Multi-Task Edge Prediction in Temporally-Dynamic Video Graphs

Osman Ülger[1]
o.ulger@uva.nl
Julian Wiederer[2]
julian.wiederer@mercedes-benz.com
Mohsen Ghafoorian
mohsenghafoorian@gmail.com
Vasileios Belagiannis[3]
vasileios.belagiannis@fau.de
Pascal Mettes[1]
p.s.m.mettes@uva.nl

[1] University of Amsterdam
Amsterdam, NL
[2] Mercedes-Benz
Stuttgart, DE
[3] Friedrich-Alexander-Universität
Erlangen-Nürnberg,
Erlangen, DE

Abstract

Graph neural networks have shown to learn effective node representations, enabling node-, link-, and graph-level inference. Conventional graph networks assume static relations between nodes, while relations between entities in a video often evolve over time, with nodes entering and exiting dynamically. In such temporally-dynamic graphs, a core problem is inferring the future state of spatio-temporal edges, which can constitute multiple types of relations. To address this problem, we propose MTD-GNN, a graph network for predicting temporally-dynamic edges for multiple types of relations. We propose a factorized spatio-temporal graph attention layer to learn dynamic node representations and present a multi-task edge prediction loss that models multiple relations simultaneously. The proposed architecture operates on top of scene graphs that we obtain from videos through object detection and spatio-temporal linking. Experimental evaluations on ActionGenome and CLEVRER show that modeling multiple relations in our temporally-dynamic graph network can be mutually beneficial, outperforming existing static and spatio-temporal graph neural networks, as well as state-of-the-art predicate classification methods. Code is available at https://github.com/ozzyou/MTD-GNN.

1 Introduction

Graph neural networks (GNN) have become an established framework for visual recognition and understanding, with applications such as action recognition [26, 45, 49, 57], semantic segmentation [29, 37], and visual relation detection [32]. A common assumption in GNNs is that nodes are stationary or at least always present. In practice, however, especially in the video domain, visual relations evolve dynamically over time. Entities, modeled as graph nodes, can enter or exit scenes, while edges have evolving semantics. In this paper, we address the problem of predicting edge labels in dynamically-evolving spatio-temporal graphs.

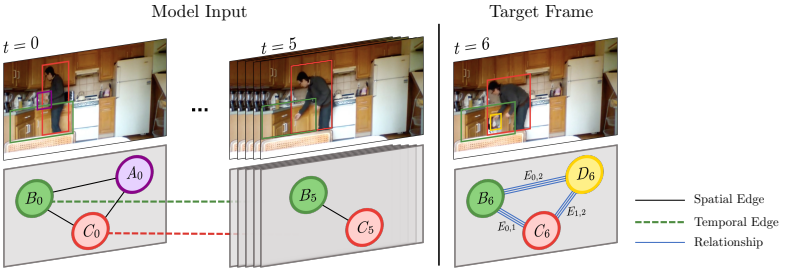

Figure 1: **Problem setting.** The model input is a spatio-temporal graph $\mathcal{G}_{\text{input}}$, built from object detections in multiple, adjacent frames. Visual feature representations of detections are modeled as graph nodes which can enter and exit scenes over time. Our approach is able to handle such dynamic graph changes. We predict relationships between objects in the target frame, which follows the last input frame. Based on the input, our model is able to predict multiple relation types - pictured are three - depicted with $E_{i,j}$ for objects i and j.

Multiple works have investigated spatio-temporal graph neural networks dealing with the temporal dynamics of graphs, for example for skeleton-based action recognition [24, 49] and object-action relations in video [36, 63]. While such networks incorporate temporal dynamics, there is typically one set of entities that remains present in the scene across time. In contrast, we are interested in a more challenging setting where entities may enter and/or exit the scene over time. Such a setting is relevant for real-world applications such as autonomous driving [40, 46]. Fig. 1 illustrates our setup. We propose a graph network that does not rely on stationary node assumptions and enables learning multiple relations simultaneously.

In this work, we introduce the task of future state multi-relational edge label prediction in temporally-dynamic graphs. Moreover, we introduce a Multi-task Temporally-Dynamic Graph Neural Network (MTD-GNN), a graph network centered around a factorized spatio-temporal graph attention layer as a natural solution to learn with dynamic node sets in both space and time, inspired by static graph attention [43]. On top of the graph attention layers, MTD-GNN learns multiple relation types as a weighted multi-task optimization. The graph network is learned on top of spatio-temporal interaction graphs constructed through detection and temporal linking. Experiments on CLEVRER [52] and Action Genome [12] demonstrate that our approach can handle temporally-dynamic spatio-temporal scene graphs, while also outperforming most existing static, as well as state-of-the-art methods. Additionally, we find that learning in a multi-task manner can boost the model's performance on individual tasks.

2 Related Work

Static graph networks. Graph neural networks denote a family of representation learning algorithms on graph-structured data. Graph networks learn node-level representations while abiding by the permutation invariant nature of graphs. Well-known instantiations of graph neural networks include the original Graph Neural Network by Scarselli *et al.* [39], Graph Convolutional Networks [21], and Graph Attention Networks [43]. Commonly, each node aggregates information from its neighbours within a graph layer, accompanied by a shared weight matrix. By stacking multiple graph layers, node representations are learned using information from nodes throughout the graph. Given learned node representations from

stacked graph layers, inference can be performed on nodes [9, 58, 62], links [2, 12, 50, 60, 61], and graphs [4, 8, 23]. Close to our approach are works on link prediction [1, 18] in graph neural networks [34], *e.g.* for recommendation [6, 22, 41]. For example, Wang *et al.* [44] use an attention mechanism in user-item graphs to predict user recommendations as links. A number of works have also investigated multiple relation types between nodes in graph networks [9, 27]. While these works target static multi-relational learning, we focus on multi-relational learning in temporally-dynamic graphs. The work of Kipf *et al.* [20] uses edges between objects to model their latent interactions as a useful encoding to predict object dynamics. This example shows the expressive power of edges for downstream tasks. Closest to our work, Kim *et al.* [17] follow a more explicit formulation of edges. They use an edge-labeling graph neural network with edges representing assignments to clusters in supervised and semi-supervised image classification. The mentioned methods are designed for static graphs and therefore rely on a constant number of input nodes. Similarly, we seek to label edges. However, rather than inferring on edges in static graphs, we do so in temporally-dynamic spatio-temporal graphs, where nodes and relations evolve over time.

Spatio-temporal graph networks. Multiple works have investigated extensions of graph networks to the spatio-temporal domain, for tasks such as activity recognition [11, 24, 49] and traffic forecasting [3, 5, 55]. Spatio-temporal graph networks are commonly tackled using recurrent networks or through spatio-temporal convolutions. For example, structural-RNNs are used to learn about interactions between humans and objects in videos [13]. Likewise, Wang *et al.* [45] model interactions in videos using convolutional appearance features as graph nodes and Yang *et al.* [52] use a spatio-temporal graph convolution network for person re-identification. In this paper, we also focus on spatio-temporal graphs, but consider the more challenging scenario where nodes can enter and exit scenes over time and where nodes exhibit multiple relation types. Different from Xu *et al.* [47], who used functional time encodings for node classification and link prediction tasks, our method operates directly on the graph for multi-relational edge prediction.

Scene graph generation. Scene graphs have many applications, such as in image captioning [2, 16, 53], visual question answering [25] and image generation [15, 33]. To construct a scene graph, classifiers are trained to predict the categories of object detections and their relationships. Different from scene graph generation, we seek to predict the future state of object relationships based on a temporally-dynamic spatio-temporal graph, rather than relationships with a known state in a static scene graph. Nevertheless, we provide comparisons with state-of-the-art predicate classifiers of the scene graph generation task in Sec. 4.3.

3 MTD-GNN

For the problem of temporally-dynamic edge prediction of future states, we construct a spatio-temporal graph $\mathcal{G}_{\text{input}}$ with F timesteps from a video with T frames, denoted as $\mathcal{G}_{\text{input}} = \{G_0, \ldots, G_{F-1}\}$, where $F < T$. Rather than representing an object with a unique node, we define a new node for each detected object at each timestep. Let $\mathcal{N}$ denote the total number of detected nodes in the graph and N_i the nodes at timestep i. We denote the set of spatial edges as $\mathcal{E}^s = \{\mathcal{E}^s_0, ..., \mathcal{E}^s_{F-1}\}$ and the temporal edges as $\mathcal{E}^t = \{\mathcal{E}^t_0, ..., \mathcal{E}^t_{F-2}\}$. Spatial edges are between different objects at the same timestep, temporal edges are between the same object in consecutive timesteps (see Fig. 2). Our goal is to predict the spatial edge labels between pairs of nodes for all relations $r \in \mathcal{R}$ in the final frame T using $\mathcal{G}_{\text{input}}$. In other words, the model observes object interactions until timestep F and predicts their future state

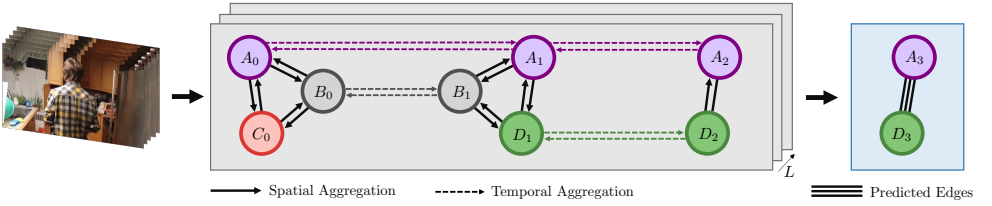

Figure 2: **Proposed architecture.** Nodes in the spatio-temporal graph are attended in space and time over L graph attention layers. Each pair of updated nodes is fed to the edge prediction module, which predicts a pair-wise edge value for all relations R.

at timestep T. We compute a pairwise categorical edge matrix $E_r \in \mathbb{R}^{N_F \times N_F}$, where N_F is the last known number of detected nodes in $\mathcal{G}_{\text{input}}$. In our graph network, we first perform graph attention in space and time simultaneously to learn node-level representations, after which we utilize the node representations to optimize edge prediction for multiple relation types.

Spatio-temporal scene graph generation. To operate on video graphs, we first construct a temporally-dynamic spatio-temporal graph $\mathcal{G}_{\text{input}}$ from a video (Fig. 2). Given pre-trained Mask-RCNN [10] and Faster-RCNN [38] backbones for CLEVRER and ActionGenome, respectively, we extract d-dimensional feature representations of each detected object such that the total set of node features of $\mathcal{G}_{\text{input}}$ becomes $\mathbf{v} = \{\vec{v}_1, \vec{v}_2, \ldots, \vec{v}_{\mathcal{N}}\}$ with $\vec{v}_i \in \mathbb{R}^d$. In our experiments, we set d to 256 and 2048 for CLEVRER and ActionGenome, respectively. We then generate a joint spatio-temporal adjacency matrix $\mathbf{A} \in \mathbb{R}^{\mathcal{N} \times \mathcal{N}}$ with adjacencies between spatial and temporal neighbours over consecutive timesteps. Specifically, we connect all detected objects in a particular frame z spatially, while temporally connect each object only with itself if detected in frame $z-1$ or $z+1$. We apply Hungarian matching between frames $(\mathbf{v}_z, \mathbf{v}_{z+1})$ and $(\mathbf{v}_z, \mathbf{v}_{z-1})$ to ensure an accurate appearance-based connection without accessing its ground truth location. When the object detector misses an object, its relationships with detected nodes are ignored. False positive detections can occur when building the spatio-temporal graph, but corresponding predicted edges are not evaluated. We ensure this by also applying Hungarian matching between ground truth and proposed bounding boxes.

Factorized spatio-temporal graph attention. Using graph attention [43], we seek to learn node representations that allow for invariance to the number of neighboring nodes. For each node, we compute hidden representations by attending over its corresponding spatio-temporal neighbours (Fig. 2). We propose a factorized multi-headed spatio-temporal graph attention layer that takes as input the set of node features $\mathbf{v} \in \mathbb{R}^{\mathcal{N} \times D}$ and outputs features $\mathbf{h} = \{\vec{h}'_1, \vec{h}'_2, \ldots, \vec{h}'_{\mathcal{N}}\}$ with $\vec{h}'_i \in \mathbb{R}^{D'}$ averaged over K attention heads and latent dimension D':

$$\vec{h}'^{k}_i = \sum_{j \in A_i} \mathbb{1}[A_{ij} \in \mathcal{E}^s]\, \alpha^k_{ij} \mathbf{W}^k_j v_j \; + \; \mathbb{1}\left[A_{ij} \in \mathcal{E}^t\right] \gamma^k_{ij} \mathbf{W}^k_j v_j \quad , \tag{1}$$

where α^k_{ij} is the spatial and γ^k_{ij} the temporal attention coefficient computed for node pairs (v_i, v_j) in attention head k, $\mathbf{W}^k_j \in \mathbb{R}^{D' \times D}$ is a weight matrix, $\mathcal{E}^s$ and $\mathcal{E}^t$ are mutually exclusive sets of spatial and temporal connections, and $\mathbb{1}[C] = \{1 \text{ if } C = true,\ 0 \text{ else}\}$ is the indicator function. The key idea of our factorized spatio-temporal graph attention is the separation between relational *spatial* information as well as *temporal* object information, thus enabling a richer graph layer. We obtain the final node representation by averaging over the output

features of the K attention heads and by applying a sigmoid non-linearity σ:

$$\vec{h}'_{i,0} = \sigma\left(\frac{1}{K}\sum_{k=1}^{K}\vec{h}_i^k\right), \qquad \vec{h}'_{i,L} = \sigma\left(\frac{1}{K}\sum_{k=1}^{K}\vec{h}_{i,L-1}^k\right), \qquad L \geq 1 \ . \tag{2}$$

After one iteration, each node is informed about its first-order neighbors. By performing factorized graph attention repeatedly, information of higher-order neighborhoods is included.

Multi-relational edge learning. The graph with all detected objects in target frame T is assumed to be fully connected. A prediction is made for all pair-wise connections, where edges are undirected and task-dependent. We aim to infer multiple types of relations for each edge simultaneously by learning task-specific fully-connected layers for each relation $r \in R$ using the node representation outputs from graph attention layers. We obtain a loss value for each separate task r per sample by evaluating the predicted task-specific edges E_r with the respective ground truth labels Y_r, resulting in a total loss represented as:

$$\mathcal{L}_{\text{total}} = \sum_{r\in R}\sum_{i\in E_r}\sum_{j\in E_r^i}\mathcal{L}(E_r^{ij}, Y_r^{ij}) \quad \text{with} \quad E_r^{ij} = E_r^{ji} = \psi_r\left(\frac{1}{2}(\vec{h}'_{i,L} + \vec{h}'_{j,L}) + b\right) \ , \tag{3}$$

where $\vec{h}'_{i,L}$ and $\vec{h}'_{j,L}$ are the final output features of nodes i and j, respectively, b is a bias vector and ψ constitutes the fully-connected layers with non-linear activations. For each edge, we ensure permutation invariance for its two nodes by averaging their respective representations. Some datasets contain a large number of objects and therefore have many edges to predict. Depending on the edge type, this can lead to class imbalance, *e.g.* with collision. To balance the losses across different tasks, we outline a *prioritized loss*, which emphasizes the edge class which is less frequent in the training set. The binary cross-entropy (BCE) losses of over-represented class o for an individual task r are down-weighted by normalizing it with the number of ground truth labels of the larger class in the inferred frame, represented as:

$$\mathcal{L}_{\text{prio}} = \left(\mathbb{1}\left[Y_r^i = 0\right] o^{-1} + \mathbb{1}\left[Y_r^i = 1\right]\right) \cdot \mathcal{L}_{\text{BCE}}(E_r^i, Y_r^i) \ . \tag{4}$$

4 Experiments

In our experimental evaluation, we perform a series of ablation studies to investigate the core components of our MTD-GNN, a comparative evaluation to existing approaches that generalize to our setting and qualitative analyses showing success and failure cases.

4.1 Experimental Setup

Datasets. We evaluate on the synthetic CLEVRER dataset [54] and real-world dataset Action Genome [14], as both datasets contain both dynamic object interactions and multiple edge types. CLEVRER features multiple object-object relation types, whereas Action Genome focuses on multiple human-object relation types. In both datasets, a target is defined for each pair of objects that share a spatial edge. For CLEVRER, we consider two types of prospective, indirected relations: collision (contacting) and relative motion (spatial). The goal is to predict the future state of these relations among objects present in a particular scene. To achieve this setting, we leave out X frames in the model's input, where X is uniformly sampled between 5 and 20. For Action Genome, we follow the relation types

	Collision prediction			Relative motion		
	F1 ↑	AP ↑	AUC ↑	F1 ↑	AP ↑	AUC ↑
Latent Dimensions						
256	**0.505**	**0.441**	**0.668**	**0.798**	**0.812**	**0.672**
512	0.472	0.417	0.624	0.780	0.810	0.668
1024	0.436	0.395	0.601	0.776	0.816	0.674
Attention Heads						
3	0.458	0.373	0.589	**0.798**	0.812	0.672
5	**0.505**	**0.441**	**0.668**	0.786	0.819	0.676
7	0.426	0.446	0.642	0.784	0.824	**0.687**
9	0.440	0.435	0.642	0.766	**0.826**	0.684
Attention Layers						
1	**0.505**	**0.441**	**0.668**	**0.798**	**0.812**	**0.672**
2	0.459	0.340	0.517	0.780	0.786	0.616
3	0.485	0.341	0.520	0.797	0.790	0.613
Learning Method						
Single-task	0.505	0.441	0.668	0.798	0.812	0.672
Multi-task	**0.594**	**0.607**	**0.768**	**0.839**	**0.820**	**0.688**

Table 1: **Overview of all ablation studies on CLEVRER [54]** with the following insights: (i) fewer graph dimensions are beneficial; (ii) five attention heads help to balance complexity and generalization; (iii) one attention layer is all you need; and (iv) multi-task learning is favored over individual optimization.

from the original paper, which constitute a total of 25 human-object directional relationship classes, divided into three *attention*, six *spatial* and seventeen *contacting* relationship types. In the spatial and contacting categories, targets can be multi-label. For example, an object might be beneath, behind, and on the side of the human at the same time. Here, the relationships in the final frame are predicted. Frames with less than two objects are omitted, as their scene graphs lack edges.

Implementation. All models and baselines are implemented in Python and PyTorch 1.7 [35]. Adam [19] is used for optimization, with an initial learning rate of $\eta_0 = 0.001$ which is decreased every epoch e following $\eta_e = \eta_{e-1}/(1+0.9e)$. Each model is trained for 100 and 10 epochs for CLEVRER and Action Genome respectively. Due to its imbalanced nature, prioritized loss is used for CLEVRER, while we train with a BCE loss on Action Genome.

4.2 Ablation Studies

Attention dimensionality. The attention mechanism is trained with a latent representation of the original input $\vec{h} \in \mathbb{R}^{N \times D'}$, where D' is the number of latent features. First, we investigate the effect of the feature dimensionality in our factorized spatio-temporal graph attention layer. In this initial setting, we use a single graph attention layer and 3, 5, 7, or 9 attention heads. The results are shown in Table 1 in the supplementary materials for CLEVRER and in Table 2 for Action Genome. Factorized attention enables us to utilize spatial and temporal information through separate channels. Using 256 latent dimensions works best for CLEVRER and increasing it further results in a small but consistent performance decrease. For Action Genome, the number of latent dimensions has less of an impact on the performance, as metric scores are rather consistent across all settings. In further experiments, we keep 256 dimensions for CLEVRER and 512 for Action Genome in the graph attention layer.

Attention heads. Having multiple attention heads, *i.e.*. attention mechanisms, stabilizes the learning process in attention-based approaches [42, 43]. In our second study, we investigate the effect of the number of attention heads on the performance of MTD-GNN. The results are presented in Table 1 and Table 2 for CLEVRER and Action Genome respectively. The amount of attention heads directly impacts the collision detection performance on CLEVRER across all metrics. When using three attention heads, we obtain an F1 score of 0.458 for collision detection, which improves to 0.505 with five heads. Similar behavior is observed for spatial edge types in Action Genome, where the F1 score, AP and AUC score

	Attention			Spatial			Contacting		
	F1↑	AP↑	AUC↑	F1↑	AP↑	AUC↑	F1↑	AP↑	AUC↑
Latent Dimensions									
256	0.365	0.743	0.702	**0.366**	**0.746**	**0.883**	**0.371**	**0.743**	**0.963**
512	**0.367**	**0.746**	**0.706**	0.366	0.745	0.882	0.350	0.737	0.962
1024	0.364	0.745	0.705	0.364	0.745	0.882	0.362	0.712	0.951
Attention Heads									
3	**0.367**	**0.746**	**0.706**	0.344	0.731	0.876	0.364	**0.744**	**0.963**
5	0.356	0.740	0.699	0.361	0.744	0.882	0.362	0.743	0.962
7	0.364	0.745	0.705	0.364	0.745	0.882	**0.371**	0.743	**0.963**
9	0.364	0.745	0.705	**0.366**	**0.746**	**0.883**	0.360	0.743	**0.963**
Attention Layers									
1	**0.367**	**0.746**	**0.706**	**0.366**	**0.746**	**0.883**	**0.371**	0.743	**0.963**
2	0.230	0.636	0.595	0.359	0.740	0.880	0.364	**0.744**	**0.963**
3	0.364	0.744	0.705	0.327	0.721	0.872	0.364	**0.744**	**0.963**
Learning Method									
Single-task	**0.367**	0.746	0.706	0.366	**0.746**	**0.883**	**0.371**	**0.743**	**0.963**
Multi-task	**0.367**	**0.747**	**0.708**	**0.392**	0.711	0.802	0.300	0.607	0.889

Table 2: **Overview of all ablation studies on ActionGenome [14]** with the following insights: i) finding the optimal model parameters for real-life data is challenging; ii) edge types with few classes benefit from multi-task learning.

increase from 0.344, 0.731 and 0.876 to 0.366, 0.746 and 0.883 when increasing the amount of attention heads from three to nine. A potential reason for the effectiveness of multiple heads is that it enables learning multiple dynamics in a single layer, an important ability given our setting. However, not all tasks seem to benefit consistently across all metrics. For example, in the relative motion task, using more than three attention heads benefits the AP and AUC score while decreasing the F1 score. In Action Genome, three heads lead to best performance across all three metrics in the attention edge type, whereas with the spatial edge type, nine heads is highly preferred. In the following experiments we maintain five heads for CLEVRER and nine heads for Action Genome as overall well performing configurations.

Number of aggregations. The architecture of MTD-GNN allows for repeated spatio-temporal feature aggregation. In theory, this allow each node's features to be informed about nodes further down the graph, *i.e.*. second-, third, n-order neighbors, thereby capturing more temporal dynamics. This ablation study investigates how many aggregations are preferred. The results are shown in Table 1 and Table 2 for CLEVRER and Action Genome respectively. Interestingly, more than one aggregation layers is not preferred across both datasets. The performance decreases consistently across recorded metrics, *e.g.* from an F1 score of 0.505 for one aggregation level to 0.459 and 0.485 for respectively two and three levels of aggregation in collision prediction. The same phenomenon occurs across all metrics in Action Genome for attention and spatial edge types. For contacting edge types, having more than one attention layer has a negative effect on the F1 score, while slightly enhancing the AP. Using a large number of attention layers likely causes over-smoothing in the final feature representations. We conclude that it is more important to richly model a single aggregation layer with multiple attention heads and latent attention dimensions, than to model higher-order neighbourhood relations and temporal self-relations.

Multi-relational learning. In the fourth ablation study, we investigate the importance of multi-relational modeling. In Table 1, results for collision detection and relative motion prediction on CLEVRER are shown. The results demonstrate that learning multiple relations in parallel enhances the prediction performance of each individual task. For collision detection, especially, the addition of the relative motion task is important, as performance improves from an F1 score of 0.505 to 0.594. The outcomes indicate that the model is able to exploit the dynamics of the scene better when presented in multi-relational manner, where weights are optimized using multiple targets. The result also aligns with our intuition that information about relative motion provides a useful cue for collision detection and vice versa.

	Attention			Spatial			Contacting		
	F1↑	AP↑	AUC↑	F1↑	AP↑	AUC↑	F1↑	AP↑	AUC↑
Vanilla baselines									
RNN [13]	0.364	0.744	0.705	0.387	0.699	0.775	0.291	0.575	0.868
LSTM [31]	0.365	0.745	0.700	**0.394**	0.714	0.800	0.316	0.627	0.897
TCN [4]	0.347	0.730	0.686	0.378	0.698	0.786	0.287	0.603	0.888
Graph attention (GA) baselines									
RNN + GA	0.364	0.745	0.705	0.387	0.695	0.766	0.292	0.705	0.879
LSTM + GA	0.364	0.744	0.705	0.391	0.703	0.784	0.298	0.614	0.893
TCN + GA	0.354	0.728	0.680	0.363	0.681	0.781	0.306	0.598	0.883
This paper									
MTD-GNN	**0.367**	**0.746**	**0.706**	0.366	**0.746**	**0.883**	**0.371**	**0.743**	**0.963**

Table 4: **Comparative evaluation on Action Genome [14].** MTD-GNN compares favorably to the baselines across nearly all metrics and edge types, showing its effectiveness on real-world data.

Table 2 shows the results for the edge types in Action Genome. Interestingly, learning in multi-task setting is preferred for some edge types, but not all. Specifically, we see a performance increase for the attention edge type with three classes, at the cost of the contacting edge type with seventeen classes. This outcome is surprising, since one could argue that attention and spatial edge types provide useful cues for contacting edge types. The outcome might indicate that when edge types have different number of classes, only the one with the least classes benefits from the multi-task setting. However, the performance increase in F1-score from 0.366 to 0.392 for the spatial edge type contradicts this, since it has double as many classess as attention edge types. We conclude that the performance gain from multi-task learning in Action Genome is dependent on the edge type to predict.

4.3 Comparative Evaluation

We compare against RNN [13], LSTM [31], and TCN [4] architectures, along with an attention-based variant of each baseline, which are generalized to the temporally-dynamic and multi-relational nature of our problem. To enable the baselines to cope with temporally-dynamic data, each graph is padded with nodes. The amount of padded nodes depends on the maximum number of objects per frame throughout each dataset (six/ten in CLEVRER/Action Genome). Nodes are ordered by index, hence structural information is lost. Each baseline encodes the spatio-temporal graph over the temporal domain, and the final output is used to predict the edges. In the attention-based variants, the final features in the temporal encoding are used to perform feature aggregation. Here, before the edges are predicted, the time-encoded nodes are aggregated with neighboring nodes. The resulting features are fed to an edge prediction layer which predicts the edge values per relationship type.

	Collision prediction			Relative motion		
	F1↑	AP↑	AUC↑	F1↑	AP↑	AUC↑
Vanilla baselines						
RNN [13]	0.247	0.410	0.527	0.733	0.710	0.514
LSTM [31]	0.289	0.404	0.595	0.750	0.805	0.634
TCN [4]	0.345	0.341	0.548	**0.839**	0.708	0.524
Graph attention (GA) baselines						
RNN + GA	0.168	0.410	0.597	0.835	0.758	0.591
LSTM + GA	0.283	0.389	0.604	0.796	0.783	0.606
TCN + GA	0.253	0.389	0.602	0.739	0.807	0.637
This paper						
MTD-GNN	**0.594**	**0.607**	**0.768**	**0.839**	**0.820**	**0.688**

Table 3: **Comparative evaluation on CLEVRER [54].** MTD-GNN compares favorable to the baselines, showing its effectiveness for multi-relational edge prediction in temporally-dynamic spatio-temporal graphs.

Comparisons on CLEVRER and Action Genome are reported in Table 3 and Table 4 respectively. For collision detection, we outperform the baselines on all metrics. This result shows the potential of modelling the dynamic spatio-temporal nature of the graphs, which is done in MTD-GNN but not in the baselines. The baselines do not benefit from additional graph attention, possibly due to aggregation of the present entities' node features with that of necessary padded nodes. We obtain an F1 score of 0.594 with MTD-GNN for the collision

task, compared to F1 scores of 0.345 for the best performing baseline, namely a vanilla TCN. We observe similar gaps for AP and AUC (0.607 versus 0.410 for RNN with graph attention as best baseline and 0.768 versus 0.604 for LSTM with graph attention as best baseline). For the relative motion task, we again obtain the highest overall performance, however with smaller gaps. Especially in F1 score, some baselines perform similar. One possible reason is that the baselines learn to predict the negative class more often, causing the model to obtain a lower false positive rate compared to a model which predicts the positive class more often.

MTD-GNN also outperforms nearly all baselines on Action Genome, however usually with smaller gaps for attention and spatial edge types. A greater difference occurs when evaluated on contacting edge type classes: MTD-GNN obtains an F1 score of 0.371, compared to 0.371 of the best performing baseline, namely a TCN with graph attention. An even greater difference occurs in AP and AUC score (0.743 and 0.963 versus 0.598 and 0.883). While the baselines struggle when the number of classes per edge type is large, such as with the "contacting" edge type, MTD-GNN succeeds in maintaining consistent performance throughout all metrics (*e.g.* an F1 score of 0.371 versus 0.316 for LSTM as best baseline).

Lastly, we compare against existing methods in the predicate classification task on Action Genome [14], which expect ground truth bounding boxes and object categories to predict predicate labels. We adopt our method to this ground truth setting and report results in Table 5. MTD-GNN outperforms all existing predicate classification methods when using ground truth boxes and object categories. It also achieves noteworthy performance when predicted boxes and categories are used, as can be seen in the last row of Table 5. However, this result also displays a limitation of MTD-GNN, which is that it's performance strongly depends on the accuracy of the detection backbone and the accuracy of temporal linking.

	Image		Video	
	R@20↑	R@50↑	R@20↑	R@50↑
With ground-truth detections				
VRD [30]	24.92	25.20	24.63	24.87
Freq Prior [54]	45.50	45.67	44.91	45.05
Graph R-CNN [51]	23.71	23.91	23.42	23.60
MSDN [23]	48.05	48.32	47.43	47.67
IMP [48]	48.20	48.48	47.58	47.83
RelDN [53]	49.37	49.58	48.80	48.98
MTD-GNN (Ours)	**50.09**	**50.09**	**49.54**	**49.54**
Without ground-truth detections				
MTD-GNN (Ours)	46.49	46.49	46.85	46.85

Table 5: Comparison with SOTA methods on ActionGenome [14].

The outcomes of our experiments could indicate that padding graphs is not a viable solution to deal with temporally-dynamic scenes. Preserving and aggregating original spatial and temporal dynamics displays benefits in performance. Judging from all outcomes altogether, we conclude that our approach obtains the best performance for multi-relational edge prediction in temporally-dynamic spatio-temporal graphs.

4.4 Qualitative Analysis

We perform a qualitative analysis on ActionGenome by providing success and failure cases for a multi-task MTD-GNN. Figure 3 shows test samples where the model achieved the highest and lowest F1 score. MTD-GNN can readily account for settings with multiple visible objects, while dealing with occluded objects over time and distinct relationships of visually similar objects remain an open problem due to inaccurate temporal linking.

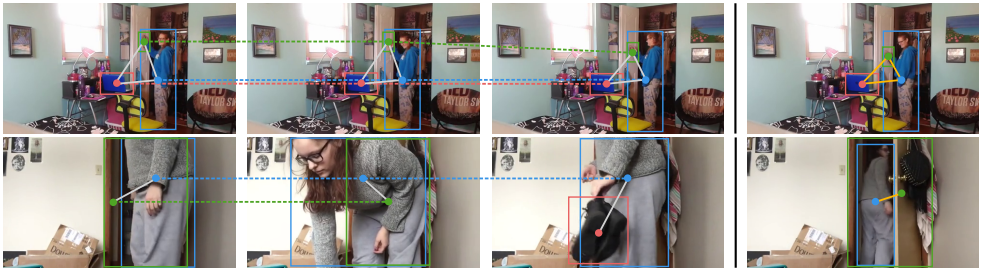

Figure 3: **Qualitative analysis on ActionGenome [14].** The top row shows a success case, where the model predicts all relations (*e.g.* looking at laptop and holding towel) in the final frame correctly. For this, it uses the spatio-temporal graph built from the video frames before the black separation line. When objects are visible throughout the sequence, MTD-GNN benefits from temporal linking and infers relationships accurately. The bottom row shows a failure case for relations between the person and doorway. The object detector classifies the doorway as a door, which causes our method to make the wrong associations between the person and the other detected object. Hence, an inaccurate set of relationships is predicted: "person *in front of* door" instead of "is *in* doorway".

5 Conclusion

This paper investigates how to perform edge prediction of multiple relation types simultaneously in a temporally-dynamic spatio-temporal graph network. Different from common spatio-temporal graph networks, we do not assume a single set of entities, but allow for the number of entities in the scene to vary over time. This makes our model more suitable for real-world scenarios with dynamic scenes. To address this challenging problem, we propose a factorized spatio-temporal graph attention layer. On top of this layer, we outline a multi-task optimization with an optional prioritized loss for multi-task learning. Our experiments on CLEVRER and Action Genome show that our attention-based approach can model dynamic relations in graphs, while modelling multiple relations simultaneously can be beneficial when predicting individual relations. Our approach compares favorably to approaches that can generalize to the proposed setting, as well as to state-of-the-art methods in the predicate classification task. Future research on this topic could focus on recovering from false or missed detections by the detection backbone and applying MTD-GNN to other applications, such as action forecasting.

6 Acknowledgement

This work has been financially supported by Mercedes-Benz, TomTom, the University of Amsterdam and the allowance of Top consortia for Knowledge and Innovation (TKIs) from the Netherlands Ministry of Economic Affairs and Climate Policy. We furthermore thank Theo Gevers, Sezer Karaoglu, Martin Oswald, Yu Wang, Ysbrand Galama and Georgi Dikov for their valuable input when writing this paper.

References

[1] Lada A Adamic and Eytan Adar. Friends and neighbors on the web. *Social Networks*, 2003.

[2] Antoine Bordes, Nicolas Usunier, Alberto Garcia-Durán, Jason Weston, and Oksana Yakhnenko. Translating embeddings for modeling multi-relational data. In *NeurIPS*, 2013.

[3] Weiqi Chen, Ling Chen, Yu Xie, Wei Cao, Yusong Gao, and Xiaojie Feng. Multi-range attentive bicomponent graph convolutional network for traffic forecasting. In *AAAI*, 2020.

[4] Zhao-Min Chen, Xiu-Shen Wei, Peng Wang, and Yanwen Guo. Multi-label image recognition with graph convolutional networks. In *CVPR*, 2019.

[5] Zulong Diao, Xin Wang, Dafang Zhang, Yingru Liu, Kun Xie, and Shaoyao He. Dynamic spatial-temporal graph convolutional neural networks for traffic forecasting. In *AAAI*, 2019.

[6] Wenqi Fan, Yao Ma, Qing Li, Yuan He, Eric Zhao, Jiliang Tang, and Dawei Yin. Graph neural networks for social recommendation. In *The World Wide Web Conference*, 2019.

[7] Lizhao Gao, Bo Wang, and Wenmin Wang. Image captioning with scene-graph based semantic concepts. In *ICMLC*, 2018.

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

[9] Liyu Gong and Qiang Cheng. Exploiting edge features for graph neural networks. In *CVPR*, 2019.

[10] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask r-cnn. In *2017 IEEE International Conference on Computer Vision (ICCV)*, 2017.

[11] Roei Herzig, Elad Levi, Huijuan Xu, Hang Gao, Eli Brosh, Xiaolong Wang, Amir Globerson, and Trevor Darrell. Spatio-temporal action graph networks. In *ICCV workshops*, 2019.

[12] Yue Hu, Siheng Chen, Xu Chen, Ya Zhang, and Xiao Gu. Neural message passing for visual relationship detection. In *ICML workshops*, 2019.

[13] Ashesh Jain, Amir R Zamir, Silvio Savarese, and Ashutosh Saxena. Structural-rnn: Deep learning on spatio-temporal graphs. In *CVPR*, 2016.

[14] J. Ji, R. Krishna, L. Fei-Fei, and J. Niebles. Action genome: Actions as compositions of spatio-temporal scene graphs. In *CVPR*, 2020.

[15] Justin Johnson, Agrim Gupta, and Li Fei-Fei. Image generation from scene graphs. 2018.

[16] Dong-Jin Kim, Jinsoo Choi, Tae-Hyun Oh, and In So Kweon. Dense relational captioning: Triple-stream networks for relationship-based captioning. In *CVPR*, 2019.

[17] Jongmin Kim, Taesup Kim, Sungwoong Kim, and Chang D Yoo. Edge-labeling graph neural network for few-shot learning. In *CVPR*, 2019.

[18] Sungwoong Kim, Sebastian Nowozin, Pushmeet Kohli, and Chang Yoo. Higher-order correlation clustering for image segmentation. In *NeurIPS*, 2011.

[19] Diederik Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *ICLR*, 2014.

[20] T. Kipf, E. Fetaya, K. Wang, M. Welling, and R. Zemel. Neural relational inference for interacting systems. In *ICML*, 2018.

[21] T. N. Kipf and M. Welling. Semi-supervised classification with graph convolutional networks. *ICLR*, 2017.

[22] Yehuda Koren, Robert Bell, and Chris Volinsky. Matrix factorization techniques for recommender systems. *Computer*, 2009.

[23] Juho Lee, Yoonho Lee, Jungtaek Kim, Adam Kosiorek, Seungjin Choi, and Yee Whye Teh. Set transformer: A framework for attention-based permutation-invariant neural networks. In *ICML*, 2019.

[24] Bin Li, Xi Li, Zhongfei Zhang, and Fei Wu. Spatio-temporal graph routing for skeleton-based action recognition. In *AAAI*, 2019.

[25] Linjie Li, Zhe Gan, Yu Cheng, and Jingjing Liu. Relation-aware graph attention network for visual question answering. 2019.

[26] Ruiyu Li, Makarand Tapaswi, Renjie Liao, Jiaya Jia, Raquel Urtasun, and Sanja Fidler. Situation recognition with graph neural networks. In *ICCV*, 2017.

[27] Yang Li, Yadan Luo, and Zi Huang. Fashion recommendation with multi-relational representation learning. In *PAKDD*, 2020.

[28] Yikang Li, Wanli Ouyang, Bolei Zhou, Kun Wang, and Xiaogang Wang. Scene graph generation from objects, phrases and caption regions. 2017.

[29] Xiaodan Liang, Xiaohui Shen, Jiashi Feng, Liang Lin, and Shuicheng Yan. Semantic object parsing with graph lstm. In *ECCV*, 2016.

[30] Cewu Lu, Ranjay Krishna, Michael Bernstein, and Li Fei-Fei. Visual relationship detection with language priors. In *ECCV*, 2016.

[31] Zhilong Lu, Weifeng Lv, Yabin Cao, Zhipu Xie, Hao Peng, and Bowen Du. Lstm variants meet graph neural networks for road speed prediction. *Neurocomputing*, 2020.

[32] Li Mi and Zhenzhong Chen. Hierarchical graph attention network for visual relationship detection. In *CVPR*, 2020.

[33] Gaurav Mittal, Shubham Agrawal, Anuva Agarwal, Sushant Mehta, and Tanya Marwah. Interactive image generation using scene graphs. *ICLR*, 2019.

[34] Maximilian Nickel, Kevin Murphy, Volker Tresp, and Evgeniy Gabrilovich. A review of relational machine learning for knowledge graphs. *Proceedings of the IEEE*, 2016.

[35] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library. In *NeurIPS*. 2019.

[36] Siyuan Qi, Wenguan Wang, Baoxiong Jia, Jianbing Shen, and Song-Chun Zhu. Learning human-object interactions by graph parsing neural networks. In *ECCV*, 2018.

[37] Xiaojuan Qi, Renjie Liao, Jiaya Jia, Sanja Fidler, and Raquel Urtasun. 3d graph neural networks for rgbd semantic segmentation. In *ICCV*, 2017.

[38] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster R-CNN: Towards real-time object detection with region proposal networks. In *Advances in Neural Information Processing Systems (NIPS)*, 2015.

[39] Franco Scarselli, Marco Gori, Ah Chung Tsoi, Markus Hagenbuchner, and Gabriele Monfardini. The graph neural network model. *IEEE Transactions on Neural Networks*, 2008.

[40] Gurkirt Singh, Stephen Akrigg, Manuele Di Maio, Valentina Fontana, Reza Javanmard Alitappeh, Suman Saha, Kossar Jeddi Saravi, Farzad Yousefi, Jacob Culley, Tom Nicholson, Jordan Omokeowa, Salman Khan, Stanislao Grazioso, Andrew Bradley, Giuseppe Di Gironimo, and Fabio Cuzzolin. ROAD: the road event awareness dataset for autonomous driving. *CoRR*, 2021.

[41] Rianne van den Berg, Thomas N. Kipf, and Max Welling. Graph convolutional matrix completion. In *KDD*, 2017.

[42] 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.

[43] P. Veličković, G. Cucurull, A. Casanova, A. Romero, P. Liò, and Y. Bengio. Graph Attention Networks. In *ICLR*, 2018.

[44] Xiang Wang, Xiangnan He, Yixin Cao, Meng Liu, and Tat-Seng Chua. Kgat: Knowledge graph attention network for recommendation. In *ACM SIGKDD*, 2019.

[45] Xiaolong Wang and Abhinav Gupta. Videos as space-time region graphs. In *ECCV*, 2018.

[46] Zizhang Wu, Man Wang, Jason Wang, Wenkai Zhang, Muqing Fang, and Tianhao Xu. Deepword: A gcn-based approach for owner-member relationship detection in autonomous driving. *ICME*, 2021.

[47] Da Xu, Chuanwei Ruan, Evren Korpeoglu, Sushant Kumar, and Kannan Achan. Inductive representation learning on temporal graphs, 2020.

[48] Danfei Xu, Yuke Zhu, Christopher B. Choy, and Li Fei-Fei. Scene graph generation by iterative message passing. 2017.

[49] S. Yan, Y. Xiong, and D. Lin. Spatial temporal graph convolutional networks for skeleton-based action recognition. In *AAAI*, 2018.

[50] Bishan Yang, Wen-tau Yih, Xiaodong He, Jianfeng Gao, and Li Deng. Embedding entities and relations for learning and inference in knowledge bases. In Yoshua Bengio and Yann LeCun, editors, *ICLR*, 2015.

[51] Jianwei Yang, Jiasen Lu, Stefan Lee, Dhruv Batra, and Devi Parikh. Graph R-CNN for scene graph generation. 2018.

[52] Jinrui Yang, Wei-Shi Zheng, Qize Yang, Ying-Cong Chen, and Qi Tian. Spatial-temporal graph convolutional network for video-based person re-identification. In *CVPR*, 2020.

[53] Xu Yang, Kaihua Tang, Hanwang Zhang, and Jianfei Cai. Auto-encoding scene graphs for image captioning. 2018.

[54] Kexin Yi, Chuang Gan, Yunzhu Li, Pushmeet Kohli, Jiajun Wu, Antonio Torralba, and Joshua B. Tenenbaum. Clevrer: Collision events for video representation and reasoning. In *ICLR*, 2019.

[55] Bing Yu, Haoteng Yin, and Zhanxing Zhu. Spatio-temporal graph convolutional neural network: A deep learning framework for traffic forecasting. In *IJCAI*, 2018.

[56] Rowan Zellers, Mark Yatskar, Sam Thomson, and Yejin Choi. Neural motifs: Scene graph parsing with global context. 2018.

[57] Runhao Zeng, Wenbing Huang, Mingkui Tan, Yu Rong, Peilin Zhao, Junzhou Huang, and Chuang Gan. Graph convolutional networks for temporal action localization. In *ICCV*, 2019.

[58] Chuxu Zhang, Dongjin Song, Chao Huang, Ananthram Swami, and Nitesh V. Chawla. Heterogeneous graph neural network. In *KDD*, 2019.

[59] Ji Zhang, Kevin J. Shih, Ahmed Elgammal, Andrew Tao, and Bryan Catanzaro. Graphical contrastive losses for scene graph generation. 2019.

[60] Muhan Zhang and Yixin Chen. Weisfeiler-lehman neural machine for link prediction. In *KDD*, 2017.

[61] Muhan Zhang and Yixin Chen. Link prediction based on graph neural networks. In *NeurIPS*, 2018.

[62] Long Zhao, Xi Peng, Yu Tian, Mubbasir Kapadia, and Dimitris N Metaxas. Semantic graph convolutional networks for 3d human pose regression. In *CVPR*, 2019.

[63] Penghao Zhou and Mingmin Chi. Relation parsing neural network for human-object interaction detection. In *ICCV*, 2019.