DYNAMIC GRAPH MESSAGE PASSING NETWORKS

Anonymous authors
Paper under double-blind review

ABSTRACT

Modelling long-range dependencies is critical for scene understanding tasks in computer vision. Although CNNs have excelled in many vision tasks, they are still limited in capturing long-range structured relationships as they typically consist of layers of local kernels. A fully-connected graph is beneficial for such modelling, however, its computational overhead is prohibitive. We propose a dynamic graph message passing network, that significantly reduces the computational complexity compared to related works modelling a fully-connected graph. This is achieved by adaptively sampling nodes in the graph, conditioned on the input, for message passing. Based on the sampled nodes, we dynamically predict node-dependent filter weights and the affinity matrix for propagating information between them. Using this model, we show significant improvements with respect to strong, state-of-the-art baselines on three different tasks and backbone architectures. Our approach also outperforms fully-connected graphs while using substantially fewer floating-point operations.

1 INTRODUCTION

Capturing long-range dependencies is crucial for complex scene understanding tasks such as semantic segmentation, instance segmentation and object detection. Although convolutional neural networks (CNNs) have excelled in a wide range of scene understanding tasks (Krizhevsky et al., 2012; Simonyan & Zisserman, 2015; He et al., 2016), they are still limited by their ability to capture these long-range interactions. To improve the capability of CNNs in this aspect, a recent popular model Non-local networks (Wang et al., 2018) proposes a generalisation of the attention model of (Vaswani et al., 2017) and achieves significant advance in several computer vision tasks.

Non-local networks essentially model pairwise structured relationships among all feature elements in a feature map to produce the attention weights which are used for feature aggregation. Considering each feature element as a node in a graph, Non-local networks effectively model a fully-connected feature graph and thus have a quadratic inference complexity with respect to the number of the feature elements. This is infeasible for dense prediction tasks on high-resolution imagery, as commonly encountered in semantic segmentation (Cordts et al., 2016). Moreover, in dense prediction tasks, capturing relations between all pairs of pixels is usually unnecessary due to the redundant information contained within the image (Fig. 1). Simply subsampling the feature map to reduce the memory requirements is also suboptimal, as such naïve subsampling would result in smaller objects in the image not being represented adequately.

To tackle the complexity issue of the fully connected models, graph convolution networks (GCNs) (Kipf & Welling, 2017; Gilmer et al., 2017), which propagate information along graph-structured input data, can alleviate the computational issues of non-local networks to a certain extent. However, this stands only if local neighbourhoods are considered for each node. Employing such local-connected graphs means that the long-range contextual information needed for complex vision tasks such as segmentation and detection (Rabinovich et al., 2007; Oliva & Torralba, 2007) will only be partially captured. Along this direction, GraphSAGE (Hamilton et al., 2017) introduces an efficient graph learning model based on a graph sampling method. However, the sampling considers a spatial uniform sampling strategy and is not dependant on the feature nodes. Thus the modelling capacity is also restricted as it basically assumes a static input graph where the neighbours for each node are fixed and the filter weights are shared among all nodes.

(a) Fully-connected message passing (b) Locally-connected message passing (c) Dynamic graph message passing

Figure 1: Contextual information is crucial for complex scene understanding tasks. To recognise the "boathouse", one needs to consider the "boat" and the "water" next to it. Fully-connected message passing models (a) are able to obtain this information, but are prohibitively expensive. Furthermore, they capture a lot of redundant information (*i.e.*"trees" and "sky"). Locally-connected models (b) are more efficient, but miss out on important context. Our proposed approach (c), dynamically samples a small subset of relevant feature nodes based on a *learned* dynamic sampling scheme, *i.e.* the *learned* position-specific random walk (indicated by the white dashed arrow lines), and also dynamically predicts filter weights and affinities (indicated by unique edge and square colors.), which are both conditioned on the sampled feature nodes.

To address the aforementioned shortcomings, we propose a novel dynamic graph message passing network (DGMN) model, targeting effective and efficient deep representational learning with a joint modeling of two key dynamic properties as illustrated in Fig. 1. Our contribution is twofold: (i) We dynamically samples the neighbourhood of a node from the feature graph and conditioned on the node features. The sampling is *position-specifically learned* with the model training. Intuitively, this allows the network to efficiently gather long-range context by only selecting a subset of the most relevant nodes in the graph; (ii) Based on the nodes that have been sampled, we further dynamically predict node-dependant filter weights and the affinity matrix, which are used to propagate information among the feature nodes via message passing. The dynamic weights and affinities are especially beneficial for us to specifically model each sampled feature context, thus leading to more effective message passing. Both of these dynamic properties are jointly optimised in a single model, and we modularise the DGMN as a network layer for simple deployment into existing networks.

We demonstrate the proposed model on the tasks of semantic segmentation, object detection and instance segmentation on the challenging Cityscapes (Cordts et al., 2016) and COCO (Lin et al., 2014) datasets. We achieve clearly better performance than the fully-connected Non-local model, while using substantially fewer floating point operations (FLOPs). Significantly, one variant of our model with dynamic filters and affinities (*i.e.* the second dynamic property) achieves similar performance to Non-local while only using 9.4% of its FLOPs and 25.3% of its parameters. Furthermore, "plugging" our module into existing networks, we show considerable improvements with respect to strong, state-of-the-art baselines on three different tasks and backbone architectures.

2 Related work

An early technique for modelling context for computer vision tasks involved conditional random fields. In particular, the DenseCRF model (Krhenbhl & Koltun, 2011) was popular as it modelled interactions between all pairs of pixels in an image. Although such models have been integrated into neural networks (Chen et al., 2015; Zheng et al., 2015; Arnab et al., 2016; Xu et al., 2017), they are limited by the fact that the pairwise potentials are based on simple handcrafted features, and mostly model on the discrete label space, thus not directly applicable in the feature learning task in which the feature variables are typically considered continuous. Coupled with the fact that CRFs are computationally expensive, CRFs are no longer used for most computer vision tasks.

A complementary technique for increasing the receptive field of CNNs was to use dilated convolutions (Chen et al., 2015; Yu & Koltun, 2016; Chen et al., 2017). Other modifications to the convolution operation include deformable convolution (Dai et al., 2017; Zhu et al., 2019), which learns the offset with respect to a predefined grid. While the deformable strategy could learn the random walks of the feature nodes, it only considers very local field, and the filters are *shared* across all convolutional positions. In contrast, our dynamic sampling aims to sample over the whole feature graph to obtain a large receptive field, and also the predicted affinities and the weights for message passing are *position specific* and *conditioned* on the dynamically sampled nodes. Our model is thus able to better capture position-based semantic context to enable more effective message passing among feature nodes.

The idea of graph node sampling has previously been explored in GraphSAGE (Hamilton et al., 2017). GraphSAGE simply uniformly samples nodes. In contrast, our sampling strategy is *learned* based on the node features. Specifically, we first sample the nodes uniformly in the spatial dimension, and then dynamically predict *walks* of each node conditioned on the node features. Therefore the sampling in our model is input-dependant and clearly different from GraphSAGE. Furthermore, GraphSAGE does not consider our second important property, *i.e.* the dynamic prediction of the affinities and the message passing kernels, and it does not explore the challenging detection and segmentation tasks.

We also note that Jia et al. (2016) developed an idea of "dynamic convolution", that is predicting a dynamic convolutional filter for each feature position. More recently, Wu et al. (2019) further reduced the complexity of this operation in the context of natural laguage processing with lightweight group convolutions. Unlike (Jia et al., 2016; Wu et al., 2019), we present a graph-based formulation, and jointly learn dynamic weights and dynamic affinities, which are conditioned on an *adaptively sampled* neighbourhood for each feature node in the graph using the proposed dynamic sampling strategy for effective message passing.

3 Dynamic Graph Message Passing Networks

3.1 Problem Definition and Notation

Given an input feature map interpreted as a set of feature vectors, *i.e.* $\mathbf{F} = \{\mathbf{f}_i\}_{i=1}^{N}$ with $\mathbf{f}_i \in \mathbb{R}^{1\times C}$, where N is the number of pixels and C is the feature dimension respectively, our goal is to learn a set of refined latent feature vectors $\mathbf{H} = \{\mathbf{h}_i\}_{i=1}^{N}$ by utilising hidden structured information among the feature vectors at different pixel locations. $\mathbf{H}$ has the same dimension as the observation $\mathbf{F}$. To learn such structured representations, we convert the feature map into a graph domain by constructing a feature graph $\mathcal{G} = \{\mathcal{V}, \mathcal{E}, A\}$ with $\mathcal{V}$ as its nodes, $\mathcal{E}$ as its edges and A as its adjacency matrix. Specifically, the nodes of the graph are represented by the latent feature vectors, *i.e.* $\mathcal{V} = \{\mathbf{h}_i\}_{i=1}^{N}$, and $A \in \mathbb{R}^{N\times N}$ is a binary or learnable matrix with self-loops describing the connections between nodes. In this work, we propose a novel dynamic graph message passing network Gilmer et al. (2017) for deep representation learning, which refines each graph feature node by passing messages on the graph $\mathcal{G}$. Different from existing message passing neural networks considering a fully- or locally-connected static graph (Wang et al., 2018; Gilmer et al., 2017), we propose a dynamic graph network model with two dynamic properties, *i.e.* dynamic sampling of graph nodes to approximate the full graph distribution, and dynamic prediction of node-conditioned filter weights and affinities, in order to achieve more efficient and effective message passing.

3.2 Graph Message Passing Neural Networks for Deep Representation Learning

Message passing neural networks (MPNNs) (Gilmer et al., 2017) present a generalised form of graph neural networks such as graph convolution networks (Kipf & Welling, 2017), gated graph sequential networks (Li et al., 2015) and graph attention networks (Veličković et al., 2018). In order to model structured graph data, in which latent variables are represented as nodes on an undirected or directed graph, feed-forward inference is performed through a message passing phase followed by a readout phase upon the graph nodes. The message passing phase usually takes T iteration steps to update feature nodes, while the readout phase is for the final prediction *e.g.* graph classification with updated nodes. In this work, we focus on the message passing phase for learning efficient and effective feature refinement, since well-represented features are critical in all downstream tasks. The message passing phase consists of two steps, *i.e.* a message calculation step M^t and a message updating step U^t. Given a latent feature node $\mathbf{h}_i^{(t)}$ at an iteration t, for computational efficiency, we consider a locally connected node field with $v_i \subset \mathcal{V}$ and $v_i \in \mathbb{R}^{(K\times C)}$, where $K \ll N$ is the number of sampled nodes in v_i. Thus we can define the message calculation step for node i operated locally as

$$\mathbf{m}_i^{(t+1)} = M^t\left(A_{i,j}, \{\mathbf{h}_1^{(t)}, \cdots, \mathbf{h}_K^{(t)}\}, \mathbf{w}_j\right) = \sum_{j\in\mathcal{N}(i)} A_{i,j}\mathbf{h}_j^{(t)}\mathbf{w}_j, \text{ with } A_{i,j} = A[i,j], \quad (1)$$

where $A_{i,j}$ describes the connection relationship *i.e.* the affinity between latent nodes $\mathbf{h}_i^{(t)}$ and $\mathbf{h}_j^{(t)}$, $\mathcal{N}(i)$ denotes a self-included neighborhood of the node $\mathbf{h}_i^{(t)}$ which can be derived from v_i and $\mathbf{w}_j \in \mathbb{R}^{C\times C}$ is a transformation matrix for message calculation on the hidden node $\mathbf{h}_j^{(t)}$. Then the

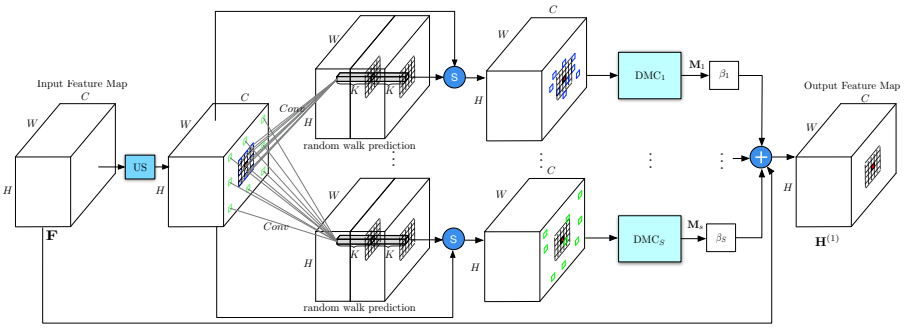


Figure 2: Overview of our proposed dynamic graph message passing network (DGMN). The neighbourhood used to update the feature representation of each node (we show a single node with a red square) is predicted dynamically conditioned on each input. This is done by first uniformly sampling (denoted by "US") a set of S neighbourhoods around each node. Each neighbourhood contains K (*e.g.* 3×3) sampled nodes. Here, the blue nodes were sampled with a low sampling rate, and the green ones with a high sampling rate. Walks are predicted (conditioned on the input) from these uniformly sampled nodes, denoted by the Ⓢ symbol representing the random walk sampling operation described in Sec. 3.3. $\mathrm{DMC}_1, \cdots, \mathrm{DMC}_S$ and $\beta_1, \cdots, \beta_S$ denotes S dynamic message calculation operations and S message scaling parameters, respectively. The DMC module is detailed in Figure 3. The symbol $\oplus$ indicates an element-wise addition operation.

message updating function U^t is applied to update the node $\mathbf{h}_i^{(t)}$ with a linear combination of the calculated message and the observation feature $\mathbf{f}_i$ at the node position i as:

$$\mathbf{h}_i^{(t+1)} = U^t\left(\mathbf{f}_i, \mathbf{m}_i^{(t+1)}\right) = \sigma\left(\mathbf{f}_i + \alpha_i^m \mathbf{m}_i^{(t+1)}\right), \tag{2}$$

where α_i^m of a learnable parameter for scaling the message, and the operation $\sigma(\cdot)$ is a non-linearity function *e.g.* ReLU. By iteratively performing message passing on each node with T steps, we obtain a refined feature map $\mathbf{H}^{(T)}$ as output.

3.3 From a Fully-Connected Graph to a Dynamic Sampled Graph

A fully-connected graph typically contains many connections and parameters, which thus naturally brings redundancy in the connections, and also makes the network optimisation more difficult especially when dealing with limited training data. Therefore, as in Eq. 1, a local node connection field is considered in the graph message passing network, in order to substantially reduce the computational overhead of large fully-connected graphs. However, in various computer vision tasks, such as detection and segmentation, learning deep representations capturing both local and global receptive fields is important for the model performance (Rabinovich et al., 2007; Oliva & Torralba, 2007). To maintain a large receptive field while utilising much fewer parameters than the fully-connected setting, we further explore dynamic sampling strategies in our proposed graph message passing network. We develop a uniform sampling scheme, which we extend to a predicted random walk sampling scheme which aims to reduce the redundancy found in a fully-connected graph. This sampling is performed in a dynamic fashion, meaning that for a given node $\mathbf{h}_i$, we aim to sample an optimal subset of v_i from $\mathcal{V}$ to update $\mathbf{h}_i$ via message passing as shown in Fig. 2.

Multiple uniform sampling for dynamic receptive fields. Uniform sampling is a commonly used strategy for graph node sampling (Leskovec & Faloutsos, 2006) based on Monte-Carlo estimation. To approximate the distribution of $\mathcal{V}$, we consider a set of S uniform sampling rates φ with $\varphi = \{\rho_q\}_{q=1}^S$, where ρ_q is a sampling rate. Let us assume that the latent feature nodes are located in a P-dimensional space $\mathbb{R}^P$. For instance, $P = 2$ for images considering the x- and y-dimensions of the image plane. For each latent node $\mathbf{h}_i$, a total of K neighbouring nodes are sampled from $\mathbb{R}^P$. The receptive field of v_i is thus determined by ρ_q and K. Note that the sampling rate ρ_q corresponds to the "dilation rate" often used in convolution (Yu & Koltun, 2016) and is thus able to capture a large receptive field whilst maintaining a small number of connected nodes. Thus we can achieve much lower computational overhead compared with fully-connected message passing in which typically all N nodes are used when one of the nodes is updated. Each node receives S complementary messages from distinct receptive fields for updating as

$$\mathbf{m}_i^{(t+1)} = \sum_q \sum_{j \in \mathcal{N}_q(i)} \beta_q A_{i,j}^q \mathbf{h}_j^{(t)} \mathbf{w}_j^q, \text{ with } A_{i,j}^q = A^q[i,j] \text{ and } q = 1, \cdots, S \tag{3}$$

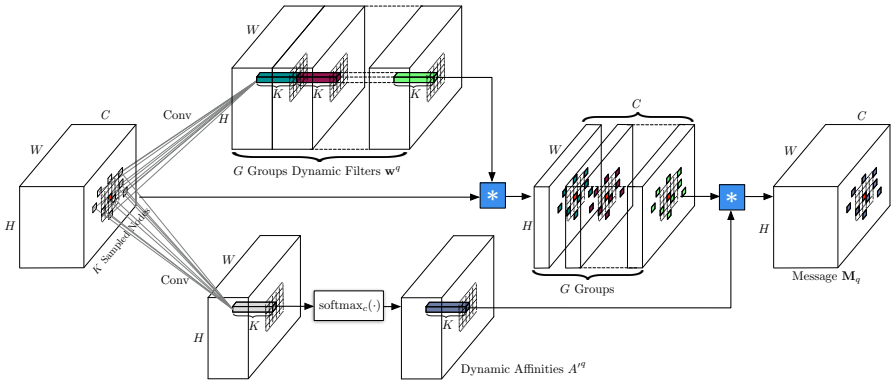


Figure 3: Schematic illustration of the proposed dynamic message passing calculation (DMC) module. The small red square indicates the receiving node whose message is calculated from its neighbourhood, *i.e.* the sampled K (*e.g.* 3×3) features nodes. The module accepts a feature map as input and produces its corresponding message map. The symbol $*$ denotes group convolution operation using the dynamically predicted and position specific group kernels and affinities.

where β_q is a weighting parameter for the message from the q-th sampling rate. A^q denotes a formed adjacent matrix under a sampling rate ρ_q, with $A^q_{i,j}$, $\mathbf{w}^q_j$ and $\mathcal{N}_q(i)$ defined analogously. The uniform sampling scheme acts as a linear sampler based on spatial distribution while not considering the original feature distribution of the hidden nodes, *i.e.* sampling independently of the node features. Eq. 2 can still be used to update the nodes.

Learning position-specific random walks for node-dependant adaptive sampling. To take into account the feature data distribution when sampling nodes, we further present a random walk strategy upon the uniform sampling. Walks around the uniformly sampled nodes could sample the graph in a non-linear and adaptive manner, and we believe that it could facilitate learning better approximation of the original feature distribution. The "random" here refers to the fact that the walks are predicted in a data-driven fashion from stochastic gradient descent. Given a set of uniformly sampled nodes $v^q_i \in \mathbb{R}^{K \times C}$ under a sampling rate ρ_q, the random walk of each node is further estimated based on the feature data of the sampled node set. Given the P-dimensional space where the nodes distribute ($P = 2$ for images), let us denote $\triangle \mathbf{d}^q_j \in \mathbb{R}^{P \times 1}$ as predicted walks from a uniformly sampled node $\mathbf{h}_j$ with $j \in \mathcal{N}_q(i)$. The node walk prediction can then be performed using a matrix transformation as

$$\triangle \mathbf{d}^q_j = \mathbf{W}^q_{i,j} v^q_i + \mathbf{b}^q_{i,j}, \tag{4}$$

where $\mathbf{W}^q_{i,j} \in \mathbb{R}^{P \times (K \times C)}$ and $\mathbf{b}^q_{i,j} \in \mathbb{R}^{P \times 1}$ are the matrix transformation parameters, which are learned separately for each node v^q_i. With the predicted walks, we can obtain a new set of adaptively sampled nodes v'^q_i, and generate the corresponding adjacent matrix A'^q, which can be used to calculate the messages as

$$\mathbf{m}^{(t+1)}_i = \sum_q \sum_{j \in \mathcal{N}_q(i)} \beta_q A'^q_{i,j} \varrho\left(\mathbf{h}'^{(t)}_j | \mathcal{V}, j, \triangle \mathbf{d}^q_j\right) \mathbf{w}^q_j, \text{ with } A'^q_{i,j} = A'^q[i,j], \tag{5}$$

where the function $\varrho(\cdot)$ is a bilinear sampler (Jaderberg et al., 2015) which samples a new feature node $\mathbf{h}'^{(t)}_j$ around $\mathbf{h}^{(t)}_j$ given the predicted walk $\triangle \mathbf{d}^q_j$ and the whole set of graph vertexes $\mathcal{V}$.

3.4 Joint Learning of Node-Conditioned Dynamic Filters and Affinities

In the message calculation formulated in Eq. 5, the set of weights $\{\mathbf{w}^q_j\}^K_{j=1}$ of the filter is shared for each adaptively sampled node field v'^q_i. However, since each v'^q_i essentially defines a node-specific local feature context, it is more meaningful to use a node-conditioned filter to learn the message for each hidden node $\mathbf{h}'^{(t)}_i$. In additional to the filters for the message calculation in Eq. 5, the affinity $A'_{i,j}$ of any pair of nodes $\mathbf{h}'^{(t)}_i$ and $\mathbf{h}'^{(t)}_j$ could be also be predicted and should also be conditioned on the node field v'^q_i, since the affinity reweights the message passing only in v'^q_i. As shown in Fig. 3, we thus use matrix transformations to simultaneously estimate the dynamic filter and affinity which are both conditioned on v'^q_i,

$$\{\mathbf{w}^q_j, A'^q_{i,j}\} = \mathbf{W}^{k,A}_{i,j} v'^q_i + \mathbf{b}^{k,A}_{i,j}, \tag{6}$$

Figure 4: Visualisation of the nodes sampled via *learning* the random walks with our network. The red point indicates a receiving node i. Different color families (*i.e.* yellow and blue) indicate the learned position specific weights and affinities of the sampled nodes. Different colors in the same family show the sampled nodes with different sampling rates for the same receiving node.

$$A'^{q}_{i,j} = \mathrm{softmax}_c(A'^{q}_{i,j}) = \frac{\exp(A'^{q}_{i,j})}{\sum_{l\in\mathcal{N}_q(i)}\exp(A'^{q}_{i,l})}, \tag{7}$$

where the function $\mathrm{softmax}_c(\cdot)$ denotes a softmax operation along the channel axis, which is used to perform a normalisation on the estimated affinity $A'^{q}_{i,j} \in \mathbb{R}^1$. $\mathbf{W}^{k,A}_{i,j} \in \mathbb{R}^{(G\times C+1)\times(K\times C)}$ and $\mathbf{b}^{k,A}_{i,j} \in \mathbb{R}^{(G\times C+1)}$ are matrix transformation parameters. To reduce the number of the filter parameters, we consider the group transformation (Chollet, 2017) with a set of G groups split from the total C feature channels, and $G \ll C$, *i.e.* each group of C/G feature channels shares the same set of filter parameters. For the dynamic weights, we relax them in the prediction without using non-linearity after the matrix transformation. The predicted dynamic filter weights and the affinities are used in Eq. 5 for dynamic message calculation.

3.5 Modular Instantiation

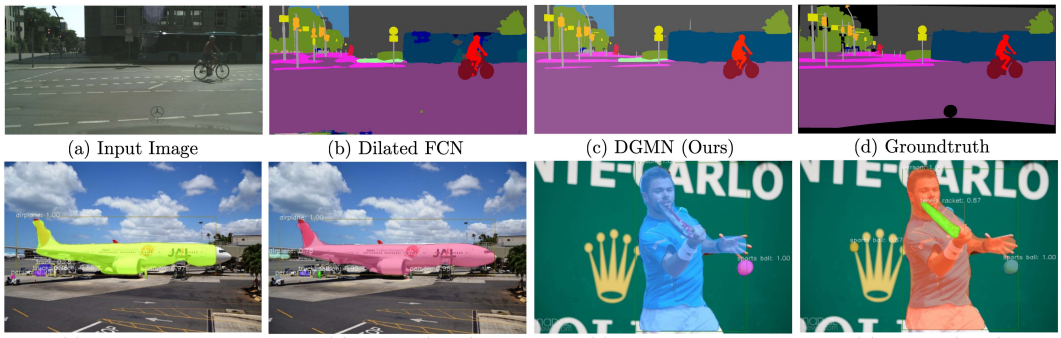
Figures 2 and 3 shows how our proposed dynamic graph message passing network (DGMN) can be implemented in neural networks. The proposed module accepts a single feature map $\mathbf{F}$ as input, which can be derived from any CNN layer. $\mathbf{H}^{(0)}$ denotes an initial state of the latent feature map, $\mathbf{H}$, and is initialised with $\mathbf{F}$. $\mathbf{H}$ and $\mathbf{F}$ have the same dimension, *i.e.* $\mathbf{F}, \mathbf{H} \in \mathbb{R}^{H\times W\times C}$, where H, W and C are the height, width and the number of feature channels of the feature map respectively. We first define a set of S uniform sampling rates (we show two uniform sampling rates in Fig. 2 for clarity). The uniform and the random walk sampler sample the nodes from the full graph and return the node indices for subsequent dynamic message calculation (DMC) in Fig. 3. The matrix transformation $\mathbf{W}^{q}_{i,j}$ to estimate the node random walk in Eq. 4 is implemented by a 3×3 convolution layer. Note that other sampling strategies could also be flexibly employed in our framework. The sampled feature nodes are processed along two data paths: one for predicting the node-dependant dynamic affinities $A'^{q} \in \mathbb{R}^{H\times W\times K}$ and another path for dynamic filters $\mathbf{w}^{q} \in \mathbb{R}^{H\times W\times K\times G}$ where K (*e.g.* , 3×3) is the kernel size for the receiving node. The matrix transformation $\mathbf{W}^{k,A}_{i,j}$ used to jointly predict the dynamic filters and affinities in Eq 6 is implemented by a 3×3 convolution layer. Message $\mathbf{M}_q \in \mathbb{R}^{H\times W\times C}$ corresponding to the q-th sampling rate is then scaled to perform a linear combination with the observation feature map $\mathbf{F}$, to produce a refined feature map $\mathbf{H}^{(1)}$ as output. To balance performance and efficiency, as in existing graph-based feature learning models (Wang et al., 2018; Chen et al., 2019; Lin et al., 2015), we also perform $T = 1$ iteration of message updating.

4 Experiments

4.1 Experimental setup

Tasks and Datasets. We evaluate our proposed model on two challenging public benchmarks, *i.e.* Cityscapes (Cordts et al., 2016) for semantic segmentation, and COCO (Lin et al., 2014) for object detection and instance segmentation. Both datasets have hidden test sets which are evaluated on a public evaluation server. We follow the standard protocol and evaluation metrics used by these public benchmarks. More details can be found in Appendix A.2.

Baseline models. For semantic segmentation on Cityscapes, our baseline architecture is Dilated-FCN (Yu & Koltun, 2016) with a ResNet-101 backbone pretrained on ImageNet. A 3×3 convolution layer, together with batch normalisation and ReLU is used after the backbone to produce a dimension-reduced feature map of 512 channels which is then fed into the proposed DGMN module as input.



(a) Input Image (b) Dilated FCN (c) DGMN (Ours) (d) Groundtruth

(a) Mask – RCNN (b) DGMN (Ours) (a) Mask – RCNN (b) DGMN (Ours)

Figure 5: Qualitative examples of the semantic segmentation on Cityscapes (the first row), and object detection and instance segmentation on COCO (the second row).

For the task of object detection and instance segmentation on COCO, our baseline is Mask-RCNN (He et al., 2017; Massa & Girshick, 2018) with FPN and ResNet/ResNeXt (He et al., 2016; Xie et al., 2017) as a backbone architecture. We insert one or multiple random initialised DGMN modules into the backbone for deploying our approach for the feature learning. Across all tasks and datasets, we consider Non-local networks (Wang et al., 2018) as an additional baseline. We use single scale and single model testing for all the evaluation experiments on COCO without using other complementary performance boosting tricks.

To have a direct comparison with the Deformable Convolution method (Dai et al., 2017; Zhu et al., 2019), we consider two baselines: (i) deformable message passing, which is a variant of our model using the deformable convolution for message calculation without using the proposed dynamic sampling and dynamic weights/affinities strategies, and (ii) the original deformable method which replaces the convolutional operations as in (Dai et al., 2017).

To demonstrate the effectiveness of the proposed different components of our model, we conduct ablation studies of: (i) DGMN w/ DA, which adds the dynamic affinity (DA) strategy on the DGMN base model; (ii) DGMN w/ DA+DW, adding the proposed dynamic weights (DW) prediction upon DGMN w/ DA; (iii) DGMN w/ DA+DW+DS, which is our full model with the dynamic sampling (DS) scheme added upon DGMN w/ DA+DW. Note that DGMN Base was described in Sec. 3.2, DS in Sec. 3.3, and DA and DW in 3.4.

Implementation details. When predicting dynamic filter weights, we used the grouping parameter $G = 4$. For our experiments on Cityscapes, the sample rates are set to $\varphi = \{1, 6, 12, 24, 36\}$. For experiments on COCO, we use smaller sampling rates of $\varphi = \{1, 4, 8, 12\}$. The effect of this hyperparameter is studied in Appendix A.3, whislt additional implementation and training details are described in Appendix A.2.

4.2 Model Analysis

Effectiveness of the dynamic sampling strategy. Table 1a shows quantitative results of semantic segmentation on the Cityscapes validation set. DGMN w/ DA+DW+DS clearly outperforms DGMN w/ DA+DW on the challenging segmentation task, meaning that the feature-conditioned adaptive sampling based on learned random walks is more effective compared with spatial uniform sampling strategy when selecting nodes. More importantly, all variants of our module for both semantic segmentation and object detection that use dynamic sampling (Table 1a and Table 2) achieve higher performance than a fully-connected model (*i.e.* Non-local Wang et al. (2018)) with substantially fewer FLOPs. This emphasises the performance benefits of our dynamic graph sampling model.

Effectiveness of joint learning the dynamic filters and affinities. As shown in Table 1a, DGMN w/ DA is 1.5 points better than Dilated FCN baseline with only a slight increase in FLOPs and parameters, showing the benefit of using predicted dynamic affinities for reweighting the messages in message passing. By further employing the estimated dynamic filter weights for message calculation, the performance increases substantially from a mIoU of 76.5% to 79.1%, which is almost the same as the 79.2% of the Non-local model (Wang et al., 2018). Furthermore, our approach only uses 9.4% of the FLOPs and 25.3% of the parameters compared to Non-local. These results clearly demonstrate our motivation of joint learning the dynamic filters and dynamic affinities from sampled graph nodes.

	mIoU (%)	Params	FLOPs
Dilated FCN (Yu & Koltun, 2016)	75.0	–	–
+ Deformable (Zhu et al., 2019)	78.2	+1.31M	+12.34G
+ ASPP (Chen et al., 2017)	78.9	+4.42M	+38.45G
+ Non-local (Wang et al., 2018)	79.2	+2.88M	+73.33G
+ DGMN w/ DA	76.5	+0.57M	+5.32G
+ DGMN w/ DA+DW	79.1	+0.73M	+6.88G
+ DGMN w/ DA+DW+DS	**80.4**	+3.02M	+28.45G

(a) Ablation study on the validation set.

	Backbone	mIoU (%)
PSPNet (Zhao et al., 2017)	R101	78.4
PSANet (Zhao et al., 2018)	R101	80.1
DenseASPP (Yang et al., 2018)	D161	80.6
GloRe (Chen et al., 2019)	R101	80.9
Non-local (Wang et al., 2018)	R101	81.2
CCNet (Huang et al., 2019)	R101	81.4
DANet (Fu et al., 2019)	R101	81.5
DGMN (Ours)	R101	**81.8**

(b) State-of-the-art comparison on the test set.

Table 1: Results comparison for the semantic segmentation task on Cityscapes. In the ablation study, all the methods are evaluated in single-scale mIoU and with a ResNet-101 backbone. For the state-of-art comparison, all the methods are trained on the fine-annotation training and validation sets.

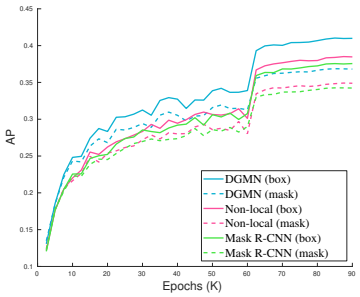


Figure 6: Validation curves of $\mathrm{AP}^{\mathrm{box}}$ and $\mathrm{AP}^{\mathrm{mask}}$ on COCO for Mask-RCNN baseline, Non-local and the proposed DGMN. The number of training epochs is 90K.

	AP^{b}	$\mathrm{AP}^{\mathrm{b}}_{50}$	$\mathrm{AP}^{\mathrm{b}}_{75}$	AP^{m}	$\mathrm{AP}^{\mathrm{m}}_{50}$	$\mathrm{AP}^{\mathrm{m}}_{75}$
Mask R-CNN baseline	37.8	59.1	41.4	34.4	55.8	36.6
+ Non-local (Wang et al., 2018)	38.5	60.0	41.8	34.9	56.5	37.3
+ GCNet (Cao et al., 2019)	38.1	60.0	41.2	34.9	56.5	37.2
+ CCNet (Huang et al., 2019)	39.3	-	-	36.1	-	-
+ Deformable Message Passing	38.7	60.4	42.4	35.0	56.9	37.4
+ DGMN	**39.5**	**61.0**	**43.3**	**35.7**	**58.0**	**37.9**
+ GCNet (C5) (Cao et al., 2019)	38.7	61.1	41.7	35.2	57.4	37.4
+ Deformable (C5) (Zhu et al., 2019)	39.9	-	-	34.9	-	-
+ DGMN (C5)	**40.2**	**62.0**	**43.4**	**36.0**	**58.3**	**38.2**

Table 2: Quantitative results of different models on the COCO 2017 validation set for object detection (AP^{b}) and instance segmentation (AP^{m}). C5 denotes inserting DGMN after all 3×3 convolutional layers in *res5*. All methods are based on the Mask R-CNN with ResNet-50 as backbone.

Comparison with other baselines. We clearly outperform the ASPP module of Deeplab v3 (Chen et al., 2017) which also considers increasing the receptive field via using various dilation rates. Notably, we beat the direct competitor Non-local (Wang et al., 2018) which models a fully-connected graph whilst using only 39% of the FLOPs of Non-local. This suggests that a fully-connected graph models redundant information, and further confirms the performance and efficiency of our model.

For fair comparison with Non-local (Wang et al., 2018) as well as other alternatives on COCO, we insert one random initialised DGMN module right before the last residual block of *res4* (Table 2). GCNet and CCNet both aim to reduce the complexity of the fully connected graph model Non-local. Our proposed DGMN model substantially improves upon these strong baselines and alternatives. Figure 6 further shows the validation curves of our method and different baselines in terms of the metric AP. Our method is consistently better than the Non-local and Mask R-CNN baseline throughout the training procedure.

Effectiveness of multiple DGMN modules. We further show the effectiveness of our approach for representation learning by inserting multiple of our DGMN modules into the ResNet-50 backbone. Specifically, we add our full DGMN module *after* all 3×3 conv layers in *res5* which we denote as "C5". The second part of Table 2 shows that our model significantly improves upon the Mask R-CNN baseline with the improvements of 2.4 points for the $\mathrm{AP}^{\mathrm{box}}$ on object detection, and 1.6 points for the $\mathrm{AP}^{\mathrm{mask}}$ on instance segmentation. We insert the GCNet module (Cao et al., 2019) in the same places for comparison, and our model achieves clearly better performance. A straightforward comparsion to the deformable method of Zhu et al. (2019) is the deformable message passing baseline in which we disable the proposed dyanamic sampling and the dynamic weights and affinities learning strategies. In Table 2, our model significantly improves the deformable message passing method, which is a direct evidence of the effectiveness of joint modeling the dynamic sampling and dynamic filters and affinities for more effective feature learning. Furthermore, we consider the original Deformable Conv method (Zhu et al., 2019) in C5, which is complementary to our model since it is plugged *before* 3×3 layers to replace the convolution operations. Our approach also achieves better performance compared with it.

	Backbone	AP^{box}	AP^{box}_{50}	AP^{box}_{75}	AP^{mask}	AP^{mask}_{50}	AP^{mask}_{75}
Mask R-CNN baseline	ResNet 50	38.0	59.7	41.5	34.6	56.5	36.6
+ DGMN (C5)		40.2	62.5	43.9	36.2	59.1	38.4
+ DGMN (C4, C5)		**41.0**	**63.2**	**44.9**	**36.8**	**59.8**	**39.1**
Mask R-CNN baseline	ResNet 101	40.2	61.9	44.0	36.2	58.6	38.4
+ DGMN (C5)		**41.9**	**64.1**	**45.9**	**37.6**	**60.9**	**40.0**
Mask R-CNN baseline	ResNeXt 101	42.6	64.9	46.6	38.3	61.6	40.8
+ DGMN (C5)		**44.3**	**66.8**	**48.4**	**39.5**	**63.3**	**42.1**

Table 3: Quantitative results via plugging our DGMN module on different backbones on the COCO 2017 `test-dev` set for object detection (AP^{box}) and instance segmentation (AP^{mask}).

4.3 Comparison to State-of-the-art

Performance on Cityscapes test set. Table 1b compares our approach with state-of-the-art methods on Cityscapes. Note that all methods are trained using only the fine annotations and evaluated on the public evaluation server as test-set annotations are withheld from the public. As shown in the table, DGMN (ours) achieves an mIoU of 81.8%, surpassing all previous works. Among competing methods, GloRe (Chen et al., 2019), Non-local (Wang et al., 2018), CCNet (Huang et al., 2019) and DANet (Fu et al., 2019) are the most related to us as they all based on graph neural network modules. Note that we followed common practice and employed several complementary strategies used in semantic segmentation to boost performance, including Online Hard Example Mining (OHEM) (Fu et al., 2019), Multi-Grid (Chen et al., 2017) and Multi-Scale (MS) ensembling (Zhao et al., 2017). The contribution of each strategy on the final performance is reported in the Appendix A.3.

Performance on COCO 2017 test set. Table 3 presents our results on the COCO test-dev set, where we compare to various existing strong backbones. By inserting DGMN into all layers of C4 and C5, we substantially improve the performance of Mask R-CNN, observing a gain of of 3.0 and 2.2 points on the AP^{box} and the AP^{mask} of object detection and instance segmentation respectively. We observe similar improvements when using the ResNet-101 or ResNeXt-101 backbones as well, showing that our proposed DGMN module generalises to multiple backbone architectures. Furthermore, we compare ours with previous state-of-the-arts shown in Appendix A.4.

5 Conclusion

We proposed Dynamic Graph Message Passing Networks, a novel graph neural network module that dynamically determines the graph structure for each input. It learns dynamic sampling of a small set of relevant neighbours for each node, and also predicts the weights and affinities dependant on the feature nodes to propagate information through this sampled neighbourhood. This formulation significantly reduces the computational cost of static, fully-connected graphs such as Non-local networks (Wang et al., 2018) which contain many redundancies. This is exhibited by the fact that we are able to clearly improve upon the accuracy of Non-local, and several state-of-art baselines, on three complex computer vision problems.

References

Anurag Arnab, Sadeep Jayasumana, Shuai Zheng, and Philip HS Torr. Higher order conditional random fields in deep neural networks. In *ECCV*, 2016.

Yue Cao, Jiarui Xu, Stephen Lin, Fangyun Wei, and Han Hu. Gcnet: Non-local networks meet squeeze-excitation networks and beyond. *arXiv preprint arXiv:1904.11492*, 2019.

Liang-Chieh Chen, George Papandreou, Iasonas Kokkinos, Kevin Murphy, and Alan L. Yuille. Semantic image segmentation with deep convolutional nets and fully connected CRFs. In *ICLR*, 2015.

Liang-Chieh Chen, George Papandreou, Florian Schroff, and Hartwig Adam. Rethinking atrous convolution for semantic image segmentation. *arXiv preprint arXiv:1706.05587*, 2017.

Liang-Chieh Chen, Yukun Zhu, George Papandreou, Florian Schroff, and Hartwig Adam. Encoder-decoder with atrous separable convolution for semantic image segmentation. In *ECCV*, 2018.

Yunpeng Chen, Marcus Rohrbach, Zhicheng Yan, Shuicheng Yan, Jiashi Feng, and Yannis Kalantidis. Graph-based global reasoning networks. In *CVPR*, 2019.

François Chollet. Xception: Deep learning with depthwise separable convolutions. In *CVPR*, 2017.

Marius Cordts, Mohamed Omran, Sebastian Ramos, Timo Rehfeld, Markus Enzweiler, Rodrigo Benenson, Uwe Franke, Stefan Roth, and Bernt Schiele. The cityscapes dataset for semantic urban scene understanding. In *CVPR*, 2016.

Jifeng Dai, Kaiming He, and Jian Sun. Boxsup: Exploiting bounding boxes to supervise convolutional networks for semantic segmentation. In *ICCV*, 2015.

Jifeng Dai, Yi Li, Kaiming He, and Jian Sun. R-fcn: Object detection via region-based fully convolutional networks. In *NeurIPS*, 2016.

Jifeng Dai, Haozhi Qi, Yuwen Xiong, Yi Li, Guodong Zhang, Han Hu, and Yichen Wei. Deformable convolutional networks. In *ICCV*, 2017.

Cheng-Yang Fu, Wei Liu, Ananth Ranga, Ambrish Tyagi, and Alexander C Berg. Dssd: Deconvolutional single shot detector. *arXiv preprint arXiv:1701.06659*, 2017.

Jun Fu, Jing Liu, Haijie Tian, Zhiwei Fang, and Hanqing Lu. Dual attention network for scene segmentation. In *CVPR*, 2019.

Justin Gilmer, Samuel S Schoenholz, Patrick F Riley, Oriol Vinyals, and George E Dahl. Neural message passing for quantum chemistry. In *ICML*, 2017.

Will Hamilton, Zhitao Ying, and Jure Leskovec. Inductive representation learning on large graphs. In *NeurIPS*, 2017.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, 2016.

Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask r-cnn. In *ICCV*, 2017.

Zilong Huang, Xinggang Wang, Lichao Huang, Chang Huang, Yunchao Wei, and Wenyu Liu. Ccnet: Criss-cross attention for semantic segmentation. In *ICCV*, 2019.

Max Jaderberg, Karen Simonyan, Andrew Zisserman, and Koray Kavukcuoglu. Spatial transformer networks. In *NeurIPS*, 2015.

Xu Jia, Bert De Brabandere, Tinne Tuytelaars, and Luc V Gool. Dynamic filter networks. In *NeurIPS*, 2016.

Thomas N Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. In *ICLR*, 2017.

Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. ImageNet classification with deep convolutional neural networks. In *NeurIPS*, 2012.

Philipp Krhenbhl and Vladlen Koltun. Efficient inference in fully connected crfs with gaussian edge potentials. *NeurIPS*, 2011.

Hei Law and Jia Deng. Cornernet: Detecting objects as paired keypoints. In *ECCV*, 2018.

Jure Leskovec and Christos Faloutsos. Sampling from large graphs. In *SIGKDD*, 2006.

Qizhu Li, Anurag Arnab, and Philip HS Torr. Holistic, instance-level human parsing. In *BMVC*, 2017.

Yujia Li, Daniel Tarlow, Marc Brockschmidt, and Richard Zemel. Gated graph sequence neural networks. In *ICLR*, 2015.

Guosheng Lin, Chunhua Shen, Ian Reid, and Anton van den Hengel. Deeply learning the messages in message passing inference. In *NeurIPS*, 2015.

Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *ECCV*, 2014.

Tsung-Yi Lin, Piotr Dollr, Ross B. Girshick, Kaiming He, Bharath Hariharan, and Serge J. Belongie. Feature pyramid networks for object detection. In *CVPR*, 2017.

Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. Focal loss for dense object detection. In *ICCV*, 2017.

Wei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott Reed, Cheng-Yang Fu, and Alexander C Berg. Ssd: Single shot multibox detector. In *ECCV*, 2016.

Francisco Massa and Ross Girshick. maskrcnn-benchmark: Fast, modular reference implementation of Instance Segmentation and Object Detection algorithms in PyTorch. `https://github.com/facebookresearch/maskrcnn-benchmark`, 2018.

Aude Oliva and Antonio Torralba. The role of context in object recognition. *Trends in cognitive sciences*, 2007.

Jiangmiao Pang, Kai Chen, Jianping Shi, Huajun Feng, Wanli Ouyang, and Dahua Lin. Libra r-cnn: Towards balanced learning for object detection. In *CVPR*, 2019.

Tobias Pohlen, Alexander Hermans, Markus Mathias, and Bastian Leibe. Full-resolution residual networks for semantic segmentation in street scenes. In *CVPR*, 2017.

Andrew Rabinovich, Andrea Vedaldi, Carolina Galleguillos, Eric Wiewiora, and Serge Belongie. Objects in context. In *ICCV*, 2007.

Joseph Redmon and Ali Farhadi. Yolov3: An incremental improvement. *arXiv preprint arXiv:1804.02767*, 2018.

Samuel Rota Bulò, Lorenzo Porzi, and Peter Kontschieder. In-place activated batchnorm for memory-optimized training of dnns. In *CVPR*, 2018.

Abhinav Shrivastava, Abhinav Gupta, and Ross Girshick. Training region-based object detectors with online hard example mining. In *CVPR*, 2016.

Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. In *ICLR*, 2015.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *NeurIPS*, 2017.

Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Lio, and Yoshua Bengio. Graph attention networks. In *ICLR*, 2018.

Xiaolong Wang, Ross Girshick, Abhinav Gupta, and Kaiming He. Non-local neural networks. In *CVPR*, 2018.

Felix Wu, Angela Fan, Alexei Baevski, Yann N Dauphin, and Michael Auli. Pay less attention with lightweight and dynamic convolutions. In *ICLR*, 2019.

Saining Xie, Ross Girshick, Piotr Dollár, Zhuowen Tu, and Kaiming He. Aggregated residual transformations for deep neural networks. In *CVPR*, 2017.

Dan Xu, Wanli Ouyang, Xavier Alameda-Pineda, Elisa Ricci, Xiaogang Wang, and Nicu Sebe. Learning deep structured multi-scale features using attention-gated crfs for contour prediction. In *NeurIPS*, 2017.

Maoke Yang, Kun Yu, Chi Zhang, Zhiwei Li, and Kuiyuan Yang. Denseaspp for semantic segmentation in street scenes. In *CVPR*, 2018.

Changqian Yu, Jingbo Wang, Chao Peng, Changxin Gao, Gang Yu, and Nong Sang. Bisenet: Bilateral segmentation network for real-time semantic segmentation. In *ECCV*, 2018.

Fisher Yu and Vladlen Koltun. Multi-scale context aggregation by dilated convolutions. *ICLR*, 2016.

Fisher Yu and Vladlen Koltun. Multi-scale context aggregation by dilated convolutions. In *ICLR*, 2016.

Yuhui Yuan and Jingdong Wang. Ocnet: Object context network for scene parsing. *arXiv preprint arXiv:1809.00916*, 2018.

Hengshuang Zhao, Jianping Shi, Xiaojuan Qi, Xiaogang Wang, and Jiaya Jia. Pyramid scene parsing network. In *CVPR*, 2017.

Hengshuang Zhao, Yi Zhang, Shu Liu, Jianping Shi, Chen Change Loy, Dahua Lin, and Jiaya Jia. Psanet: Point-wise spatial attention network for scene parsing. In *ECCV*, 2018.

Shuai Zheng, Sadeep Jayasumana, Bernardino Romera-Paredes, Vibhav Vineet, Zhizhong Su, Dalong Du, Chang Huang, and Philip H. S. Torr. Conditional random fields as recurrent neural networks. In *ICCV*, 2015.

Xizhou Zhu, Han Hu, Stephen Lin, and Jifeng Dai. Deformable convnets v2: More deformable, better results. In *CVPR*, 2019.

A ADDITIONAL EXPERIMENTS

A.1 QUALITATIVE RESULTS

Figures 7 and 8 show qualitative results for semantic segmentation (on Cityscapes) and instance segmentation (on COCO) respectively.

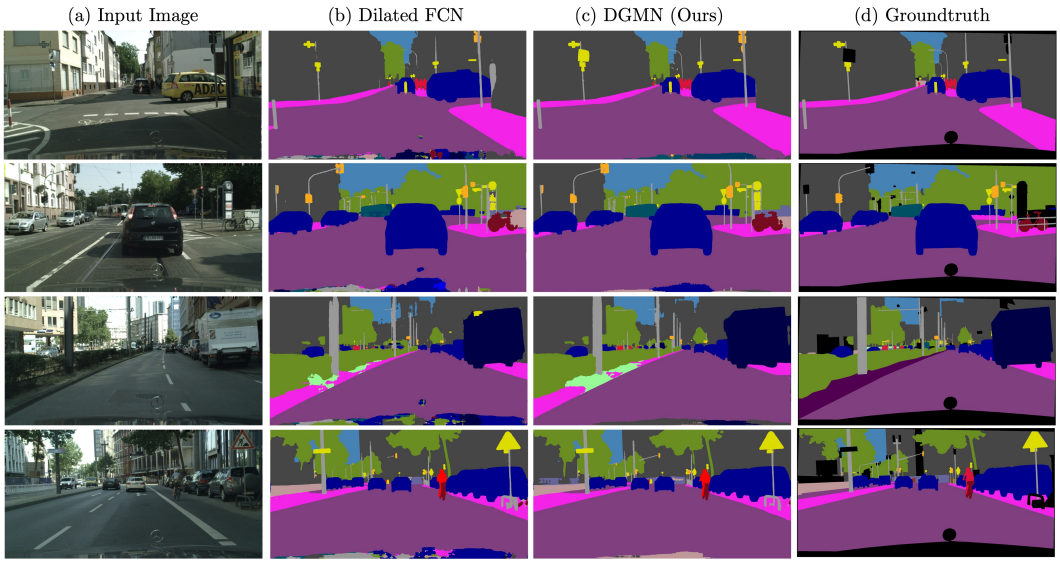


Figure 7: Qualitative results of Dilated FCN baseline Yu & Koltun (2016) and DGMN (ours) on the Cityscapes dataset.

A.2 ADDITIONAL EXPERIMENTAL DETAILS

A.2.1 DATASETS

Cityscapes has densely annotated semantic labels for 19 categories in urban road scenes, and contains a total of 5000 finely annotated images, divided into 2975, 500, and 1525 images for training, validation and testing, respectively. We do not use the coarsely annotated data in our experiments. The images of this dataset have a high resolution of 1024×2048. Following the standard evaluation protocol (Cordts et al., 2016), the metric of mean Intersection over Union (mIoU) averaged over all classes is used. *COCO 2017* consists of 80 object classes with a training set of 118,000 images, a validation set of 5000 images, and a test set of 2000 images. We follow the standard COCO evaluation metrics (Lin et al., 2014; He et al., 2017) to evaluate the performance of object detection and instance segmentation, employing the metric of mean average-precision (mAP) at different box and mask IoUs respectively.

A.2.2 SEMANTIC SEGMENTATION ON CITYSCAPES

For the semantic segmentation task on Cityscapes, we follow (Zhao et al., 2017) and use a polynomial learning rate decay with an initial learning rate of 0.01. The momentum and the weight decay are set to 0.9 and 0.0001 respectively. We use 4 Nvidia V100 GPUs, batch size 8 and run $40k$ iterations for the experiments on Cityscapes training set. For data augmentation, random cropping with a crop size of 769 and random mirror flipping are applied on the fly during training. Note that following common practice (Zhao et al., 2017; Yuan & Wang, 2018; Zhao et al., 2018; Rota Bulò et al., 2018) we used synchronised batch normalisation for better estimation of the batch statistics for the experiments on Cityscapes. When predicting dynamic filter weights, we use the grouping parameter $G = 4$. For the experiments on Cityscapes, we use a set of the sampling rates of $\varphi = \{1, 6, 12, 24, 36\}$.

Figure 8: Qualitative examples of the object detection task on the COCO validation dataset. The first and third rows are the results from the Mask R-CNN baseline (Massa & Girshick, 2018; He et al., 2017). The second and fourth row are the detection results from our DGMN approach.

A.2.3 Object detection and instance segmentation on COCO

The backbone parameters of all the models in the experiments are pretrained on the ImageNet classification. The input images are resized such that their shorter side is of 800 pixels and the longer side is limited to 1333 pixels. The batch size is set to 16. The initial learning rate is set to 0.02 with a decrease by a factor of 0.1 after 60k and 80k iterations, and finally terminates at 90k iterations. Following (Massa & Girshick, 2018), the training warm-up is employed by using a smaller learning rate of 0.02×0.3 for the first 500 iterations. All the batch normalisation layers in the backbone are "frozen" during the fine-tuning on COCO. Our models and all baselines are trained with the typical "1x" training settings from the public Mask R-CNN benchmark (Massa & Girshick, 2018) for all experiments on COCO. When predicting dynamic filter weights, we use the grouping parameter $G = 4$. For the experiments on COCO, a set of the sampling rates of $\varphi = \{1, 4, 8, 12\}$ is considered. We train models on COCO training set only and test on validation set and test-dev set.

A.3 Additional ablation studies

Effectiveness of different training and inference strategies. To boost the performance on Cityscapes, we followed common practice and employed several complementary strategies used in semantic segmentation, including Online Hard Example Mining (OHEM) Shrivastava et al. (2016); Pohlen et al. (2017); Li et al. (2017); Yuan & Wang (2018); Yu et al. (2018), Multi-Grid Chen et al. (2017); Fu et al. (2019); Chen et al. (2019) and Multi-Scale (MS) ensembling Chen et al. (2015);

Chen et al. (2018); Zhao et al. (2017); Yuan & Wang (2018); Dai et al. (2015). The contribution of each strategy is reported in Table 4.

Inference time We tested the average run time on the Cityscapes validation set with a Tesla V100. The Dilated FCN baseline and the Non-local model take 0.230s and 0.276s per image, respectively, while the proposed model uses 0.253s, which is only slightly higher than the baseline.

Effectiveness of different sampling rate φ and group of predicted weights G (Section 3.3 and 3.4 in main paper). For our experiments on Cityscapes, where are network has a stride of 8, the sampling rates are set to $\varphi = \{1, 6, 12, 24, 36\}$. For experiments on COCO, where the network stride is 32, we use smaller sampling rates of $\varphi = \{1, 4, 8, 12\}$ in C5. We keep the same sampling rate in C4 when DGMN modules are inserted into C4 as well. Without otherwise stated, we use $G = 4$ as default. Each group of C/G feature channels shares the same set of filter parameters Chollet (2017). The effect of different sampling rate and group of predicted filter weights are studied in Table 5 and Table 6.

Table 4: Ablation studies of different training and inference strategies. Our method (DGMN w/ DA+DW+US) is evaluated under single scale mIoU with ResNet-101 backbone on Cityscapes validation set.

	OHEM	Multi-grid	MS	mIoU (%)
FCN w/ DGMN	✘	✘	✘	79.2
FCN w/ DGMN	✔	✘	✘	79.7
FCN w/ DGMN	✔	✔	✘	80.3
FCN w/ DGMN	✔	✔	✔	**81.1**

Table 5: Quantitative analysis on different sampling rate of our dynamic sampling strategy in the proposed DGMN model on the Cityscapes validation set. We use a ResNet-101 as backbone. All methods are evaluated on the single scale mIoU.

	DA	DW	US	RWS	mIoU (%)
Dilated FCN	✘	✘	✘	✘	75.0
+ DGMN ($\varphi = \{1\}$)	✔	✘	✘	✘	76.5
+ DGMN ($\varphi = \{1\}$)	✔	✔	✘	✘	79.1
+ DGMN ($\varphi = \{1, 1, 1, 1, 1\}$)	✔	✔	✔	✘	79.2
+ DGMN ($\varphi = \{1, 6, 12\}$)	✔	✔	✔	✘	79.7
+ DGMN ($\varphi = \{1, 6, 12, 24, 36\}$)	✔	✔	✔	✘	80.3
+ DGMN ($\varphi = \{1, 6, 12, 24, 36\}$)	✔	✔	✘	✔	**80.4**

Table 6: Quantitative analysis on different group number and sampling rate of the proposed DGMN model on the COCO 2017 validation set. All methods are based on the Mask R-CNN detection pipeline with ResNet-50 backbone. Modules are inserted after all the 3×3 convolution layers of C5 (*res5*) of ResNet-50.

	DA	DW	US	RWS	AP^{box}	AP^{mask}
Mask R-CNN baseline	✘	✘	✘	✘	37.8	34.4
+ DGMN ($\varphi = \{1, 4, 8, 12\}$, $G = 0$)	✔	✘	✘	✘	39.4	35.6
+ DGMN ($\varphi = \{1, 4, 8, 12\}$, $G = 0$)	✔	✘	✘	✔	39.9	35.9
+ DGMN ($\varphi = \{1, 4, 8\}$, $G = 2$)	✔	✔	✘	✔	39.5	35.6
+ DGMN ($\varphi = \{1, 4, 8\}$, $G = 4$)	✔	✔	✘	✔	39.8	35.9
+ DGMN ($\varphi = \{1, 4, 8, 12\}$, $G = 4$)	✔	✔	✘	✔	**40.2**	**36.0**

Effectiveness of feature learning with DGMN on stronger backbones. We insert DGMN into deeper and more powerful backbone networks, such as ResNet 101 and ResNeXt 101 He et al. (2016); Xie et al. (2017). These results are shown in Table 7. By inserting DGMN at the convolutional stage

C5 of ResNet-101, DGMN (C5) outperforms the Mask R-CNN baseline with 1.6 points on the metric AP^{box} and 1.2 points on the metric AP^{mask}. On ResNeXt-101, DGMN (C5) also improves by 1.5 and 0.9 points on the AP^{box} and the AP^{mask}, respectively.

Table 7: Quantitative results via applying the proposed DGMN module into different strong backbone networks for object detection and instance segmentation on COCO 2017 validation set.

Model	Backbone	AP^{box}	AP^{box}_{50}	AP^{box}_{75}	AP^{mask}	AP^{mask}_{50}	AP^{mask}_{75}
Mask R-CNN baseline	ResNet-101	40.1	61.7	44.0	36.2	58.1	38.3
+ DGMN (C5)		**41.7**	**63.8**	**45.7**	**37.4**	**60.4**	**39.8**
Mask R-CNN baseline	ResNeXt-101	42.2	63.9	46.1	37.8	60.5	40.2
+ DGMN (C5)		**43.7**	**65.9**	**47.8**	**38.7**	**62.1**	**41.3**

A.4 State-of-art comparison on COCO

	Backbone	AP^{box}	AP^{box}_{50}	AP^{box}_{75}	AP^{mask}	AP^{mask}_{50}	AP^{mask}_{75}
One-stage detector							
YOLOv3 (Redmon & Farhadi, 2018)	Darknet-53	33.0	57.9	34.4	-	-	-
SSD513 (Liu et al., 2016)	ResNet-101-SSD	31.2	50.4	33.3	-	-	-
DSSD513 (Fu et al., 2017)	ResNet-101-DSSD	33.2	53.3	35.2	-	-	-
RetinaNet (Lin et al., 2017)	ResNeXt-101-FPN	40.8	61.1	44.1			
CornerNet (Law & Deng, 2018)	Hourglass-104	42.2	57.8	45.2			
Two-stage detector							
F-R-CNN+++ (He et al., 2016)	ResNet-101-C4	34.9	55.7	37.4	-	-	-
F-R-CNN w FPN (Lin et al., 2017)	ResNet-101-FPN	36.2	59.1	39.0	-	-	-
R-FCN (Dai et al., 2016)	ResNet-101	29.9	51.9	-	-	-	-
Mask RCNN (He et al., 2017)	ResNet-101-FPN	40.2	61.9	44.0	36.2	58.6	38.4
Mask RCNN (He et al., 2017)	ResNeXt-101-FPN	42.6	64.9	46.6	38.3	61.6	40.8
Libra R-CNN (Pang et al., 2019)	ResNetX-101-FPN	43.0	64.0	47.0	-	-	-
DGMN (ours)	ResNeXt-101-FPN	**44.3**	**66.8**	**48.4**	**39.5**	**63.3**	**42.1**

Table 8: Object detection and instance segmentation *single-model* performance on the COCO `test-dev`. We use *single scale* testing.