2021 International Conference on Information and Communication Technology Convergence (ICTC) | 978-1-6654-2383-0/21/$31.00 ©2021 IEEE | DOI: 10.1109/ICTC52510.2021.9621009

Graph Neural Network based Scene Change Detection Using Scene Graph Embedding with Hybrid Classification Loss

Soyeon Kim, Kyung-no Joo, Chan-Hyun Youn*
School of Electrical Engineering
Korea Advanced Institute of Science and Technology
Daejeon, South Korea
{kimcando94,eu8198,chyoun}@kaist.ac.kr

***Abstract*—The advent of deep learning technologies gives satellite imagery analysis birth to unprecedented achievements to various tasks. Especially, change detection is one of the attentive fields regarding to remote sensing as a unique task to compare the paired images. While a great amount of works deals with change detection in pixel level to generate change map, its labelling cost to train the model in data driven manner is extremely high in that it should be annotated in pixel level as well and it is sensitive to pixel level distortion. Instead of change maps, scene level change detection only classifies whether the newly coming image has different contexts or not especially when the system has target objects in the scene with comparably low labelling cost and considering overall contexts. However, only few works address scene level change detection and are yet unexplored with multiple target objects. In this end, we propose a two-phase framework to screen out the redundant same images compared to the reference time point image. Instead of using image features or object features only, we utilize scene representation graph and train on our proposed GNN architecture as to compare graphs representing images with multiple objects. Due to lack of perfect matching dataset, we verify our proposed framework on correspondingly matchable datasets and show the performance improvement on scene change type classification by 13% including *move* cases over the baseline.**

***Index Terms*—Scene change detection, Graph Neural Network, Scene Graph**

I. Introduction

Recently, deep learning has been applied to multiple domain problems and has shown tremendous performance achievement. In particular, advances in sensor technology supported by high performance computing environment have attracted analysis system addressing huge amounts of data such as satellite imagery. Considering the role of satellite periodically taking certain areas, the major applications analyzing the satellite imagery is to monitor the urban or environmental development or target objects state changes. Therefore, one of the unique tasks is change detection to detect change in the given temporal gap, usually with paired images. The main approach in terms of modern deep learning is to pixel level change detection. A pair of images are fed to a neural network which shares each weight [1] in order to digest the images as the encoding part and a change map of which each pixel is decided by binary decisions whether each pixel has undergone the changes or not is generated by the decoder part. Compared to pixel level decision change map, scene change detection [2] is related to decide changes based on semantically high level information. Since two tasks imply different level of information, two approaches should be considered rather complementary each other.

However, in the view of service management which must process the heterogeneous data efficiently, the key functionality is to let users focus on the valuable scope of data, require low level annotations to sustain the system and robust to any inherent inconsistencies of data. Pixel level change detection might require heuristic threshold crafting [3], high cost ground truth to train the model and is susceptible to natural inconsistencies caused by view point changes or covariate shift problems due to heterogeneous data collecting platform, or weather dependent factors such as illuminations or occlusions [4] unless hand-crafted pre-processing. In comparison, scene level change detection only requires class label level annotations to train the model and decision based on high level contents in the scene is less likely to undergo performance degradation since it does not solely rely on pixel information.

Although scene level change detector has clear benefits for other domain applications as well [5], it is yet unexplored. Even few works [6] deciding the changes whether the target object exists or not only address the single object in the images. Straightfowardly, scenes with multiple objects are more natural settings to make use of its benefits than scenes with a single object. Furthermore, if the scene level change detector focuses more on contents or objects in the scene, it would be more beneficial for system operators to see categorized scenes based on types of changes such as *newly appeared*, *disappeared* so that each category can be processed accordingly. To focus on objects' states, a more refined approach needs to avoid losing fine-grained object change somewhere in the middle of the sequential convolution and pooling layers [7], leading to the misclassification for the changes.

In this end, we propose a two-phase framework for a

* corresponding author

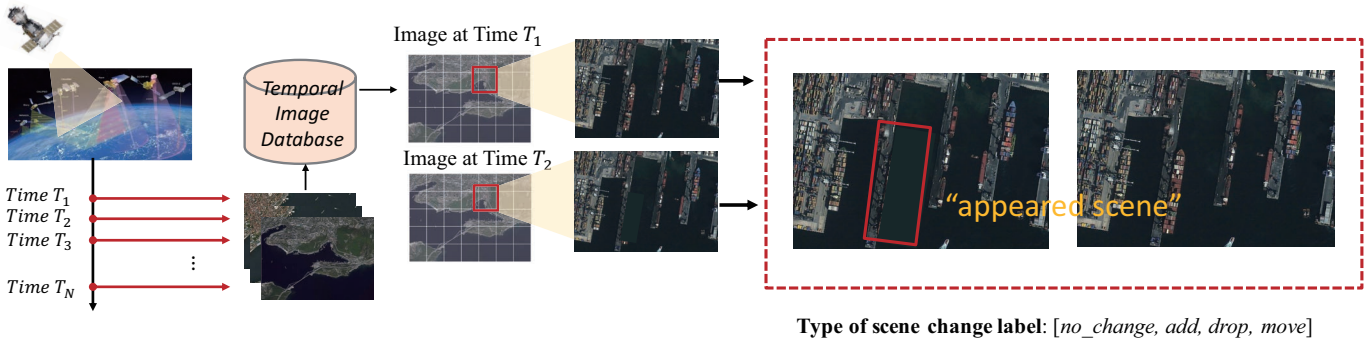

Fig. 1. An overall concept of scene change detection.

scene change detection to compared the paired images in the given temporal gap under multiple objects. The first phase involves in making scene graphs from images representing the object fine-grained information and spatio-topology relationship between objects. For the next phase, we propose a GNN architecture followed by a hybrid loss to complement differences and scene information to help the classifier decide scene change classifications. Due to the lack of the proper dataset, we validate our proposed scheme on correspondingly matcheable dataset from the previous works [5] by reorganizing and demonstrate the competitive performance compared to baseline, Siamese CNN based approach [6]. Our contribution is summarized as

- We argue that a scene level change detection is in need beyond the pixel level change detector and propose a two-phase framework for a scene change detection under multiple objects to classify the types of changes.
- In the proposed framework, we define the way to compose the nodes and edges features to construct a scene graph considering our task based on the fine-grained information and spatio-topological relationship between objects.
- To train the graph neural network for comparing the pair of graphs, we propose a GNN architecture and hybrid loss which has two branches where the one side is to focus on the differences and the other side is to incorporate the overall semantic information.

II. Related Works

A. Change Detection

Change detection refers to catch the difference by comparing pairs or several images during interval. [8], [1]. Change includes the case whether the part disappears, appears or change its properties. The research is mainly applied in remote sensing area to use change information for the sake of monitoring landscape usage or city development monitoring. [4]. Thus, one of the main approach on detecting change is related to generate change map utilizing U-net [9] style segmentation method accompanying with Siamese-like architecture [10] owing to its versatility in distinguishing between similarity and dissimilarity. Another branch to detect change facilitates object level information by giving certain labels on section of area. [6]. However, most of the previous works deal with pixel level change information. Thus, deciding whether two scenes are different or not directly from the segmentation result might follow heuristic threshold crafting [3]. Apart from the remote sensing application, several recent works highlight the importance on providing explanation in the form of captioning to describe changed objects in the paired scenes. [5]. Instead of giving details in textual information about the scene, our work tackles more fundamental problem by reformulating the problem as classifying two scenes identical or not.

B. Graph Neural Network

Graph neural network is a method to solve graph related problems such as graph or node classification and link prediction by extracting graph or node embedding. One of the known algorithm is Graph Convolutional Network [11]. Even the algorithm stems from spectral graph processing side and does approximation, its underlying mechanism is just similar to deep convolutional network [12] in a sense that it reflects its neighborhood nodes. While the image data is strictly on grid, graph neural network addresses unordered neighbor nodes storing network topological structure. Among numerous applications, our work exploits GNN to measure graph similarity having same aim with [13], [14], but we add additional classification task.

C. Scene Change Detection

Scene graph is a way to represent the scene consisting of objects, attributes and relationship between objects. [15]. Generating scene graph pipeline itself has attracted significant interests among researcher such as how to model semantic relationship between objects [16]. Recent studies to exploit scene graph is image retrieval as semantic representative of the scenes [17], visual question answering [18] to match the question in structured approach, image captioning to mix visual features and textual information to leverage the multi modalities. [19].

III. Proposed Two Step Scene Change Detection Using Scene Graphs

In this section, we explain a structured procedure for scene change type detection using scene graph representation and a new GNN architecture and loss leveraging macro and micro information to train a classifier. To clarify the scope of work, we assume there are multiple objects which have predefined attributes and each attribute is classified by detector. Specifically, we assume a cause for change is triggered by the single object. Especially, the trigger in our work includes that if one of the single object at time T has changed its attributes, disappeared, or new single object has appeared or f any of the single cardinal direction relationship between the whole objects has happened given time interval T.

A. First Step: Scene Graph Construction

1) Node features: As an essential features of objects, we use object features which are the output of the region proposal network [20]. Following the traditional assumption to use pre-trained object detector such as Faster RCNN [20], each object features after RoI pooling O_k, denoted in eq.(1), is extracted from the pre-trained region proposal network, f_{rpn}.

$$O_t = \{o_k | f_{rpn}(I^t) := o_k \text{ where } k \in \mathbb{Z}_+, 0 \leq k \leq N-1\} \tag{1}$$

where N is the total number of detected objects in the scene. Concerning that object features represent lower level features to mean the shape pattern learnt by region proposal network loss function, it is desirable to embed each node with more distinguishable feature so that each object's fine-grained attribute characteristic is not lost. To compensate it, we train multi-label function,f_{ml}, to extract each object high level attributes. For k^{th} at time t object attributes, denoted in eq.(2), can be defined as

$$A_{(k,t)} = \{a_m | f_{ml}(o_k) := a_m \text{ where } m \in \mathbb{Z}_+, 0 \leq m \leq M-1, k \in \mathbb{Z}_+, 0 \leq k \leq N-1\} \tag{2}$$

where M is the predefined number of attributes to detect for each object. With the detected results, the total node embedding is composed of object low level features and a series of attribute mapped to GloVe [21] embedding, denoted in eq.(3), as high level features.

$$g_{ge}(A_{(k,t)}) = g_{ge}(\{a_1, \cdots, a_m\}) = \{\nu_1, \cdots, \nu_m\} \tag{3}$$

where g_{ge} is a function to project a textual input to the high dimensional vector spaces. Thus, the final node feature for a single object, i at time t corresponds to eq.(4). Note that we omit the time notation of object and attribute features for readability.

$$x_i^t = [o_i : \nu_{i,1} : \nu_{i,2} : \nu_{i,m}] \in \mathbb{R}^{d+mk} \tag{4}$$

2) Edge features: To represent each object spatial interaction information as relative distance [22], we encode edge vector and normalized with original image width and height. This edge vector represents the closeness, horizontally or vertically related abstraction between nodes.

$$e_{i,j} = \Big[\frac{c_{i,x1} - c_{j,x1}}{w}, \frac{c_{i,x2} - c_{j,x2}}{w}, \frac{c_{i,y1} - c_{j,y1}}{h}, \frac{c_{i,y2} - c_{j,y2}}{h}\Big], \tag{5}$$

where the edge feature is $e_{i,j} \in \mathbb{R}^4$, each $c_{i,x}, c_{j,y}$ is Cartesian coordinate of each object bounding box. Figure 2 shows the detail of scene graph generation process. Since a change detector needs the pair of images I^t, I^{t+1}, constructing scene graph is done for each image.

B. Second Step: SC-GNN

Using the extracted scene-graph, finally we inference the graph similarity through GNN. To make the GNN infer the similarity between two given graphs, we train the GNN with the tailored method composed of two loss functions. These loss is to utilize information in a hybrid manner that has different volume of information to feed the final graph embedding into each network branch. At first, given the constructed pair of graphs, we feed graphs into GNN. Basically, any type of GNN following the message passing mechanism is applicable.

$$x_{k-1,i}^t = x_{k-2,i}^t + \Bigg[\Bigg(\sum_i x_{k-2,i}^t \oplus e_{i,j}^t\Bigg) W_{conv}^{k-2} + x_{k-2,i}^t W_{self}^{k-2} + b^{k-2} \tag{6}$$

Instead of having each own weights, Graph convolution parameters, $\theta = \{W_{conv}, b\}$, is sharable between the objects taking advantage of removing redundant parameters and increasing network ability to compare by projecting both pairs into same latent space [10]. Then, our hybrid loss function is used to train the binary classifier and categorical classifier for scene change types having each parameters ϕ, ψ respectively. After updating nodes, we use mean average function f_{aggr} to get the global graph function.

1) Micro information loss: : Micro information is to let the network more focus on the changeable information. Thus, feeding residual, subtracting graph embeddings, is to give an explicit difference hint of the paired scenes. As the underlying mechanism of binary cross entropy function that it gives a penalty if a neural network gives high probabilities for a different change, it leads to learn whether the possible change information exists or not. Thus, by feeding residuals, we amplify the training effect to train the binary classifier.

$$h_k^{t1} = f_{aggr}(g_\theta(X^{t1}, E^{t1})), h_k^{t2} = f_{aggr}(g_\theta(X^{t2}, E^{t2})) \tag{7}$$

where X^t, E^t is a set of nodes and edges constructing graph G^t at time t respectively. Again, g_θ is GNN sharing the weights and f_{aggr} is one of the pooling function but with no trainable parameters. The input of binary branch,$\hat{y}_{bin} \in [0, 1]$,

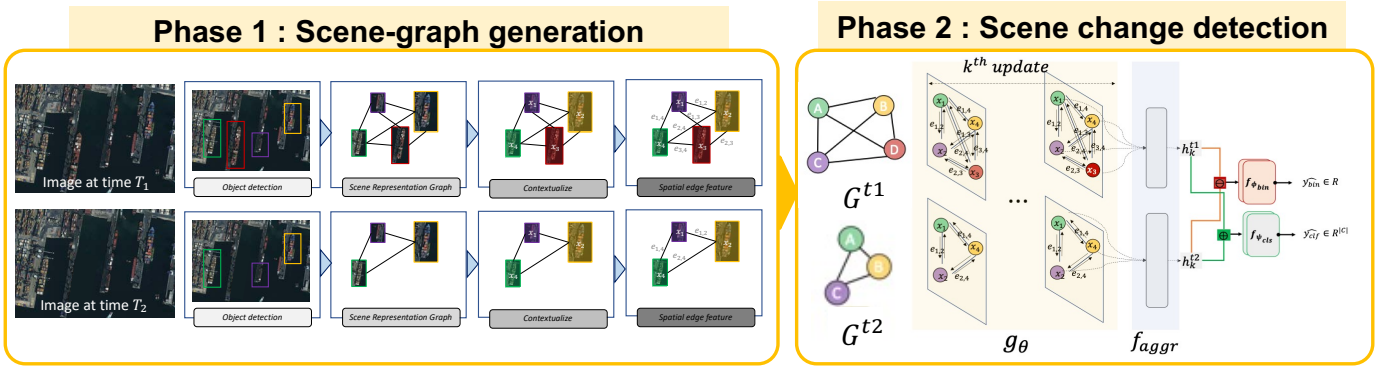

Fig. 2. Overall training process. The details for the first step to compose the node and edge for a single scene graph is on the right side. The red branch : corresponding to the binary classification task on the SC-GNN. The green branch : corresponding to the multi task classification task.

trained by micro loss and corresponding binary cross entropy with true match label y_i is

$$\hat{y}_{bin} = f_{\phi,bin}(f_{aggr}(g_\theta(X^{t1}, E^{t1})) - f_{aggr}(g_\theta(X^{t2}, E^{t2}))) \tag{8}$$

$$\mathcal{L}_{bin} = -\frac{1}{N}\sum_{i=1}^{N} y_i log(\sigma(\hat{y}_{bin})) + (1-y_i)log(1-\sigma(\hat{y}_{bin})). \tag{9}$$

2) Macro information loss: : To elaborately detect the change between two graph, we additionally adopt classification loss to distinguish types of change. It takes concatenated tensors of two output features and infers the type of change. Herein, we bridge the categorical cross entropy frequently adopted to classify the images to our approach that needs classify the type of the scene change for instance color, shape and so on. Instead of only focusing on the difference, it needs to see overall scene information to classify what types of changes the single object is causing comparing the pair of image components. Thus, by feeding concatenating information to the neural network and train with categorical loss function.

$$\hat{y}_{cls} = f_{\psi,cls}(f_{aggr}(g_\theta(X^{t1}, E^{t1})) \oplus f_{aggr}(g_\theta(X^{t2}, E^{t2}))) \tag{10}$$

$$\mathcal{L}_{cls} = -\frac{1}{N}\sum_{i=1}^{N} y_i log\Big(\frac{e^{\hat{y}_{cls}}}{\sum_j^C e^{\hat{y}_{cls}}}\Big), \tag{11}$$

where $\oplus$ means operation to concatenate two feature vector.

To sum up the training procedure, the proposed GNN model consists of trainable parameters $\Theta = \{\theta, \phi, \psi\}$ where θ is the parameters for the graph convolution parameters and ϕ, ψ are the each branch realized by the simple fully connected layers. While updating, the trainable parameters are optimized with weighted combination of the two loss function $\gamma_1\mathcal{L}_{bin} + \gamma_2\mathcal{L}_{cls}$ to balance the unchanged and changed cases whereas f_{aggr} is only applied to aggregate overall node features but have no trainable parameters.

IV. Evaluation

A. Dataset Description

We choose CLEVR [23] dataset, because the temporal data perfectly matched to our task or that can be manipulated to our objective does not exist. To briefly explain CLEVR [23] dataset, it consists of synthetic 3D–rendered objects mainly for visual question answering task and provides open API to generate images. Each object has 3 shapes, 2 materials, 8 colors, 2 sizes totally 96types can be manipulated. Since our scenario requires the with multiple objects with several attributes that can be detected with a simple classifier, having coordinate wise topological relationship between each other and being able to manipulated to make pairs of images, CLEVR does suit to our proposed algorithm verification. Although the numerical performance score itself can represent high values compared to the experiments applied to the real satellite imagery, we more focus on to the positive role of each component to train the scene graph based scene change detection.

To elaborate the dataset to our aim, we use basic separation rule that previous works [5] have done to categorize images into three groups; default image as reference image, non semantic image as considered to be same but have *distractor* factors such as viewpoint or illumination, semantic changed image as considered to be different due to *color, texture, add, drop or move*. Then, to train in pair scheme, we separate each semantic of five categories having same number and pair with default images. All of the image have the same height and width size, 224 $\times$ 224, with three channels.

B. Implementation Details

Our implementation is based on PyTorch and pytorch Geometric. All the experiments are conducted under GeForce GTX 1080 Ti and Intel(R) Xeon(R) Silver 4110 CPU @ 2.10GHz.

1) Pre-training: We pre-trained Faster-RCNN on official part of CLEVR dataset because [5] did not provide the object information labels but only image captioning labels. We calculate bounding box position based on the given information in the official dataset such as pixel coordinates, 3d coordinates.

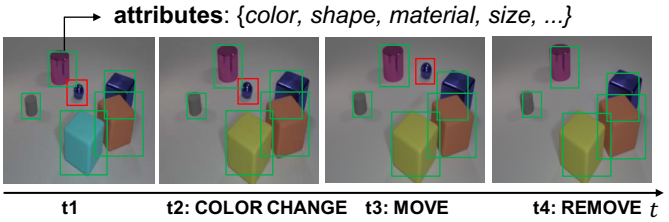

Fig. 3. Description of dataset used in [5] based on [23].

After training, we use object detector only to extract object features and attributes without optimizing.

2) Training details: The Siamese baseline and object detector backbone are all pretrained ResNet101 [24]. Regarding to the embedding dimension, RoI feature dimension is set to 2048 dimension, attributes have three types having each attribute beging 100 dimension for each. To train GNN, we stack two GNN layers followed by two fully connected. Then, for scene change type classification, output dimension set to 6 if including move else 5. Because same data case was five times bigger, we used weighted cross entropy loss for categorical classification branch. With Adam optimizer, we used learing rate 0.0001.

C. Metric for Performance Evaluation

For evaluation, we use precision, recall and F1 score. For the analysis, we mainly mention F1 score.

$$\text{Precision} = \frac{tp}{tp + fp} \qquad \text{Recall} = \frac{tp}{tp + fn} \tag{12}$$

$$\text{F1 score} = 2 \cdot \frac{precision \cdot recall}{precision + recall} \tag{13}$$

Here tp is true positive, fp means false positive, and fn denotes false negative.

We modified well known siamese network architectural style in change detection. We re-design the baseline architecture to be trainable in our dataset by subtract the image and feed them to ResNet101 [24] both for binary task and classification task. We present the result to show each component's role to improve our proposed scheme. Specifically, we show the contribution of attribute embedding, edge information updating and hybrid loss to boost up the performance.

The baseline is the modified Siamese network representing the conventional scene change detection approach. From case 1 to case 4 is our proposed scene graph based approach. **Case 1** is the very baseline to only have object feature for node embedding and no use to relationship while GNN updating. **Case 2** is to have attribute embedding, **Case 3** is whether it is trained with proposed hybrid loss. **Case 4** is same with case 3 but to use edge information while updating. That is, case 4 has all the components we propose. For the case 4, we utilize one of the architecture that incorporating edge information while updating [25].

TABLE I
SUMMARY OF EXPERIMENT 1 RESULTS.

Type	Metric	Baseline	Case 1	Case 2	Case 3	Case 4
Same	precision	0.445	0.769	0.797	0.831	0.833
	recall	0.677	0.648	0.614	0.806	0.733
	F1	0.537	0.703	0.693	**0.818**	**0.78**
Color	precision	0.879	0.24	0.653	0.82	0.815
	recall	0.817	0.43	0.722	0.797	0.937
	F1	0.847	0.3	0.686	**0.808**	**0.8179**
Material	precision	0.395	0.769	0.795	0.852	0.901
	recall	0.387	0.81	0.902	0.897	0.92
	F1	0.391	0.789	0.845	**0.874**	**0.919**
Add	precision	0.715	0.65	0.761	0.76	0.884
	recall	0.835	0.717	0.71	0.737	0.927
	F1	0.77	0.684	0.845	**0.748**	**0.901**
Drop	precision	0.667	0.632	0.657	0.715	0.826
	recall	0.872	0.717	0.72	0.73	0.927
	F1	0.756	0.672	0.689	**0.748**	**0.901**
Move	precision	0.263	0.146	0.153	0.187	0.154
	recall	0.025	0.102	0.28	0.212	0.215
	F1	0.045	0.12	0.2	**0.199**	**0.179**

1) Scene Change Classification with Move: Table.I shows the result when the experiments are conducted including all possible changed states. Considering *color, add, drop* would have significant pixel differences, the baseline work well in three cases. However, for the *same, material, move* have poor performance while our proposed method overwhelms with large margin. To see the crucial role in each component, all cases work better when attribute embedding is added except same and utilizing GNN architecture style which incorporate edge features have better performance. This is because the proposed methods give distinctive properties in order to strengthen the comparison capability. Above of all, utilizing hybrid loss has strong trends to have the best performance compared to other settings. Not surprisingly, *move* case is the hardest case for both baseline and proposed methods, still proposed approach enjoy far better numerical performance overall.

2) Modified Scene Change Classification with Move: As the shown result in I, the main cause of the performance degradation heavily depends on **Move** case both for the baseline and the proposed model. Even the proposed cases overwhelm the baseline, we modify the proposed case to improve the move case performance. To mitigate the degradation, we add the center positional information in the node features. Following the eq.(4), we concatenate the x and y coordinates for each object. Thus, the only difference is the minor dimension. In fact, only adding the positional information explicitly on the node makes a dramatic improvement on the move case as shown in the result table in II.

3) Scene Change Binary Classification: For the binary classification results, baseline F1 score achieves 85% while scene graph based methods get approximately 95%. Regardless of the additional components, scene graph based methods do not have large variations on performance, still case4, training with the proposed hybrid loss and using edge information, achieves the best performance, 96 %.

TABLE II
SUMMARY OF EXPERIMENT 2 RESULTS

Type	Metric	Case 3	Case 4	Case 3+CP	Case 4 + CP
Same	F1	0.818	0.78	0.886	0.873
Color	F1	0.808	0.818	0.875	0.747
Material	F1	0.874	0.919	0.899	0.901
Add	F1	0.748	0.901	0.766	0.909
Drop	F1	0.748	0.901	0.739	0.880
Move	F1	0.199	0.179	**0.567**	**0.523**
Overall	F1	0.74	0.76	0.829	0.833

ACKNOWLEDGMENT

This work was supported by the Institute for Information and Communications Technology Promotion (IITP) grant funded by the Korean Government (MSIT) (Service Mobility Support Distributed Cloud Technology) under Grant 2017-0-00294.

V. CONCLUSION AND FUTURE WORKS

Among various application in satellite image analysis, we tackle the scene change detection problems specifically under multiple objects condition scenario by recasting the image comparison problem into scene graph comparison to represent each object in more fine-grained manner. In our two-phase framework, a novel graph similarity training scheme with GNN architecture and hybrid loss function is proposed. To show the validity, we reorganize the dataset basedon the previous works. The experimental results demonstrate that our proposed method outperform naïve Siamese CNN based approach especially for move cases. Furthermore, throughout the ablation studies, we show the crucial role of each component of our proposed method. Even our proposed method gives better results on the experiments, we conjecture that baseline and the proposed method based on scene graph has pros and cons in terms of each case. In addition, our work covers classifying the change type of single object change under multiple objects. Thus, exploiting both advantages of CNN and GNN and considering multiple changes on multiple objects would be future works.

REFERENCES

[1] L. B. Mou, Lichao and X. X. Zhu., "Learning spectral-spatial-temporal features via a recurrent convolutional neural network for change detection in multispectral imagery." *IEEE Transactions on Geoscience and Remote Sensing 57.2*, 2018.

[2] B. D. Ru, Lixiang and C. Wu., "Multi-temporal scene classification and scene change detection with correlation based fusion." *IEEE Transactions on Image Processing*, 2020.

[3] e. a. Zhan, Yang, "Change detection based on deep siamese convolutional network for optical aerial images." *IEEE Geoscience and Remote Sensing Letters*, 2017.

[4] Z. M. Z. R. C. S. . Z. Z. Shi, W., "Change detection based on artificial intelligence: State-of-the-art and challenges," *Remote Sensing, 12(10), 1688.*, 2020.

[5] T. D. Park, Dong Huk and A. Rohrbach, "Robust change captioning," *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2019.

[6] e. a. Rahman, Faiz, "Siamese network with multi-level features for patch-based change detection in satellite imagery." *IEEE Global Conference on Signal and Information Processing*, 2018.

[7] e. a. Yu, Chaojian, "Hierarchical bilinear pooling for fine-grained visual recognition." *Proceedings of the European conference on computer vision (ECCV)*, 2018.

[8] H. G. J. C. L. M. . W. M. A. Chen, G., "Object-based change detection," *International Journal of Remote Sensing,*, 2012.

[9] P. F. Ronneberger, Olaf and T. Brox., "U-net: Convolutional networks for biomedical image segmentation," *International Conference on Medical image computing and computer-assisted intervention*, 2015.

[10] R. Z. Koch, Gregory and R. Salakhutdinov, "Siamese neural networks for one-shot image recognition." *ICML*, 2015.

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

[12] e. a. Wu, Zonghan, "A comprehensive survey on graph neural networks." *IEEE transactions on neural networks and learning systems*, 2020.

[13] e. a. Bai, Yunsheng, "Simgnn: A neural network approach to fast graph similarity computation." *Proceedings of the Twelfth ACM International Conference on Web Search and Data Mining.*, 2019.

[14] e. a. Li, Yujia, "Graph matching networks for learning the similarity of graph structured objects." *International Conference on Machine Learning.*, 2019.

[15] A. Agarwal and A. Mangal, "Visual relationship detection using scene graphs: A survey," *arXiv preprint arXiv:2005.08045*, 2020.

[16] e. a. j. y. Xu, Danfei, "Scene graph generation by iterative message passing."

[17] e. a. Johnson, Justin, "Image retrieval using scene graphs." *Proceedings of the IEEE conference on computer vision and pattern recognition.*, 2015.

[18] T. D. Park, Dong Huk and A. Rohrbach, "Graph-structured representations for visual question answering," *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2017.

[19] ——, "Auto-encoding scene graphs for image captioning," *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2019.

[20] e. a. Ren, Shaoqing, "Faster r-cnn: Towards real-time object detection with region proposal networks." *arXiv preprint*, 2015.

[21] R. S. Pennington, Jeffrey and C. D. Manning., "Glove: Global vectors for word representation," *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, 2014.

[22] T. D. Park, Dong Huk and A. Rohrbach, "Exploring visual relationship for image captioning," *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2019.

[23] e. a. Johnson, Justin, "Clevr: A diagnostic dataset for compositional language and elementary visual reasoning." *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition.*, 2017.

[24] e. a. He, Kaiming, "Deep residual learning for image recognition." *Proceedings of the IEEE conference on computer vision and pattern recognition.*, 2016.

[25] T. Xie and J. C. Grossman, "Crystal graph convolutional neural networks for an accurate and interpretable prediction of material properties." *Physical review letters 120.14*, 2018.