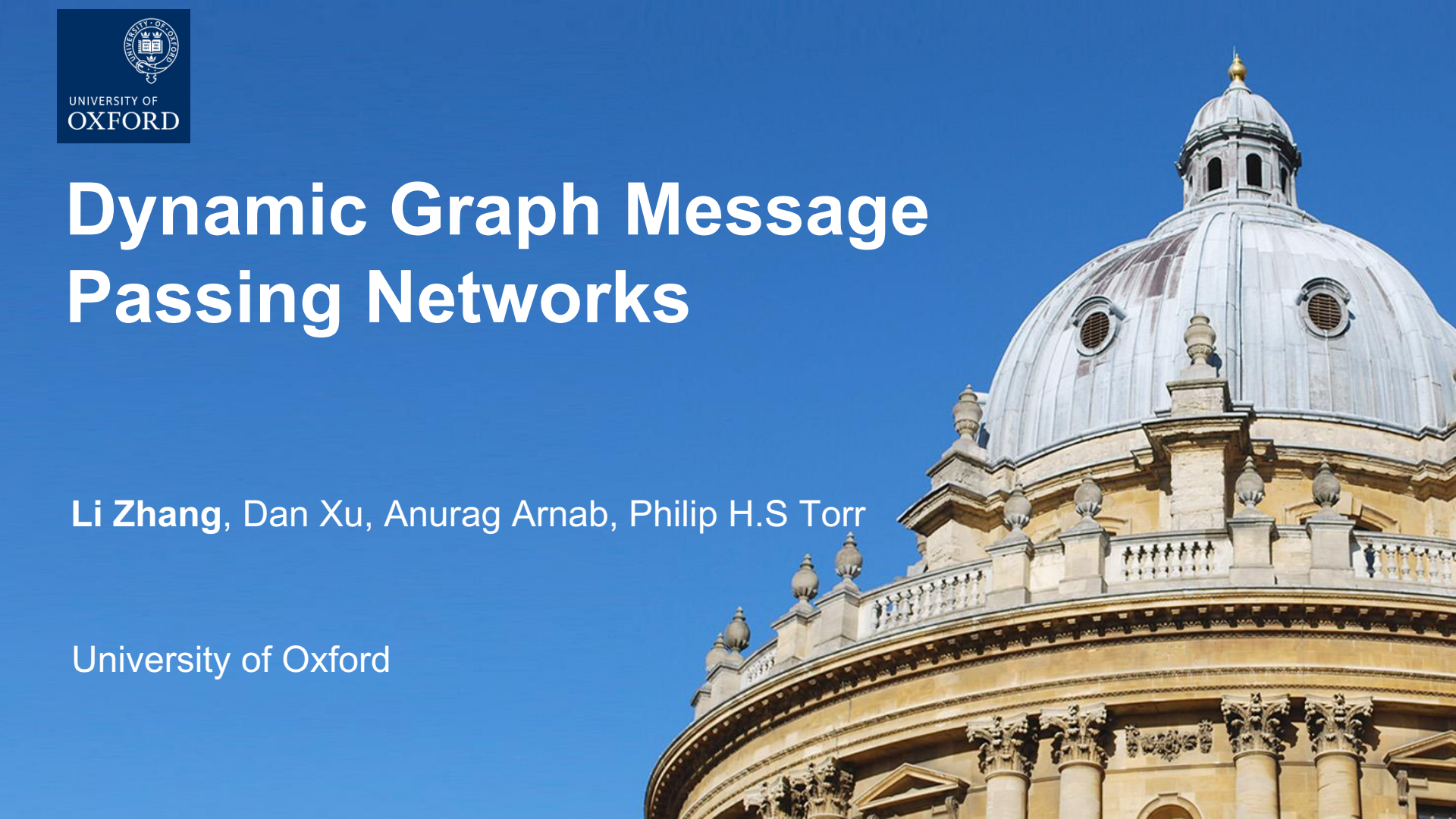


Dynamic Graph Message Passing Networks

Li Zhang, Dan Xu, Anurag Arnab, Philip H.S Torr

University of Oxford

Context in Object Recognition

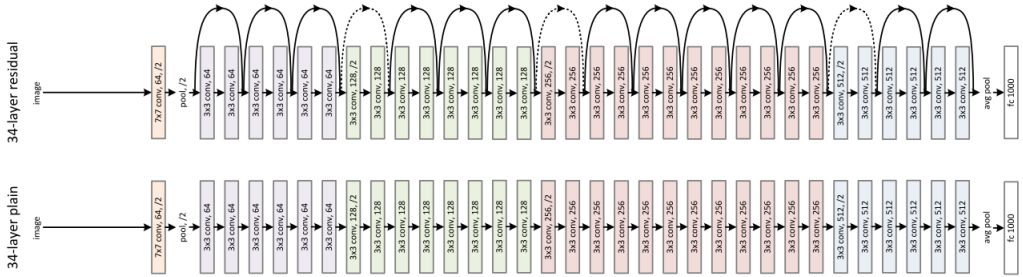
Label: ?

- Context is key for scene understanding tasks
- Understanding image patches in isolation is difficult.

Context in Object Recognition

Label: House?

- Context is key for scene understanding tasks
- Successive convolutional layers in CNNs increase the receptive field linearly.
- This is insufficient and inefficient

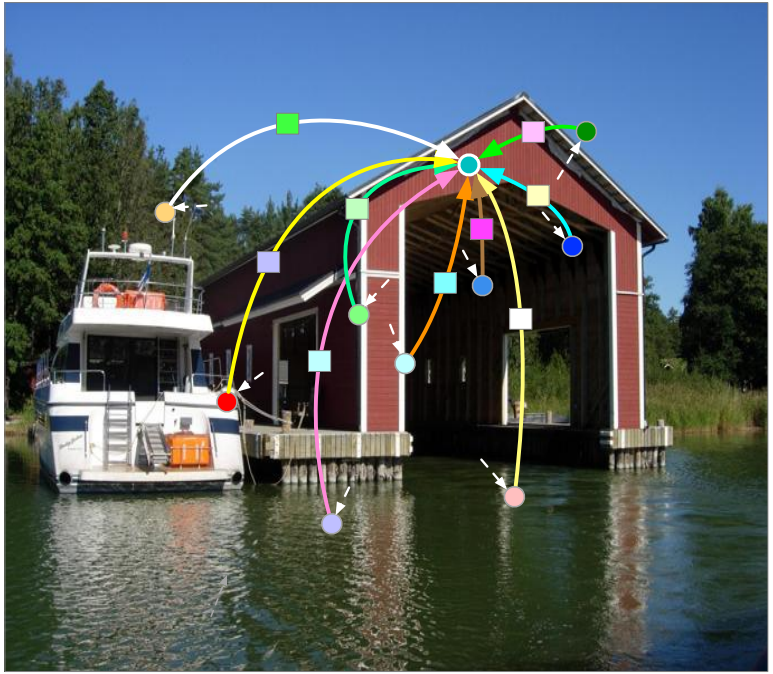


Context in Object Recognition

Label: Boathouse!

- Context is key for scene understanding tasks
- Dynamically sampling context from relevant regions of the image is accurate and efficient

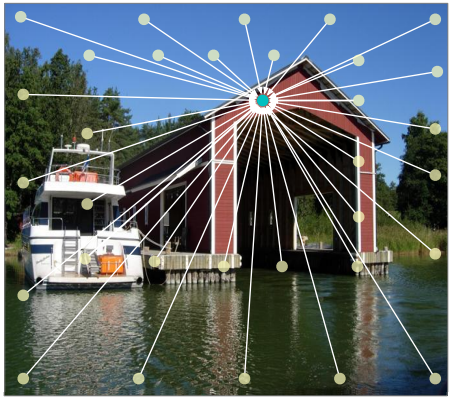
Dynamic Graph Message Passing



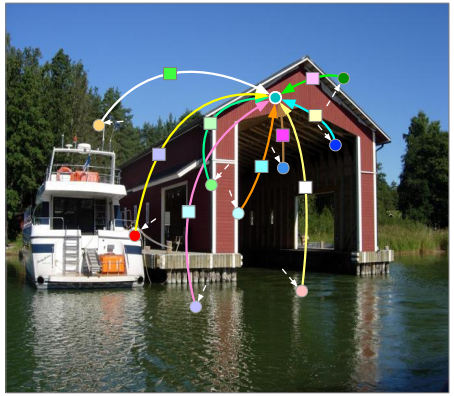
- Dynamically sample a small subset of relevant feature nodes
- Sampling scheme is learned and conditioned on the input
- Dynamically predict filter weights and affinities

Dynamic Graph Message Passing

Locally-connected



Fully-connected



DGMN

- Our model (DGMN) is more expressive than locally-connected models
- Significantly more efficient than fully-connected models (Non-local Networks, Wang *et al*, CVPR 2018)
- More accurate than both.

Graph Message Passing

UNIVERSITY OF OXFORD

$\mathbf{F} = \{\mathbf{f}_i\}_{i=1}^{N}$

Input observation feature

$\mathcal{G} = \{\mathcal{V}, \mathcal{E}, A\}$

Graph

$\mathbf{H} = \{\mathbf{h}_i\}_{i=1}^{N}$

Refined latent feature

Message calculation:

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

Message updating:

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

Graph Message Passing

$$\mathbf{F} = \{\mathbf{f}_i\}_{i=1}^{N} \longrightarrow \mathcal{G} = \{\mathcal{V}, \mathcal{E}, A\} \qquad \mathbf{H} = \{\mathbf{h}_i\}_{i=1}^{N}$$

Input observation feature | Graph | Refined latent feature

Message calculation:

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

Message updating:

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

Graph Message Passing

$$\mathbf{F} = \{\mathbf{f}_i\}_{i=1}^{N} \longrightarrow \mathcal{G} = \{\mathcal{V}, \mathcal{E}, A\} \longrightarrow \mathbf{H} = \{\mathbf{h}_i\}_{i=1}^{N}$$

Input observation feature | Graph | Refined latent feature

Message calculation:

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

Message updating:

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

Graph Message Passing

$$\mathbf{F} = \{\mathbf{f}_i\}_{i=1}^{N} \longrightarrow \mathcal{G} = \{\mathcal{V}, \mathcal{E}, A\} \longrightarrow \mathbf{H} = \{\mathbf{h}_i\}_{i=1}^{N}$$

Input observation feature | Graph | Refined latent feature

Message calculation:

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

Message updating:

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

Graph Message Passing

$$\mathbf{F} = \{\mathbf{f}_i\}_{i=1}^{N} \longrightarrow \mathcal{G} = \{\mathcal{V}, \mathcal{E}, A\} \longrightarrow \mathbf{H} = \{\mathbf{h}_i\}_{i=1}^{N}$$

Input observation feature | Graph | Refined latent feature

Message calculation:

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

Feature

Message updating:

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

Graph Message Passing

$$\mathbf{F} = \{\mathbf{f}_i\}_{i=1}^{N} \longrightarrow \mathcal{G} = \{\mathcal{V}, \mathcal{E}, A\} \longrightarrow \mathbf{H} = \{\mathbf{h}_i\}_{i=1}^{N}$$

Input observation feature | Graph | Refined latent feature

Message calculation:

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

Message updating:

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

Graph Message Passing

$$\mathbf{F} = \{\mathbf{f}_i\}_{i=1}^{N} \longrightarrow \mathcal{G} = \{\mathcal{V}, \mathcal{E}, A\} \longrightarrow \mathbf{H} = \{\mathbf{h}_i\}_{i=1}^{N}$$

Input observation feature | Graph | Refined latent feature

Message calculation:

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

Message updating:

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

Graph Message Passing

$$\mathbf{F} = \{\mathbf{f}_i\}_{i=1}^{N} \rightarrow \mathcal{G} = \{\mathcal{V}, \mathcal{E}, A\} \rightarrow \mathbf{H} = \{\mathbf{h}_i\}_{i=1}^{N}$$

Input observation feature | Graph | Refined latent feature

Message calculation:

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

Message updating:

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

Graph Message Passing

$$\mathbf{F} = \{\mathbf{f}_i\}_{i=1}^{N} \longrightarrow \mathcal{G} = \{\mathcal{V}, \mathcal{E}, A\} \longrightarrow \mathbf{H} = \{\mathbf{h}_i\}_{i=1}^{N}$$

Input observation feature | Graph | Refined latent feature

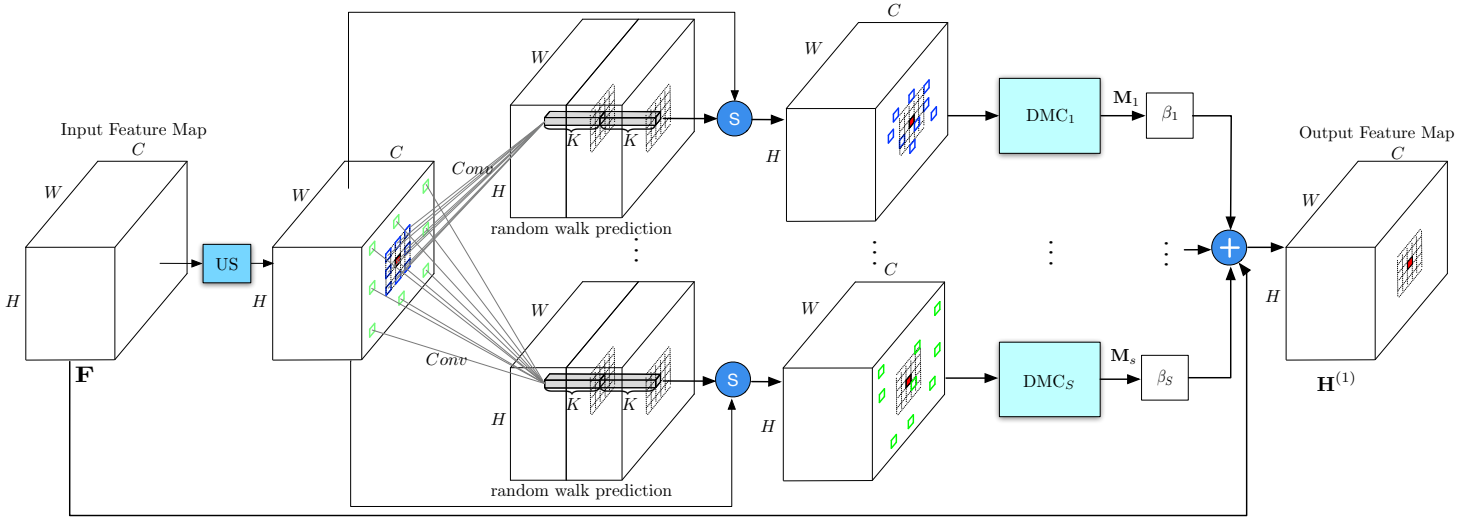
Message calculation:

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

Message updating:

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

Framework Overview of DGMN



- The neighbourhood used to update the feature representation of each node 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.

- Random walks are predicted (conditioned on the input) from these uniformly sampled nodes, denoted by the $\mathcal{S}$ symbol representing the random walk sampling operation.

Dynamic sampling nodes

Uniform sampling (US) with sampling rate: $\varphi = \{\rho_q\}_{q=1}^{S}$

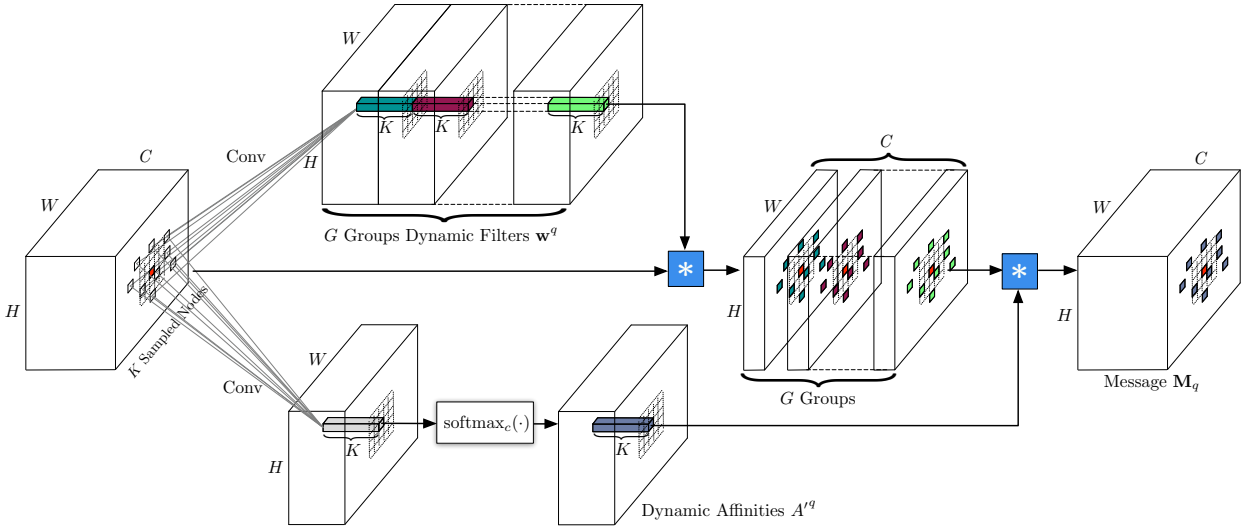
$$\mathbf{m}_i^{(t+1)} = \sum_q \sum_{j \in \mathcal{N}_q(i)} \beta_q A_{i,j}^q \mathbf{h}_j^{(t)} \mathbf{w}_j^q,$$

Random walk sampling, given a set of uniform sampling nodes:

$$\triangle \mathbf{d}_j^q = \mathbf{W}_{i,j}^q v_i^q + \mathbf{b}_{i,j}^q,$$

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

Framework Overview of DMC



- 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 x 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.

Dynamic filters and affinities

Joint learning of node-conditioned dynamic filters (DW) and affinities (DA) :

$$\{\mathbf{w}_j^q, {A'}_{i,j}^q\} = \mathbf{W}_{i,j}^{k,A} {v'}_i^q + \mathbf{b}_{i,j}^{k,A},$$

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

Modular instantiation

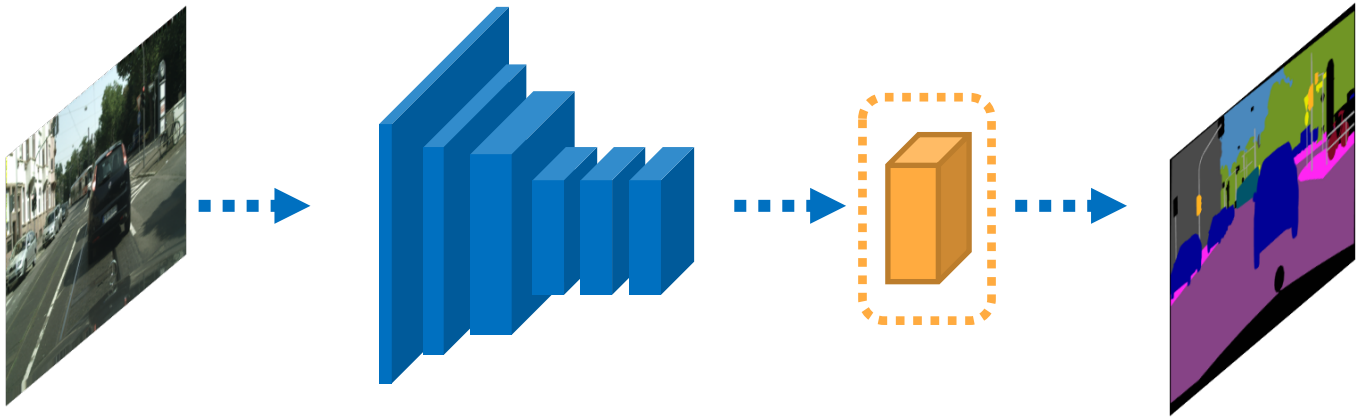
UNIVERSITY OF OXFORD

- We perform experiments on *semantic segmentation* on *Cityscapes*, and *object detection* / *instance segmentation* on *COCO*.

- Consider multiple configurations of our DGMN module:

  - Single module at the end of the network
  - Multiple modules inserted into the backbone

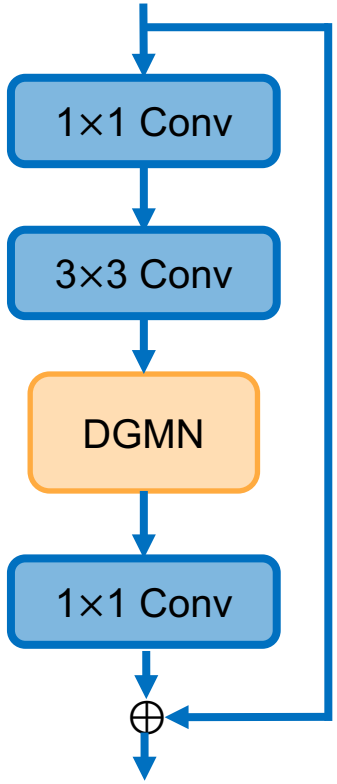
Modular instantiation: Semantic Segmentation



- Baseline is Dilated-FCN with a ResNet-101 backbone pretrained on ImageNet.
- Our DGMN module is randomly initialised and inserted between the 3 × 3 convolution layer and the final classifier.

Modular instantiation: Object Detection

- Baseline is Mask-RCNN with FPN and ResNet/ResNeXt backbone.
- We insert one or multiple randomly initialised DGMN modules into the backbone.
- We add our DGMN module after all 3 × 3 conv layers in a residual block.



Visualisation of node sampling

UNIVERSITY OF OXFORD

- Visualisation of nodes sampled by our network;
- Red point is the “receiving” node;
- Different color families show learned position-specific weights and affinities;
- Different colours in the same family show the sampled nodes with different sampling rates.

Results: Semantic Segmentation

Ablation of model components on Cityscapes validation set.

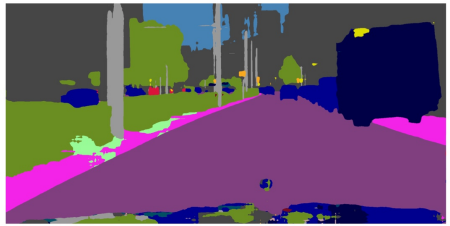
	mIoU (%)	Params	FLOPs
Dilated FCN [40]	75.0	–	–
+ Deformable [45]	78.2	+1.31M	+12.34G
+ ASPP [6]	78.9	+4.42M	+38.45G
+ Non-local [35]	79.0	+2.88M	+73.33G
+ DGMN w/ DA	76.5	+0.57M	+5.32G
+ DGMN w/ DA+DW	79.1	+0.73M	+6.88G
+ DGMN w/ DA+DW+DS	**80.4**	+2.61M	+24.55G

Results on the Cityscapes test set

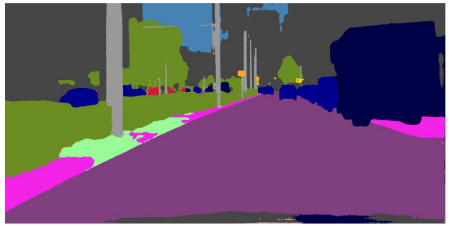
	Backbone	mIoU (%)
PSPNet [42]	ResNet 101	78.4
PSANet [43]	ResNet 101	80.1
DenseASPP [39]	DenseNet 161	80.6
GloRe [7]	ResNet 101	80.9
Non-local [35]	ResNet 101	81.2
CCNet [17]	ResNet 101	81.4
DANet [11]	ResNet 101	81.5
DGMN (Ours)	ResNet 101	**81.6**

Input

Dilated FCN



DGMN



Results: COCO Detection / Segmentation

UNIVERSITY OF OXFORD

Ablation on different network backbones on COCO 2017 val

	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**

Comparison on COCO val set.

	AP^{b}	AP^{b}_{50}	AP^{b}_{75}	AP^{m}	AP^{m}_{50}	AP^{m}_{75}
Mask R-CNN baseline	37.8	59.1	41.4	34.4	55.8	36.6
+ GCNet [4]	38.1	60.0	41.2	34.9	56.5	37.2
+ Deformable Message Passing	38.7	60.4	42.4	35.0	56.9	37.4
+ Non-local [35]	39.0	61.1	41.9	35.5	58.0	37.4
+ CCNet [17]	39.3	-	-	36.1	-	-
+ DGMN	**39.5**	**61.0**	**43.3**	**35.7**	**58.0**	**37.9**
+ GCNet (C5) [4]	38.7	61.1	41.7	35.2	57.4	37.4
+ Deformable (C5) [45]	39.9	-	-	34.9	-	-
+ DGMN (C5)	**40.2**	**62.0**	**43.4**	**36.0**	**58.3**	**38.2**

Mask R-CNN

Ours

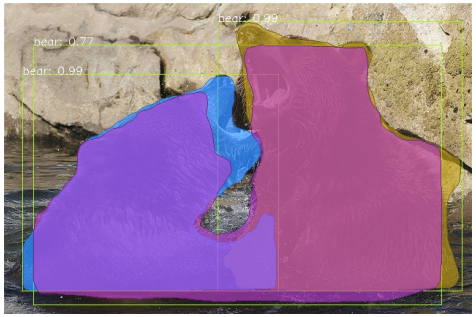
Results: COCO Detection / Segmentation

UNIVERSITY OF OXFORD

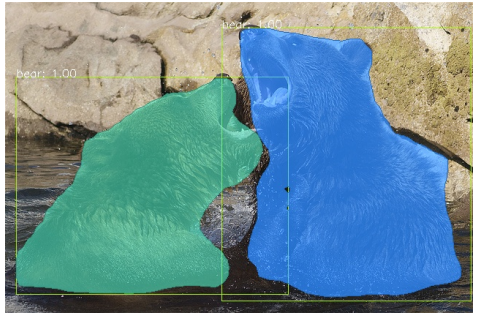
Comparison to state-of-the-art using a *single-model* on the COCO test-dev set.
We perform *single scale* testing.

	Backbone	AP^{box}	AP^{box}_{50}	AP^{box}_{75}	AP^{mask}	AP^{mask}_{50}	AP^{mask}_{75}
One-stage detectors							
YOLOv3 [44]	Darknet-53	33.0	57.9	34.4	-	-	-
SSD513 [38]	ResNet-101-SSD	31.2	50.4	33.3	-	-	-
DSSD513 [14]	ResNet-101-DSSD	33.2	53.3	35.2	-	-	-
RetinaNet [35]	ResNeXt-101-FPN	40.8	61.1	44.1			
CornerNet [28]	Hourglass-104	42.2	57.8	45.2			
Two-stage detectors							
Faster R-CNN+++ [20]	ResNet-101-C4	34.9	55.7	37.4	-	-	-
Faster R-CNN w FPN [34]	ResNet-101-FPN	36.2	59.1	39.0	-	-	-
R-FCN [12]	ResNet-101	29.9	51.9	-	-	-	-
Mask R-CNN [19]	ResNet-101-FPN	40.2	61.9	44.0	36.2	58.6	38.4
Mask R-CNN [19]	ResNeXt-101-FPN	42.6	64.9	46.6	38.3	61.6	40.8
Libra R-CNN [41]	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**

Mask R-CNN



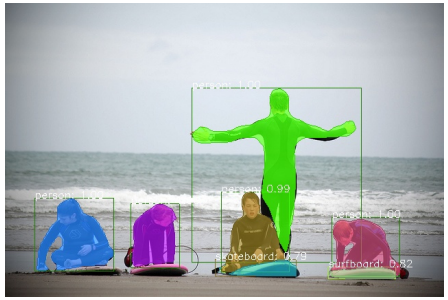
Ours

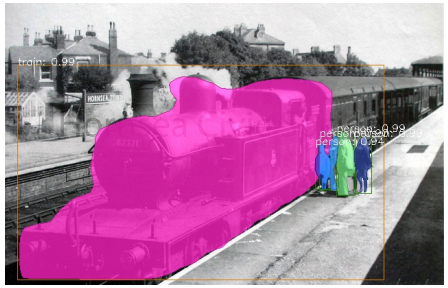


Thank you! Questions?

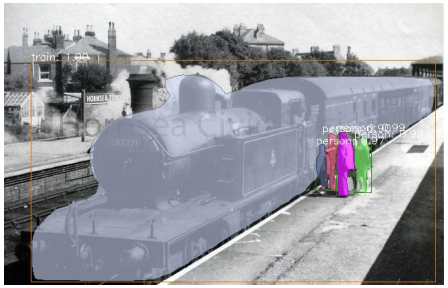
UNIVERSITY OF OXFORD

Mask-RCNN



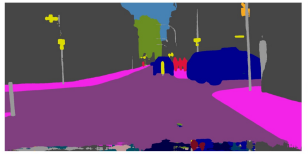


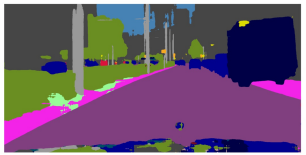
Ours



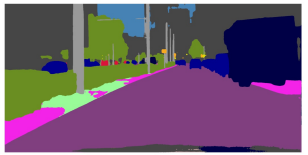
Input

Dilated FCN





DGMN



www.robots.ox.ac.uk/~lz/dgmn