Recurrent Space-time Graph Neural Networks

Andrei Nicolicioiu | Iulia Duță | Marius Leordeanu

NeurIPS 2019

Introduction

- spatial interactions
- temporal interactions

Introduction

- spatial interactions
- temporal interactions

Spatio-temporal processing

- we want to design models that implicitly take advantage of known biases in the data

Spatio-temporal processing

- we want to design models that implicitly take advantage of known biases in the data
    - **locality assumption**: bias towards local interactions

Spatio-temporal processing

- we want to design models that implicitly take advantage of known biases in the data
  - **locality assumption**: bias towards local interactions
  - **long-range assumption**: distant entities interactions could contribute in a significant way

Spatio-temporal processing

- we want to design models that implicitly take advantage of known biases in the data
  - **locality assumption**: bias towards local interactions
  - **long-range assumption**: distant entities interactions could contribute in a significant way
  - **stationarity assumption**: interactions are the same at every position in the scene

Graph methods

- **graph models** satisfy these assumptions

[1] [Duvenaud et al. [2015]], [Battaglia et al. [2016]], [Kipf and Welling [2017]]

Graph methods

- **graph models** satisfy these assumptions
- structure information as a graph:
  - **nodes** represent **regions** in video
  - **edges** represent **interactions** between nodes

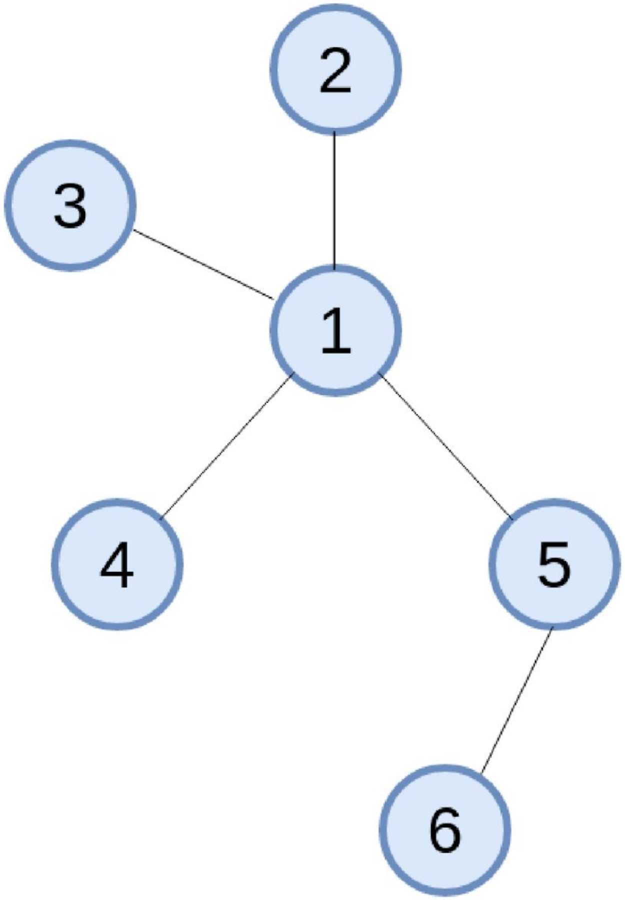
[1] [Duvenaud et al. [2015]], [Battaglia et al. [2016]], [Kipf and Welling [2017]]

Graph methods

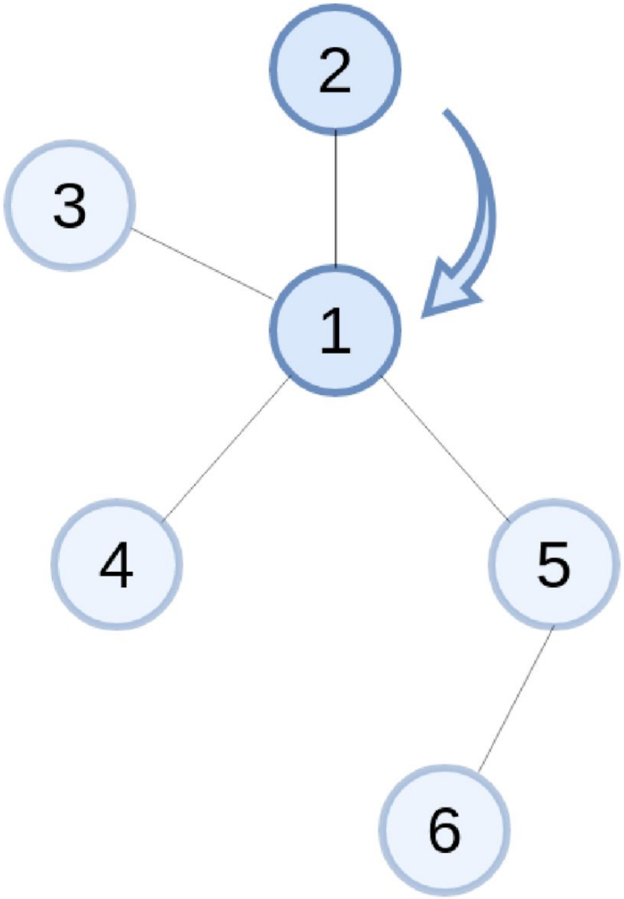
- **graph models** satisfy these assumptions
- structure information as a graph:
  - **nodes** represent **regions** in video
  - **edges** represent **interactions** between nodes
- graph models follow a general **message passing** framework[1]

[1] [Duvenaud et al. [2015]], [Battaglia et al. [2016]], [Kipf and Welling [2017]]

Message passing: General framework



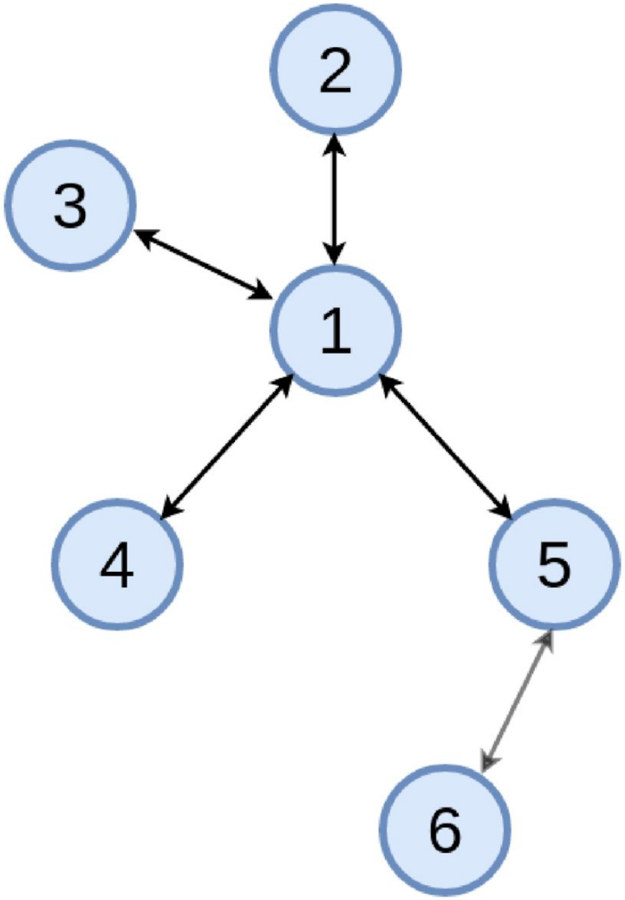
Message passing: General framework



1. **send** messages between neighbours

$$f_{send}(v_i^t, v_j^t, e_{ij}) \tag{1}$$

Message passing: General framework



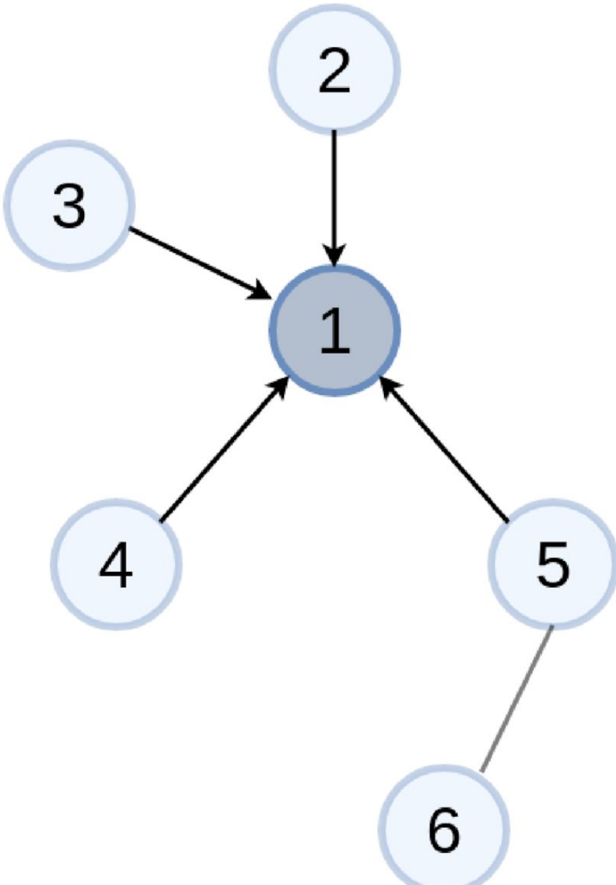
1. **send** messages between neighbours

$$f_{send}(v_i^t, v_j^t, e_{ij}) \tag{1}$$

2. **gather** messages from neighbourhood

$$m_i^{t+1} = \sum_{w \in \mathcal{N}(i)} M_t(v_i^t, v_j^t, e_{ij}) \tag{2}$$

Message passing: General framework



1. **send** messages between neighbours

$$f_{send}(v_i^t, v_j^t, e_{ij}) \quad (1)$$

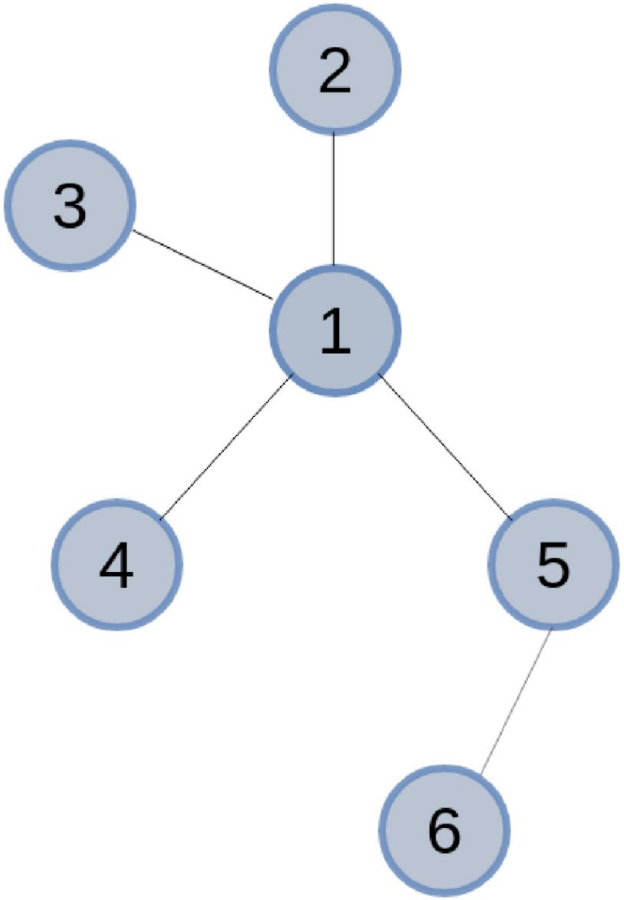
2. **gather** messages from neighbourhood

$$m_i^{t+1} = \sum_{w \in \mathcal{N}(i)} M_t(v_i^t, v_j^t, e_{ij}) \quad (2)$$

3. **update** each node with received info

$$v_i^{t+1} = f_{update}(v_i^t, m_i^{t+1}) \quad (3)$$

Message passing: General framework



1. **send** messages between neighbours

$$f_{send}(v_i^t, v_j^t, e_{ij}) \tag{1}$$

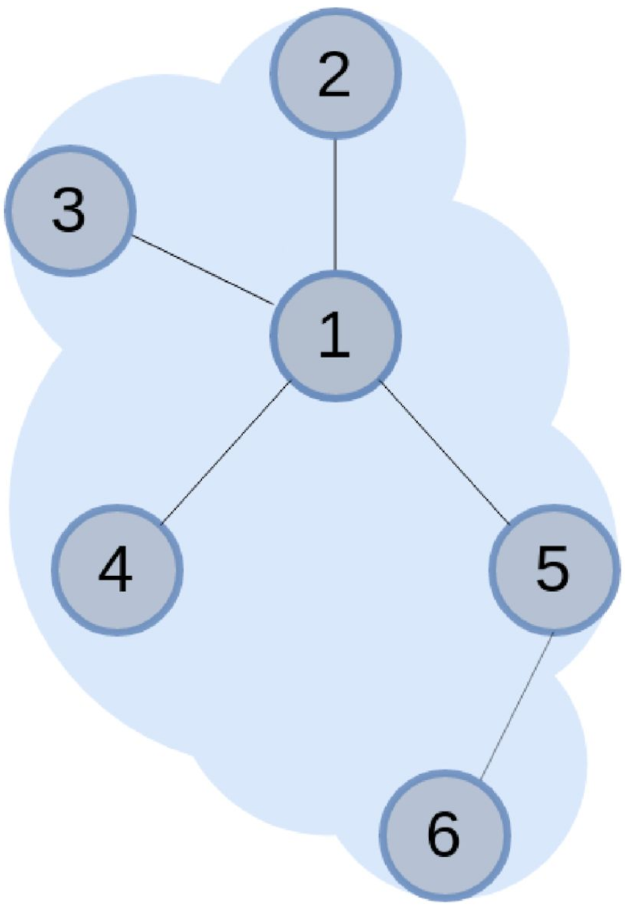
2. **gather** messages from neighbourhood

$$m_i^{t+1} = \sum_{w \in \mathcal{N}(i)} M_t(v_i^t, v_j^t, e_{ij}) \tag{2}$$

3. **update** each node with received info

$$v_i^{t+1} = f_{update}(v_i^t, m_i^{t+1}) \tag{3}$$

Message passing: General framework



1. **send** messages between neighbours

$$f_{send}(v_i^t, v_j^t, e_{ij}) \tag{1}$$

2. **gather** messages from neighbourhood

$$m_i^{t+1} = \sum_{w \in \mathcal{N}(i)} M_t(v_i^t, v_j^t, e_{ij}) \tag{2}$$

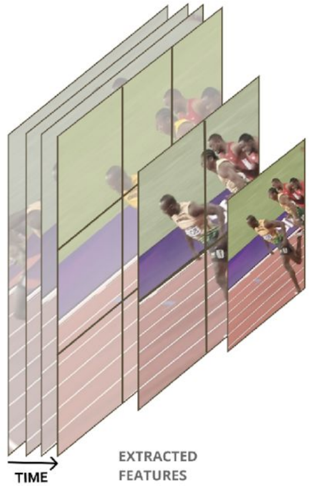
3. **update** each node with received info

$$v_i^{t+1} = f_{update}(v_i^t, m_i^{t+1}) \tag{3}$$

4. **aggregate** the whole graph

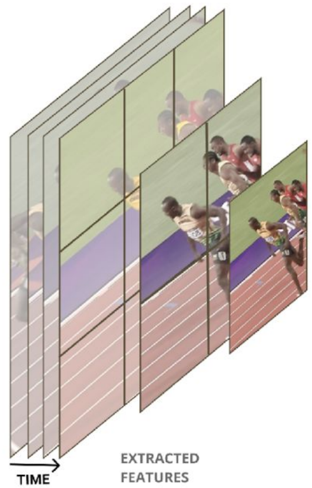
$$y = R(v_i^T | v \in G) \tag{4}$$

Overview RSTG



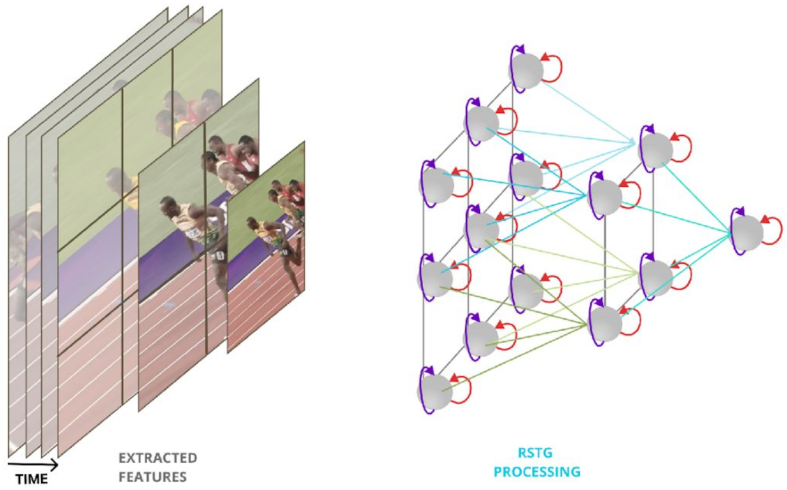
- we propose a neural graph model, **recurrent** in **space** and **time**

Overview RSTG



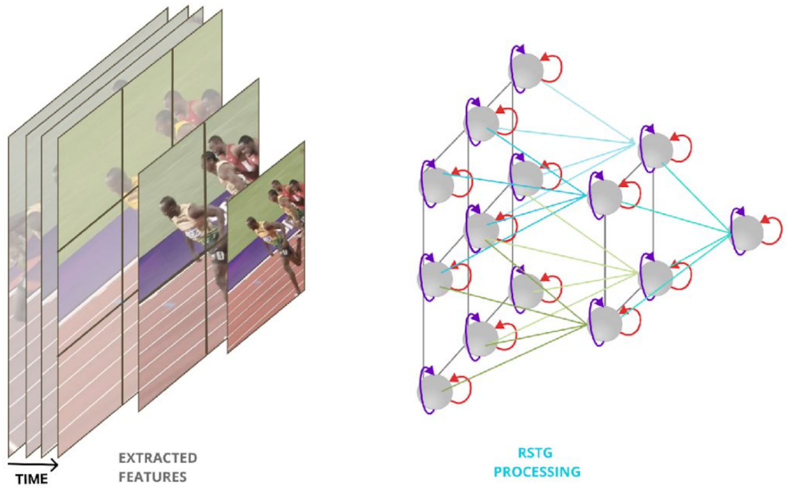
- we propose a neural graph model, **recurrent** in **space** and **time**
- extract video features using backbone model

Overview RSTG



- we propose a neural graph model, **recurrent** in **space** and **time**
- extract video features using backbone model
- create graph with information from video features

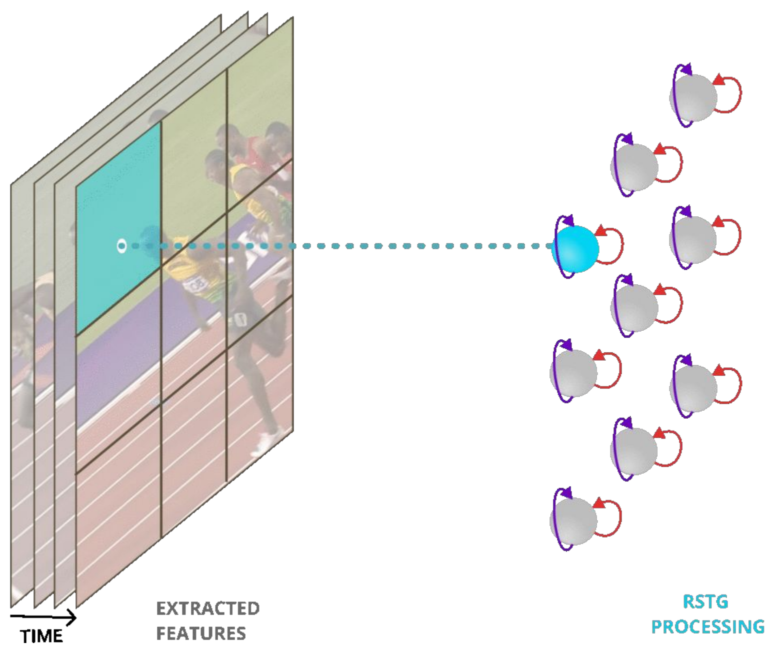
Overview RSTG



- we propose a neural graph model, **recurrent** in **space** and **time**
- extract video features using backbone model
- create graph with information from video features
- process video by message-passing to get long range interactions

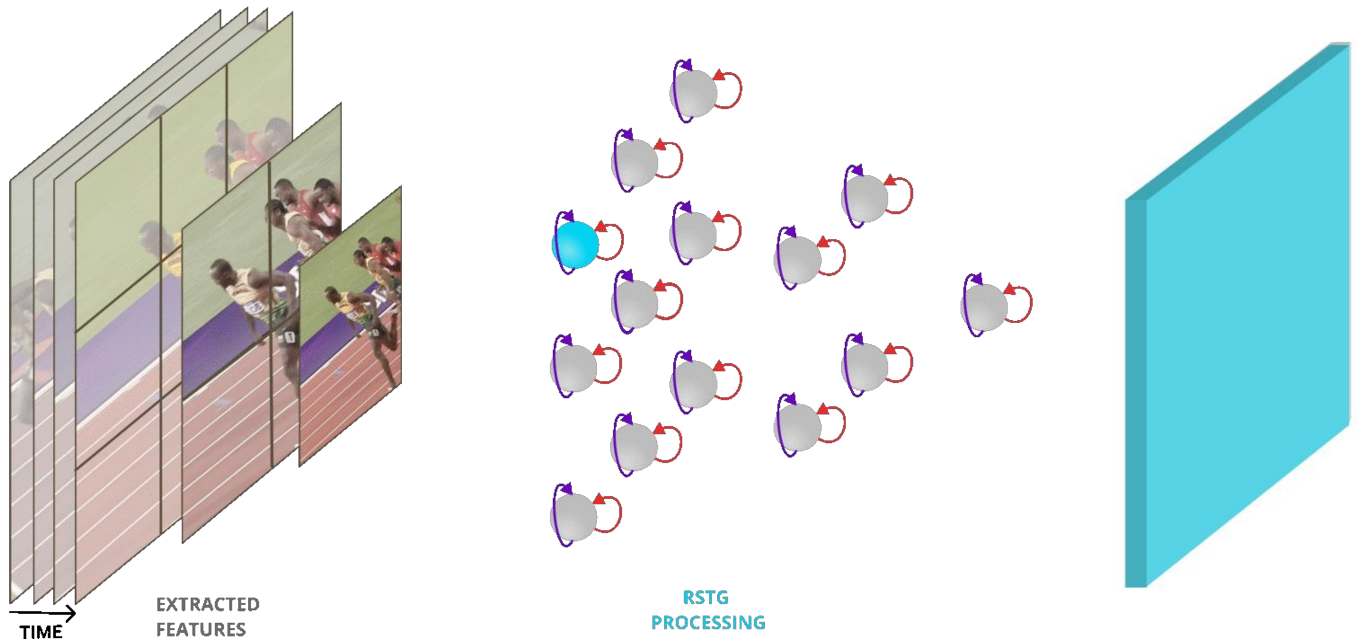
Graph Creation - Nodes

- use features maps from a pretrained 2D / 3D **backbone**
- use feature at different **scales**
- each node receives info **pooled** from a region



Graph Creation - Edges

- the nodes are **connected** if:
  - they are **neighbours** in the grid
  - their corresponding regions **overlap**
- thus we have a **sparse** graph



Graph Processing

- for video understanding we should model interaction:
  - between entities from **different regions** (space)
  - between entities at **different time steps** (time)

Graph Processing

- for video understanding we should model interaction:
  - between entities from **different regions** (space)
  - between entities at **different time steps** (time)

- we factorise our processing in two separate stages:
  - **Space Processing Stage**: captures frame level information
  - **Time Processing Stage**: captures information across time

Space Processing Stage

- model **spatial interactions** by exchanging messages
- the process involves 3 steps:
  - **send** messages between all connected nodes
  - **gather** information at each node
  - **update** internal node representation

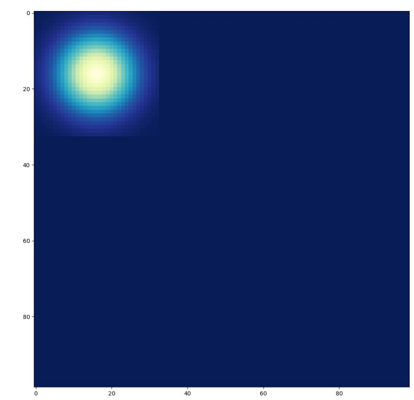
Space Processing Stage - Send

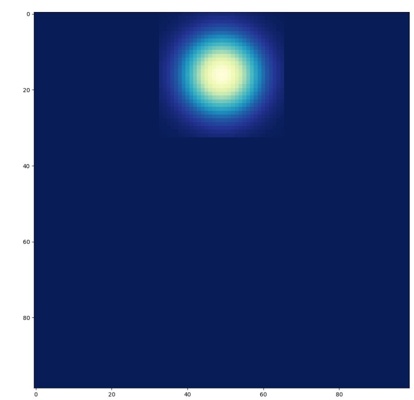
- **send**:
  - message should represent pairwise interaction
  - message is a function of both source and destination
  - the function is implemented as an MLP

$$f_{send}(\mathbf{v}_j, \mathbf{v}_i) = \mathrm{MLP}_s([\mathbf{v}_j|\mathbf{v}_i]) \in \mathbb{R}^D. \tag{5}$$

Space Processing Stage - Position Awareness

- be **aware** of nodes position
- use both nodes position as input of f_{send}
- position is a gaussian centered in node location





Space Processing Stage - Gather & Update

- **gather**:
  - aggregate messages by an attention mechanism
  - use dot product as features similarity

$$f_{gather}(\mathbf{v}_i) = \sum_{j \in \mathcal{N}(i)} \alpha(\mathbf{v}_j, \mathbf{v}_i) f_{send}(\mathbf{v}_j, \mathbf{v}_i) \in \mathbb{R}^D. \tag{6}$$

$$\alpha(\mathbf{v}_j, \mathbf{v}_i) = (W_{\alpha_1}\mathbf{v}_j)^T (W_{\alpha_2}\mathbf{v}_i) \in \mathbb{R}. \tag{7}$$

- **update**:
  - incorporate global context into each local information

$$f_{space}(\mathbf{v}_i) = \mathrm{MLP}_u([\mathbf{v}_i | f_{gather}(\mathbf{v}_i)]) \in \mathbb{R}^D. \tag{8}$$

Time Processing Stage

- node: current spatial info + previous time step info
- update uses a **recurrent** function

- for more expressive power we alternate stages
- K alternating stages + a final time stage

$$\mathbf{h}_{i,time}^{t,k} = f_{time}(\mathbf{v}_{i,space}^{k}, \mathbf{h}_{i,time}^{t-1,k}). \tag{9}$$

Scheduler

Scheduler

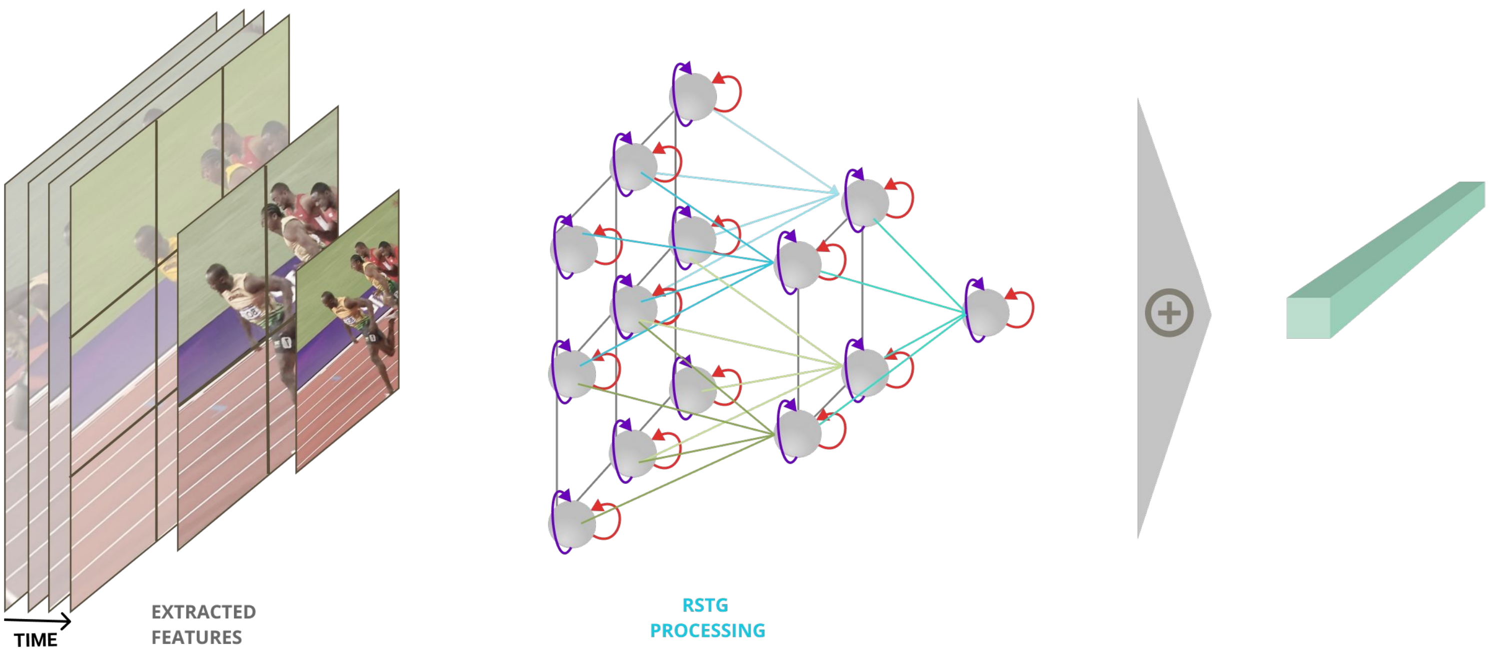
Scheduler

RSTG for Video Processing

- input: $T \times H \times W \times C$ feature maps
- two types of output:
- **RSTG-to-vec**:
  - a global **vectorial** representation of the video
- **RSTG-to-map**:
  - a feature **map** further used by spatio-temporal models

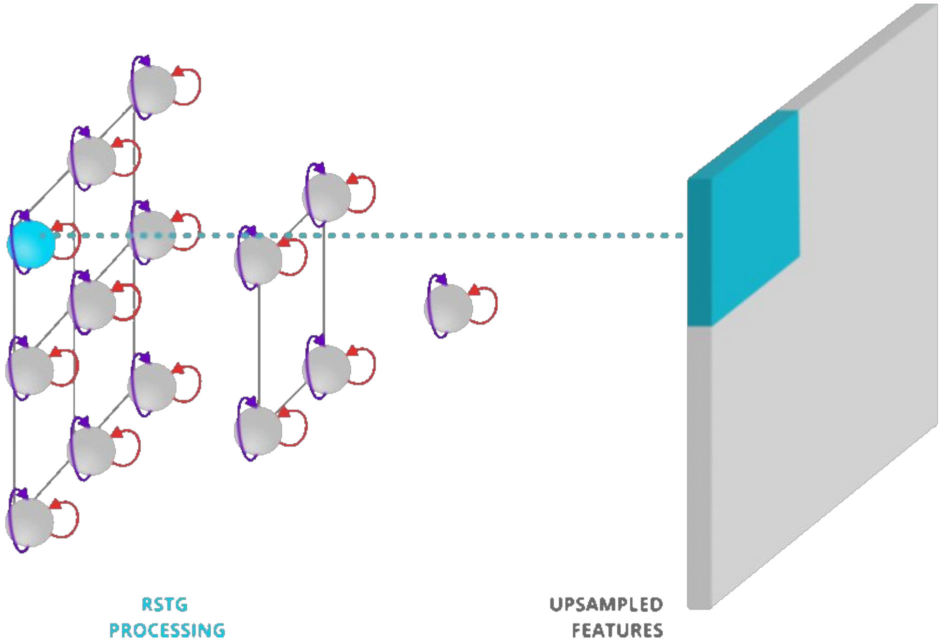
RSTG for Video Processing: RSTG-to-vec

- obtain a **vector** used for the final classification
- use the nodes information from the **final temporal step**
- **sum** all the nodes into a global representation



RSTG for Video Processing: RSTG-to-map

- obtain **3D maps** representation further processed with spatio-temporal models
- **symetric** operation to the graph creation
- for each time step we **project** the nodes into their corresponding region of the map
- **sum** the maps given by multiple scales



SyncMNIST Dataset

- involves challenging relationships in space and time
- from a set of randomly **moving digits** find the pair that moves **synchronous**
- 2 variants: 3SyncMNIST and 5SyncMNIST

Random

Sync pair - (4,2)

Results on SyncMNIST: Ablation

We change parts of our model to investigate their contributions:

- **Space-Only**: mean-pooling as Time Processing Stage
- **Time-Only**: mean-pooling as Space Processing Stage
- **Homogeneous**: use the same update function in space and time
- **1-temp-stage**: just one final Time Processing Stage
- **All-temp-stages**: interleaved stages
- **Positional All-temp**: full model with positional embeddings

Table: Accuracy on SyncMNIST dataset, showing the capabilities of different parts of our model.

Model	3SyncMNIST	5SyncMNIST
RSTG: Space-Only	61.3	-
RSTG: Time-Only	89.7	-
RSTG: Homogenous	95.7	58.3
RSTG: 1-temp-stage	97.0	74.1
RSTG: All-temp-stages	**98.9**	94.5
RSTG: Positional All-temp	-	**97.2**

Results on SyncMNIST

Table: Accuracy on SyncMNIST dataset compared against powerful baselines

Model	3 SyncMNIST	5 SyncMNIST
Mean + LSTM	77.0	-
Conv + LSTM	95.0	39.7
I3D [Carreira and Zisserman [2017]]	-	90.6
Non-Local [Wang et al. [2018]]	-	93.5
RSTG: All-temp-stages	**98.9**	94.5
RSTG: Positional All-temp	-	**97.2**

Results on Something-Something v1

- Something-Something-v1: real world scenario involving complex interactions
- 174 classes for fine-grained human-objects interactions

“Lifting up one end of something **without** letting it drop down”

“Lifting up one end of something, then letting it drop down”

Something-Something v1 - Backbone

- two types of backbone:
  - **C2D**:
    - process each frame individually using 2D ConvNet
    - use ResNet-50 pretrained on Kinetics dataset
  - **I3D**:
    - local spatio-temporal processing using 3D ConvNet
    - use I3D [Carreira and Zisserman [2017]] inflated from ResNet-50, pretrained on Kinetics dataset

Something-Something v1: Ablation

Table: Ablation study showing where to place the graph inside the I3D backbone.

Model	Top-1	Top-5
RSTG-to-vec	47.7	77.9
RSTG-to-map res2	46.9	76.8
RSTG-to-map res3	47.7	77.8
RSTG-to-map res4	48.4	78.1
RSTG-to-map res3-4	**49.2**	**78.8**

Table: RSTG-to-map res4

model	**layer**
	input
I3D	conv1
	pool1
	res2
	pool2
	res3
	res4
RSTG	Graph creation
	$\begin{bmatrix} \text{Temporal Processing Stage} \\ \text{Spatial Processing Stage} \end{bmatrix} \times 3$ Temporal Proctage
	Up-sample each grid $1 \times 1 \times 1$ conv
I3D	**res5**
	mean pool, fc

Results on Something-Something v1

Table: Top-1 and Top-5 accuracy on Something-Something-v1 on validation split.

Model	Backbone	Top-1	Top-5
C2D	2D ResNet-50	31.7	64.7
TRN [Zhou et al. [2018]]	2D Inception	34.4	-
ours C2D + RSTG	2D ResNet-50	**42.8**	**73.6**
MFNet-C50 [Lee et al. [2018]]	3D ResNet-50	40.3	70.9
I3D [Wang and Gupta [2018]]	3D ResNet-50	41.6	72.2
NL I3D [Wang and Gupta [2018]]	3D ResNet-50	44.4	76.0
NL I3D + GCN [Wang and Gupta [2018]]	3D ResNet-50	46.1	76.8
ECO-Lite 16F [Zolfaghari et al. [2018]]	2D Inc+3D Res-18	42.2	-
MFNet-C101 [Lee et al. [2018]]	3D ResNet-101	43.9	73.1
I3D [Xie et al. [2018]]	3D Inception	45.8	76.5
S3D-G [Xie et al. [2018]]	3D Inception	48.2	78.7
ours I3D + RSTG	3D ResNet-50	**49.2**	**78.8**

Conclusion

- we propose a **novel computational model** for learning in spatio-temporal domain with a graph model **recurrently** in both dimensions

Conclusion

- we propose a **novel computational model** for learning in spatio-temporal domain with a graph model **recurrently** in both dimensions

- we **factorize space and time** and process them differently, achieving low computational complexity

Conclusion

- we propose a **novel computational model** for learning in spatio-temporal domain with a graph model **recurrently** in both dimensions
- we **factorize space and time** and process them differently, achieving low computational complexity
- we **introduce a new synthetic dataset**, with complex interactions

Conclusion

- we propose a **novel computational model** for learning in spatio-temporal domain with a graph model **recurrently** in both dimensions
- we **factorize space and time** and process them differently, achieving low computational complexity
- we **introduce a new synthetic dataset**, with complex interactions
- we obtain **state-of-the-art results** on the challenging Something-Something dataset

References I

P. Battaglia, R. Pascanu, M. Lai, D. J. Rezende, et al. Interaction networks for learning about objects, relations and physics. In *Advances in neural information processing systems*, pages 4502–4510, 2016.

J. Carreira and A. Zisserman. Quo vadis, action recognition? a new model and the kinetics dataset. In *Computer Vision and Pattern Recognition (CVPR), 2017 IEEE Conference on*, pages 4724–4733. IEEE, 2017.

D. K. Duvenaud, D. Maclaurin, J. Iparraguirre, R. Bombarell, T. Hirzel, A. Aspuru-Guzik, and R. P. Adams. Convolutional networks on graphs for learning molecular fingerprints. In *Advances in neural information processing systems*, pages 2224–2232, 2015.

T. N. Kipf and M. Welling. Semi-supervised classification with graph convolutional networks. In *International Conference on Learning Representations (ICLR)*, 2017.

References II

M. Lee, S. Lee, S. J. Son, G. Park, and N. Kwak. Motion feature network: Fixed motion filter for action recognition. In *ECCV*, 2018.

X. Wang and A. Gupta. Videos as space-time region graphs. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 399–417, 2018.

X. Wang, R. Girshick, A. Gupta, and K. He. Non-local neural networks. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, volume 1, page 4, 2018.

S. Xie, C. Sun, J. Huang, Z. Tu, and K. Murphy. Rethinking spatiotemporal feature learning: Speed-accuracy trade-offs in video classification. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 305–321, 2018.

References III

B. Zhou, A. Andonian, A. Oliva, and A. Torralba. Temporal relational reasoning in videos. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 803–818, 2018.

M. Zolfaghari, K. Singh, and T. Brox. Eco: Efficient convolutional network for online video understanding. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 695–712, 2018.