Learning Combinatorial Solver for Graph Matching

Chaitanya Kulkarni Hritwik Goklaney

November 15, 2020

Abstract

Graph matching is one of the classic problems in the computer vision application. Graph matching- due to its combinatorial nature is an NP-hard problem; thus, the traditional methods give the only approximate solutions for the matching. Thus the learning-based approaches improve accuracy and outperform the traditional approaches. Most of the learning-based algorithms carry out learning on the node/edge affinities by maximizing the correlation matrix or minimizing the assignment loss functions. The paper aims to develop a fully trainable network by learning the combinatorial solver instead of just learning the affinity matrix for the graph matching.

1 Introduction

Graphs are a powerful way of representing and modelling objects and have found a variety of applications in real life. Graphs widely used in fields such as pattern recognition, molecular biology, computer vision, scene analysis in which analysis has to be conducted, and images are to be processed. Graph matching, one of the fundamental problems in computer science, refers to a class of computational problems which comprise finding an optimal correspondence between the vertices of graphs to minimize or maximize their node and edge affinities. Although researchers have conducted several studies devoted to graph matching, it remains a challenging problem due to the combinatorial constraints.

Several approaches are there to solve the attributed graph matching problem. In the past, researchers have investigated many traditional and learning-based approaches to improve the accuracy of graph matching algorithms. Some of the early models involved finding correspondences between the two sets of features and relaxation of a discrete optimization problem. However, overall performance improvement from these models is limited. The learning graph matching has proved to be superior in terms of overall performance and in improving matching accuracy. Although learning-based graph matching comprises supervised, unsupervised and semi-supervised learning algorithms, most conventional algorithms for learning graphs are supervised ones. Learning the parameters of the graph affinity metric to replace the handcrafted affinity metric effectively boosts the accuracy of learning algorithms.

The combinatorial solver approach transforms the problem of building node correspondences between input graphs into the selection of reliable nodes from a constructed assignment graph. The model includes both the methods, i.e. learning the affinity matrix and combinatorial optimization into a single learning skeleton.

2 Related Work

2.1 Traditional Graph Matching

Significant studies have been conducted in the field of Graph Matching, and many algorithms have been proposed. Due to the combinatorial nature of the graph matching problem, the optimal solutions are hardly available, especially for large graphs. On the other hand, traditional methods solve the minimization problem approximately.Approximation methods may be divided into two groups of algorithms.

2.1.1 Spectral Correspondence

The first group comprises methods which use spectral representations of adjacency matrices. Leordeanu and Hebert[1] conducted an exhaustive study and proposed an efficient spectral method to find consistent correspondence between two sets of features which comprised the adjacency matrix M of a graph in which nodes represented the potential correspondences and the weights on the links represented pairwise agreements between potential correspondences. Their study was based on the observation that the correct assignments in the strongly connected cluster resulted in agreement links(edges with positive weights). These links are formed when pairs of assignments agree at the level of pairwise relationships between the features they are putting in correspondence. On the contrary, incorrect assignments outside of that cluster or weakly connected to it do not form strongly connected clusters because of their small probability of establishing agreement links. The problem then gets reduced to the problem of detecting the main strongly connected cluster in the assignments graph, which can be solved using the eigenanalysis approach.

2.1.2 Graduated Assignment

These algorithms involve a relaxation of the discrete optimization problem. Since graph matching is an NP-hard problem, several constraint relaxation and post discretization methods have been proposed. Gold and Rangarajan[2] presented a fast graduated assignment algorithm for graph matching, which is accurate even in the presence of high noise. Zaslavskiy[3] proposed a convex-concave programming approach for the labelled weighted graph matching problem. This approach reformulates the problem as a least-square problem on the set of permutation matrices and divides it to two different optimization problems which consist of a convex and a concave optimization problem on the set of doubly randomized matrices. Both optimization trajectories are quadratic in nature. Global minimum for concave relaxation is the same as the initial graph. However, the search for the global minimum is still a hard combinatorial problem.

2.2 Learning Graph Matching

The learning-based Graph Matching algorithms mainly include three approaches, namely, Supervised, Semi-supervised and Unsupervised. The supervised algorithms employ node labelling and optimize the affinity function. The Unsupervised learning model partitions the parameter space between to sets correct and incorrect assignments of the two graphs and then minimizes the ratio of the weighted areas of those two regions to get the optimal assignment. This approach is useful when no prior information about the assignments is known. If the user has some prior knowledge about the assignments of the two graphs, then the semi-supervised method becomes more appropriate—the semi-supervised matching implemented by combining both the node labelling and areal minimization.

2.2.1 Supervised Learning

The supervised approach improves the matching accuracy of the graph matching problem by learning the affinity matrix than using the handcrafted affinities. Caetano and McAuley[4] use detailed labelling of each node correspondence in each positive graph for training. The optimal solution is obtained by minimizing the normalized Hamming loss function. Torresan, Kolmogorov and Rother[5] implement the learning algorithm by finding the global optimal in the parameter space. Both the approaches use different optimization techniques viz. large margin methods [6], nonlinear inverse optimization [7]. A smoothing based technique to train matching parameters also implemented in a supervised way[8].

2.2.2 Unsupervised Learning

Unlike the supervised approach, unsupervised learning does not need detailed node labelling. Leordeanu, Sukthankar and Hebert[9] replaced the binary matching values obtained from the spectral matching approach by the algorithmic values. The algorithm maximizes the weighted correlation between the two graphs. The learning step includes updating the weights in the affinity matrix. The weights are updated such that the parameter space is partitioned into correct and incorrect assignments.

2.2.3 Semi-supervised Learning

The semi-supervised graph matching[10] implements both the supervised and unsupervised approaches. This algorithm has binary matching values for the nodes with prior labelling. The remaining nodes are matched in the unsupervised way by maximizing the correlation between the two graphs.

3 Implementation

The paper answers the issues with the traditional graph matching and current learning-based algorithm by developing a fully trainable framework. The learning framework for the graph matching is designed on the top of the graph network block (GNB) presented in [2]. The code for GNB is available at the GitHub repository https://github.com/deepmind/graph_nets . The model provided by the paper is available at https:

//www.cs.stonybrook.edu/hling/code/LearningGraphMatching.zip. It uses four data sets viz. Willow, Points2D, CMU-House and PascalVoc. All the datasets are provided in the same link as the for the code.

Our code uses Graph Nets, which is DeepMind's library for building graph networks in Tensorflow and dm-sonnet. To set up the environment for executing the python script, graph_nets TensorFlow tensorflow_probability and graph_nets tensorflow_gpu tensorflow_probability_gpu have to be installed. The version requirement for Tensorflow and sonnet is 1.* in contrast to the latest 2.0 version. Also, OpenCV has to be installed to meet the PascalVoc dataset requirements. In addition to that, the module requirements are met by executing python script setup.py.

The version requirement for the python script is 1.* for both dm-sonnet as well as TensorFlow. The newer version of sonnet does not include the Abstract Module. The code does not support the latest versions of the libraries and causes errors and needs to be fixed. Also, because of the resource constraints, the time taken to train the model on 100000 iterations is relatively high.

We evaluate the proposed algorithm on four benchmarks. Points 2D, CMUHouse, Willow, PascalVoc. We have run the model to train, test and evaluate the model on the four data sets as mention ed in the paper for 100000 data points. The observed accuracy is 94.2% which is closed to the actual given accuracy in the paper. For the demonstration purpose we ran the code on only 100 data points which given an accuracy of 79.52%.

4 Innovation

The proposed algorithm does not take into account the visual features of the input images. Instead, the model generates the graph solely based on the coordinates of each image. To incorporate the visual features, a CNN model needs to be implemented. We propose to introduce a Feature Extractor Module before generating the graphs from the input image.

4.1 Feature Extractor Module

The feature extractor module can be adopted from [12] by Wang, Yan. CNN's keypoint feature extraction is constructed using Feature map. VGG16 pre-trained with ImageNet is used to extract visual features from the input images. The VGG network offers a deeper yet simpler variant of the convolutional structures. ConvNet takes as an input fixed size RGB image. The image is passed through a stack of convolutional (Conv.) layers which is followed by three fully-connected layers. Therefore, it helps to take into account not only the coordinates of the input images but also the visual features of the input image. Feature Extractor module returns a feature tensor. This feature tensor is used as the node attribute, which is further used in the convolutional module.

4.2 Results

The evaluation of our proposed algorithm is conducted on two datasets- PascalVOC andThe Willow dataset. The sole combinatorial solver achieves an accuracy of about 94% on the Willow object dataset

consisting of five classes namely car, duck, face, motorbike and wine bottle. On the other hand, an accuracy of about 68% on the PascalVOC dataset. The accuracy on PascalVOC dataset is lesser as compared to Willow dataset because of variation in pose,illumination and number of inliers. On the other hand, a simple model employing a non-learning combinatorial solver like Sinkhorn net which takes into account the visual features of the images gives an accuracy of about 63% on the PascalVOC dataset and an accuracy of about 96% on the Willow object dataset. This shows that visual features play an important role in the graph matching problem. The sole CNN model takes a considerable amount of training time but gives promising results.If both CNN and learning combinatorial solver are employed, considerable increase in accuracy can be achieved with compromise on the training time.

References

[1] Marius Leordeanu and Martial Hebert. *A spectral technique for correspondence problems using pairwise constraints.*. In ICCV, pages 1482–1489, 2005.

[2] Steven Gold and Anand Rangarajan. *graduated assignment algorithm for graph matching. IEEE Trans. Pattern Anal.Mach. Intell.,* 18(4):377–388, 1996.

[3] Mikhail Zaslavskiy, Francis R. Bach, and Jean-Philippe Vert. *A path following algorithm for the graph matching problem.* IEEE Trans. Pattern Anal. Mach. Intell., 31(12):2227–2242,2009.

[4] Tibério S. Caetano, Julian J. McAuley, Li Cheng, Quoc V.Le, and Alexander J. Smola. *Learning graph matching.* IEEETrans. Pattern Anal. Mach. Intell., 31(6):1048–1058, 2009.

[5] Lorenzo Torresani, Vladimir Kolmogorov, and Carsten Rother. *Feature correspondence via graph matching: Models and global optimization.* In ECCV, pages 596–609, 2008.

[6] Ioannis Tsochantaridis, Thorsten Joachims, Thomas Hofmann, and Yasemin Altun. *Large margin methods for structured and interdependent output variables.* J. Mach. Learn.Res., 6:1453–1484, 2005

[7] C. Karen Liu, Aaron Hertzmann, and Zoran Popovic. *Learning physics-based motion style with nonlinear inverse optimization.* ACM Trans. Graph., 24(3):1071–1081, 2005.

[8] Marius Leordeanu and Martial Hebert. *Smoothing-based optimization.* In CVPR, 2008.

[9] Marius Leordeanu, Rahul Sukthankar, and Martial Hebert. *Unsupervised learning for graph matching.* International Journal of Computer Vision, 96(1):28–45, 2012.

[10] Marius Leordeanu, Andrei Zanfir, and Cristian Sminchisescu. *Semi-supervised learning and optimization for hypergraph matching.* In ICCV, pages 2274–2281, 2011

[11] Tao Wang, He Liu, Yidong Li, Yi Jin, Xiaohui Hou, Haibin Ling. *Learning*

Combinatorial Solver for Graph Matching Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2020, pp. 7568-7577

[12] Runzhong Wang, Junchi Yan, Xiaokang Yang. *Learning Combinatorial Embedding Networks for Deep Graph Matching*