Survey on implementation techniques for graph neural networks.

2130411 Toru Tamagawa

Graph neural networks (GNNs) and graph convolutional neural networks (GCNs) have attracted attention for handling class classification/regression problems for various relations that can be represented by graph structures, including social networks. In order to improve the speed of inference/training and reduce power consumption by implementing them in hardware, it is necessary to design them in consideration of the fact that the data distribution of non-zero elements is very unbalanced and the matrix size is very large when the graphs given as input are represented in matrix form. Although a number of studies have been conducted on the elemental technologies of GNN and GCN, there is a problem that the knowledge on the hardware implementation of GNN and GCN is incompletely organised. In this study, we organise the knowledge in the field of GNN and GCNs by identifying the classification and the relationship with the performance of the related previous studies, paying particular attention to the hardware implementation of GNN and GCN.

Hardware implementations of GNN and GCN have various optimisation goals, such as reduced inference latency, reduced power consumption, and more efficient memory access. In addition, the assumptions made in terms of the target GNN processing phases, datasets, networks, etc., vary widely from paper to paper. In order to clarify the trends of these implementations in existing studies, we have organised them based on the target phase, objective, dataset, network, device and quantisation accuracy for each method. The target phases of the hardware implementation of GNN and GCN are classified into three categories: (i) inference (corresponding to inference only), (ii) training (corresponding to training only) and (iii) inference and training (corresponding to both inference and training). As an overall trend in hardware implementation, with regard to phases, 75 papers cover only the inference phase, 20 papers cover only the training phase, and 5 papers cover both inference and training phases. The number of papers covering the inference phase is very large. This may be due to the fact that inference is the phase where trained models are used in real applications, and low-latency processing is required for execution in many real applications, and the demand for faster inference is higher than that for training. As for the objectives, most of the studies mention latency and power reduction as the two main goals, while other existing studies also aim at reducing off-chip memory access, chip area and off-chip memory bandwidth usage. Regarding datasets, (i) In terms of target datasets for inference, there were more than 40 papers covering the

top four datasets Cora, Reddit, Citeseer and Pubmed, which accounted for the majority. (ii) As for hardware implementations targeting only training, as in (i) above, Reddit, Yelp, OGBN-products and Amazon were the top four. (iii) For hardware implementations targeting both inference and training, the same datasets as (i) and (ii) were employed, including Reddit, Cora, Citeseer and Pubmed. As for the network, GCN is very often employed, and most of the devices are ASICs or FPGAs. As for quantisation precision, among those not explicitly mentioned in the papers, 32-bit fixed-point/32-bit floating-point is often adopted for feature vectors/weight vectors.