methods on social network datasets. The best two results are shown in red and blue fonts.

	Method	COLLAB	IMDB-M	IMDB-B	RE-M5K
Graph Keneral	WL	79.02±1.77	49.33±4.75	73.40±4.63	49.44±2.36
	GK	72.84±0.28	43.89±0.38	65.87±0.98	41.01±0.17
Deep Learning	AWE	73.93±1.94	51.54±3.61	74.45±5.83	50.46±1.91
	DGK	73.09±0.25	44.55±0.52	66.96±0.56	41.27±0.18
	PSCN	72.60±2.15	45.23±2.84	71.00±2.29	49.10±0.70
	DGCNN	73.76±0.49	47.83±0.85	70.03±0.86	48.70±4.54
Graph Capsule	GCAPS-CNN	77.71±2.51	48.50±4.10	71.69±3.40	50.10±1.72
	CapsGNN	79.62±0.91	50.27± 2.65	73.10±4.83	52.88±1.48
	MoCaNet (Ours)	80.50±3.10	54.53±4.80	75.70±4.30	50.10±2.01

compared methods can be categorized into three types, including graph kernel based methods (WL (Shervashidze et al. 2011), GK (Shervashidze et al. 2009), RW (Vishwanathan et al. 2010), DGK (Yanardag and Vishwanathan 2015), MLG (Kondor et al. 2018) and SP (Borgwardt and Kriegel 2005)), graph capsule network based methods (CapsGNN (Xinyi and Chen 2019), GCAPS-CNN (Verma and Zhang 2018) and TGCapsNN (Mallea, Meltzer, and Bentley 2019)) and other deep learning based methods (Graph2vec (Narayanan et al. 2017), AWE (Ivanov and Burnaev 2018), PSCN (Niepert, Ahmed, and Kutzkov 2016), DGCNN (Zhang et al. 2018) and ECC (Simonovsky and Komodakis 2017)).

Results

Table 2 lists the classification results on biology datasets, and Table 3 on social network datasets. In these two tables, we use red and blue to mark the top2 result respectively. For each experiment, we divide the dataset into ten folds. Eight of them are used for training, one of them for validation, and the other one for testing.

From Table 2 and Table 3, we can see that MoCaNet achieves the best results on 5 out of 10 datasets and comparable performance on the other 5 datasets, clearly showing the state-of-the-art of MoCaNet on graph classification tasks. More specifically on the biology datasets, MoCaNet is able to improve the accuracy by 0.67% and 1.57% on ENZYMES and MUTAG respectively. On the social network datasets, MoCaNet has obtained performance gains by 0.88%, 2.99%, and 1.25% on COLLAB, IMDB-M, and IMDB-B respectively.

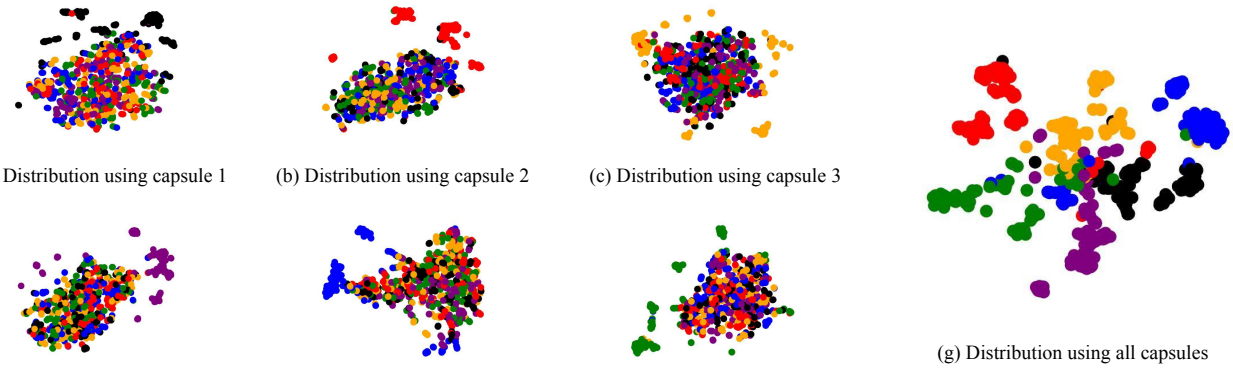
Ablation studies

To further evaluate the effect of different components in MoCaNet, we conduct ablation studies on MUTAG and PTC. Specifically, we study five variants of MoCaNet:

MoCaNet w/o Recon: In this architecture, the core graph reconstruction loss is removed from MoCaNet.

MoCaNet w/o Motif: In this architecture, we do not generate the motif graph and simply extract the graph feature embedding from the original graph directly.

MoCaNet w/o m-GCN: In this architecture, the m-GCN layers are removed, and the condensed graph is generated from the motif graph directly.



(a) Distribution using capsule 1 (b) Distribution using capsule 2 (c) Distribution using capsule 3

(d) Distribution using capsule 4 (e) Distribution using capsule 5 (f) Distribution using capsule 6

(g) Distribution using all capsules

Figure 4: Framework of the proposed structure-driven MoCaNet by full exploration of various structure in a graph. Best viewed in color.

Table 4: Ablation studies on different components in MoCaNet.

Method	MUTAG	PTC	Method	MUTAG	PTC
w/o Recon	88.42±6.31	64.00±1.71	w/o Cond	88.42±4.21	61.71±5.19
w/o Motif	90.00±5.71	62.85±5.71	w/o Cap	85.53±3.68	60.52±1.71
w/o m-GCN	86.31±3.1	62.57±6.00	MoCaNet	**90.52±4.21**	**64.86±3.72**

MoCaNet w/o Cond: In this architecture, the condensed operator is removed, and the capsule network is directly built from the graph feature embedding. We tackle the problem of various graph sizes by sharing the transform matrix.

MoCaNet w/o Cap: In this architecture, we remove the capsule network, and adopt a simple MLP to perform the graph classification.

The results are shown in Table 4. As shown, our proposed reconstruction loss improve the performance by 1.1% and 0.86% on MUTAG and PTC, respectively. The result of 'w/o Motif' shows that motifs can provide more structural information for better graph embedding learning. Without the low-level structure extractor, 'w/o m-GCN' or the higher-level high-level structure extractor 'w/o Cap' causes drastic decline. But the effectiveness of the middle-level structure extractor is not that obvious.

Visualization

To further evaluate the graph embedding generated by MoCaNet, we plot the graph distribution based on the graph capsules with t-SNE. The visualization experiment is made on ENZYMES.

The results of six different graph capsules and the combination of all capsules are shown in Figure 4. Each capsule is able to represent one certain graph. For example, the first graph capsule can well discriminate the graphs belong to the first class(Black), and the similar phenomenon can also be found in other five graph capsules. When combining all capsules, most graphs can be well discriminated.

Conclusion

In this paper, we propose a novel graph capsule model named MoCaNet that learns high-quality graph embedding in a structure-driven manner. In particular, MoCaNet captures the higher-level structure by aggregating the lower-level structure. In the graph classification task, compared to other state-of-the-art methods, our MoCaNet achieves the the best performance on 5 out 10 datasets. For future work, we plan to design a new routing mechanism that can aggregate the lower-level structure in a much better way as well as improve the robustness of MoCaNet.

References

Benson, A. R.; Gleich, D. F.; and Leskovec, J. 2016. Higher-order organization of complex networks. *Science* .

Borgwardt, K. M.; and Kriegel, H.-P. 2005. Shortest-path kernels on graphs. In *ICDM*.

Borgwardt, K. M.; Ong, C. S.; Schönauer, S.; Vishwanathan, S.; Smola, A. J.; and Kriegel, H.-P. 2005. Protein function prediction via graph kernels. *Bioinformatics* 21: i47–i56.

Bruna, J.; Zaremba, W.; Szlam, A.; and LeCun, Y. 2014. Spectral Networks and Locally Connected Networks on Graphs. In *ICLR*.

Dareddy, M. R.; Das, M.; and Yang, H. 2019. motif2vec: Motif Aware Node Representation Learning for Heterogeneous Networks. In *BigData*.

Debnath, A. K.; Lopez de Compadre, R. L.; Debnath, G.; Shusterman, A. J.; and Hansch, C. 1991. Structure-activity relationship of mutagenic aromatic and heteroaromatic nitro compounds. correlation with molecular orbital energies and hydrophobicity. *Journal of medicinal chemistry* 34(2): 786–797.

Defferrard, M.; Bresson, X.; and Vandergheynst, P. 2016. Convolutional neural networks on graphs with fast localized spectral filtering. In *NIPS*.

Dobson, P. D.; and Doig, A. J. 2003. Distinguishing enzyme structures from non-enzymes without alignments. *Journal of molecular biology* 330(4): 771–783.

Gori, M.; Monfardini, G.; and Scarselli, F. 2005. A new model for learning in graph domains. In *IJCNN*.

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

Hinton, G. E.; Krizhevsky, A.; and Wang, S. D. 2011. Transforming Auto-Encoders. In *ICANN*.

Hinton, G. E.; Sabour, S.; and Frosst, N. 2018. Matrix capsules with EM routing. In *ICLR*.

Ivanov, S.; and Burnaev, E. 2018. Anonymous Walk Embeddings. In *ICML*.

Kingma, D. P.; and Ba, J. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980* .

Kipf, T. N.; and Welling, M. 2017. Semi-supervised classification with graph convolutional networks. In *ICLR*.

Kondor, R.; Son, H. T.; Pan, H.; Anderson, B.; and Trivedi, S. 2018. Covariant compositional networks for learning graphs. *arXiv preprint arXiv:1801.02144* .

Krizhevsky, A.; Sutskever, I.; and Hinton, G. E. 2012. Imagenet classification with deep convolutional neural networks. In *NIPS*.

LeCun, Y.; Bottou, L.; Bengio, Y.; and Haffner, P. 1998. Gradient-based learning applied to document recognition. *Proceedings of the IEEE* 86(11): 2278–2324.

Mallea, M. D. G.; Meltzer, P.; and Bentley, P. J. 2019. Capsule Neural Networks for Graph Classification using Explicit Tensorial Graph Representations. *arXiv* .

Milo, R.; Shen-Orr, S.; Itzkovitz, S.; Kashtan, N.; Chklovskii, D.; and Alon, U. 2002. Network motifs: simple building blocks of complex networks. *Science* .

Monti, F.; Otness, K.; and Bronstein, M. M. 2018. Motifnet: a motif-based graph convolutional network for directed graphs. In *2018 IEEE Data Science Workshop (DSW)*, 225–228.

Narayanan, A.; Chandramohan, M.; Venkatesan, R.; Chen, L.; Liu, Y.; and Jaiswal, S. 2017. graph2vec: Learning distributed representations of graphs. *arXiv* .

Niepert, M.; Ahmed, M.; and Kutzkov, K. 2016. Learning convolutional neural networks for graphs. In *ICML*.

Rossi, R. A.; Ahmed, N. K.; Koh, E.; Kim, S.; Rao, A.; and Yadkori, Y. A. 2018. HONE: Higher-Order Network Embeddings. In *WWW*.

Sabour, S.; Frosst, N.; and Hinton, G. E. 2017. Dynamic Routing Between Capsules. In *NIPS*.

Sankar, A.; Zhang, X.; and Chang, K. C.-C. 2017. Motif-based convolutional neural network on graphs. *arXiv preprint arXiv:1711.05697* .

Scarselli, F.; Gori, M.; Tsoi, A. C.; Hagenbuchner, M.; and Monfardini, G. 2008. The graph neural network model. *IEEE Transactions on Neural Networks* 20(1): 61–80.

Schomburg, I.; Chang, A.; Ebeling, C.; Gremse, M.; Heldt, C.; Huhn, G.; and Schomburg, D. 2004. BRENDA, the enzyme database: updates and major new developments. *Nucleic acids research* 32: D431–D433.

Shervashidze, N.; Schweitzer, P.; Van Leeuwen, E. J.; Mehlhorn, K.; and Borgwardt, K. M. 2011. Weisfeiler-lehman graph kernels. *JMLR* 12(77): 2539–2561.

Shervashidze, N.; Vishwanathan, S.; Petri, T.; Mehlhorn, K.; and Borgwardt, K. 2009. Efficient graphlet kernels for large graph comparison. In *AISTATS*.

Simonovsky, M.; and Komodakis, N. 2017. Dynamic edge-conditioned filters in convolutional neural networks on graphs. In *CVPR*.

Toivonen, H.; Srinivasan, A.; King, R. D.; Kramer, S.; and Helma, C. 2003. Statistical evaluation of the predictive toxicology challenge 2000–2001. *Bioinformatics* 19(10): 1183–1193.

Tsourakakis, C. E.; Pachocki, J.; and Mitzenmacher, M. 2017. Scalable Motif-aware Graph Clustering. In *WWW*.

Verma, S.; and Zhang, Z. 2018. Capsule Graph Neural Network. In *ICML workshop*.

Vishwanathan, S. V. N.; Schraudolph, N. N.; Kondor, R.; and Borgwardt, K. M. 2010. Graph kernels. *JMLR* 11(Apr): 1201–1242.

Wale, N.; Watson, I. A.; and Karypis, G. 2008. Comparison of descriptor spaces for chemical compound retrieval and classification. *Knowledge and Information Systems* 14(3): 347–375.

Xinyi, Z.; and Chen, L. 2019. Capsule Graph Neural Network. In *ICLR*.

Yanardag, P.; and Vishwanathan, S. 2015. Deep graph kernels. In *SIGKDD*.

Ying, Z.; You, J.; Morris, C.; Ren, X.; Hamilton, W. L.; and Leskovec, J. 2018. Hierarchical Graph Representation Learning with Differentiable Pooling. In *NeurIPS*.

Zhang, M.; Cui, Z.; Neumann, M.; and Chen, Y. 2018. An end-to-end deep learning architecture for graph classification. In *AAAI*.