Graph Node Embeddings for ontology-aware Sound Event Classification: an evaluation study

Carlo Aironi, *Samuele Cornell*, *Emanuele Principi*, *Stefano Squartini*
Department of Information Engineering, Università Politecnica delle Marche, Italy
Email: (c.aironi, s.cornell)@pm.univpm.it, (e.principi, s.squartini)@univpm.it

Abstract—**Multi-label Sound Event Classification (SEC) is a challenging task which requires to handle multiple co-occurring sound event classes. Recent works proposed an ontology-aware framework for SEC in which a Graph-Neural Network (GNN) approach is trained to exploit labels co-occurrence information and improve the performance of a standard audio-feature based classifier via late-fusion. This GNN is fed a graph-based representation of the training set labels. In this paper we adopt such framework and perform an in-depth study on how the labels embeddings used to construct the graph representation can affect the performance. We perform our experiment on the FSD50K dataset and compare different embeddings strategies: two from previous works and two which haven't been considered yet for SEC applications. Our results show that node2vec embeddings lead to substantial performance improvements with respect to other embeddings strategies used in previously ontology-aware SEC works. Our best node2vec model leads to an absolute improvement of 3.39% in mean average precision with respect to the best competing embedding strategy, with a lower number of trainable parameters.**

Index Terms—**Sound Event Classification, Audio tagging, Graph Representation Learning, Graph Neural Networks, Ontology structure, node2vec.**

I. Introduction

Multi-label Sound Event Classification, also referred as multi-label Audio Tagging, aims to predict the presence of certain sound events in an audio recording. As multiple, different, sound events can occur in the same recording, in multi-label SEC class labels are not considered mutually exclusive, and algorithms must be able to handle multiple, co-occurring events. Typical SEC systems are based on deep neural networks (DNN) like, for instance, convolutional neural network (CNN) classifiers, working on either spectrogram-based features [1], [2] or directly from the raw waveform [3]. More recently, many methods use recurrent neural networks (RNN) [4]–[6] or convolutional recurrent neural networks (CRNN) [7]–[9], which can also capture temporal information of sound events, or even graph neural networks (GNN) for audio feature representation [10].

Most of these methods, do not take advantage of the intrinsic relationships between different co-occurring sound events which are detected within the same audio clip. In fact, some sound events are more likely to occur together as they can belong to the same category e.g. musical instruments or to the same acoustic scenario e.g. office environment. Some approaches have been proposed to embed information retrieved from labels relationships for classification tasks, primarily in computer vision [11]–[14], natural language processing [15], [16] and on audio domain [17], [18].

Both [17] and [18] propose an ontology-aware SEC framework in which a graph-based DNN approach is used to exploit prior knowledge about sound events ontology and co-occurrences. This graph DNN is fed a graph representation of the training set labels relationships, e.g. in [18] graph edges are defined by the labels correlation matrix and nodes embeddings by one-hot encoded vectors. The output of this graph DNN is used in both works [17], [18] to improve SEC performance of a more "classic", audio-features based, classifier via late fusion.

Hereafter, building upon these previous work, we study how the node embedding strategy employed in the graph-based DNN can affect SEC performance. In [18] only one-hot encoded vectors were considered for use as labels node embeddings. While in [17] pre-trained GloVe word embeddings are used. Here we study the use of two additional node embeddings strategies which haven't been previously explored for SEC tasks: node2vec [19] learned embeddings and pre-trained fastText [20] word embeddings. We perform experiments on the recently proposed FSD50K dataset [21] and show that node2vec embeddings bring substantial performance improvements over both one-hot and pre-trained fastText and GloVe word embeddings. Importantly, these improvements come at a modest increase in the number of trainable parameters when compared to the original audio-feature classifier used alone without the ontology graph DNN branch.

This paper is organized as follows: in Section II Graph Neural Networks, Recurrent-GNNs and Node Embedding are briefly introduced. Following, in Section III, we explain in detail our framework and approach. We present the experimental setup, the dataset used and discuss the results in section IV and finally in V we draw conclusions and outline possible future work. We make our code publicly available at: `github.com/aircarlo/MultilabelGraphSEC`.

II. Graph Neural Networks and Node Embeddings

The notion of graph neural networks was initially outlined in Gori et al. [22] and further elaborated in Scarselli et al. [23]. These early models implicitly define a Spatial Convolution structure [24]–[26], which was subsequently formalized with the concept of Message Passing (MP) mechanism, in which we try to acquire information by aggregating informative *messages* from neighboring nodes. Furthermore, they fall into the category of recurrent graph neural networks (RecGNNs) [27], in which a target node's representation is learned by propagating neighbor information in an iterative manner, sharing memory parameters, both between nodes dimension and time domain, until a stable fixed point is reached. On this work we focus on a later development proposed by Li et al., Gated Graph Neural Network (GGNN) [28] which employs a gated recurrent unit (GRU) [29] as a recurrent function, reducing the recurrence to a fixed number of steps.

Given an initial node embedding $\mathbf{x}_i^{(0)}$, from which to define a hidden representation $\mathbf{h}_i^{(0)}$ as:

$$\mathbf{h}_i^{(0)} = \mathbf{x}_i \,\|\, \mathbf{0}, \tag{1}$$

where $\|$ denotes the concatenation operation. A node hidden state $\mathbf{h}_i$ at timestep t is obtained aggregating its previous hidden state and its neighboring hidden states through the following equations:

$$\mathbf{m}_i^{(t+1)} = \sum_{j \in \mathcal{N}(i)} e_{j,i} \cdot \boldsymbol{\Theta} \cdot \mathbf{h}_j^{(t)}, \tag{2}$$

$$\mathbf{h}_i^{(t+1)} = \mathrm{GRU}(\mathbf{m}_i^{(t+1)}, \mathbf{h}_i^{(t)}), \tag{3}$$

where $e_{j,i}$ denotes the edge weight from source node j to target node i. GNN unroll the recurrence for a fixed number of steps T and use backpropagation through time in order to compute gradients.

A. Node Embedding

Node embedding plays a significant role in learning useful information from graph structured data. Broadly, node embedding refers to the task of mapping each node of a graph in a lower dimensional vector space, preserving "similarity" between native and embedded domains. Embeddings should capture the graph topology along with relationships between nodes and further relevant inherent information.

Many algorithms to generate node embeddings have been proposed [30], differing in particular in how node similarity is defined, which is a crucial aspect for effective graph-structured data modeling. On this work we focus on *node2vec* [19], a *random walk* based method which uses a random walk approach to generate network neighborhoods for nodes by stochastic sampling, and which defines similarity between nodes u and v as the probability that both u and v co-occur in a random walk over a network.

In *node2vec*, an encoder function $ENC(\cdot)$ maps the transition between graph domain and the embedding space, it is a lookup learnable matrix $\boldsymbol{Z}$:

$$ENC(v_i) = \boldsymbol{z}_i = \boldsymbol{Z}\mathbf{v}_i, \tag{4}$$

where $\mathbf{v}_i \in \mathbb{I}^{|\mathcal{V}|}$ is an indicator vector.

Similarity of a pair of nodes v_i and v_j is defined as the probability of visiting node v_j on a random walk of fixed length, starting from node v_i:

$$s_G(v_i, v_j) = Pr(v_j | v_i). \tag{5}$$

On the other hand, *similarity* in the embedding space is a softmax of the dot product between node's embedding vectors:

$$s_E(\boldsymbol{z}_i, \boldsymbol{z}_j) = \frac{\exp\left(\boldsymbol{z}_i^\top \boldsymbol{z}_j\right)}{\sum_{k \in \mathcal{N}_i} \exp\left(\boldsymbol{z}_i^\top \boldsymbol{z}_k\right)}, \tag{6}$$

where $\mathcal{N}_i$ is the neighbourhood set of node v_i, sampled during the random walk.

The objective of embedding is to maximize the likelihood of random walk co-occurrences, defined by the following loss function:

$$\mathcal{L} = \sum_{v_i, v_j \in V} \log\left(s_E(\boldsymbol{z}_i, \boldsymbol{z}_j)\right). \tag{7}$$

The strategy implemented in *node2vec* (illustrated in Figure 1) to define node neighbours is to use biased random walks that can trade off between local and global views of the network. Specifically, given a graph and starting from a specific node, two parameters p and q define the next hop probability during the walk, allowing to choose between a global macro-view of neighborhood (*Depth First Search* approach) or by first exploring the nearest nodes (*Breadth First Search* approach).

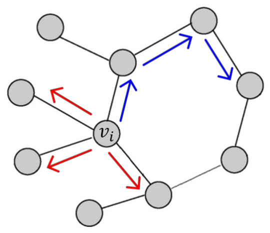

Fig. 1. *node2vec* sampling strategies: BFS-like walks (red) vs DFS-like walks (blue), starting from node v_i.

III. Ontology-Aware Sound Event Classification

As in [17], [18], also here we adopt an ontology-aware framework composed of two main modules: an *audio embedding module* and the label (graph) *co-occurrence learning module*. The proposed approach is illustrated in Figure 2.

The audio embedding module is used to extract an high-level, condensed representation from feature vectors extracted from the audio waveform, such as log-Mel filterbank energies (LFBE). On the other hand, the label co-occurrence learning module is based on a graph neural network. This latter learns node mappings via multi-layer GGNN and is fed with a fixed pre-defined graph, where each node has an initial embedding representation. We describe both modules in detail thereafter.

A. Audio embedding module

In this work, we consider for the audio embedding branch the Convolutional-Recurrent Neural Network (CRNN) architecture proposed in [31] and employed as a baseline method in [21]. Other previously proposed classifiers, such as [1]–[9], can be employed in principle. This model is fed in input a feature vector $V \in \mathbb{R}^{F \times T}$ obtained from a clip segment, where F and T are the dimension of each feature vector (e.g. number of Mel bands) and the number of time frames of the

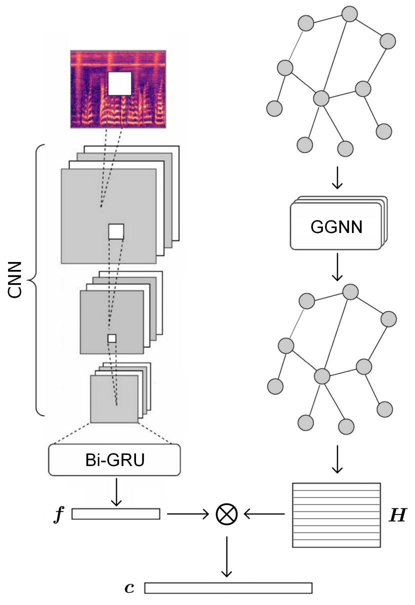

Fig. 2. Audio embedding module (left branch) and graph co-occurrence learning module (right branch) of the proposed CRNN-GGNN architecture.

input feature, respectively. These features are then processed by three convolutional blocks. Each block convolves the input feature map with two-dimensional filters; then, ReLU, max-pooling and batch-normalization are applied in this order. Following, a single-layer Bidirectional Gated Recurrent Unit (BiGRU) is applied on the output of the last convolutional layer. The forward and backward output vectors are then concatenated and placed as input of a single-layer dense network which outputs a final embedding vector $\boldsymbol{f} \in \mathbb{R}^D$ with dimension D which matches the output size of the label co-occurrence learning module.

B. *Label co-occurrence learning module*

The graph learning module depends on the initial labels node embeddings and the label correlation matrix M. These two components are in fact used to build up a graph representation where each node i is a vector of embedding size $\boldsymbol{x}_i \in \mathbb{R}^D$ and the graph edges are defined by the labels correlation matrix.

Previous works [12], [14], [17], [32], propose several approaches to build this correlation matrix. For example, [17] builds the correlation matrix as a binary matrix, where entries $M_{i,j}$ which represents edges between i-th and j-th label are either one or zero wether or not they share a common Audioset [33] parent class label. By contrast, here we adopt a strategy similar to [12], [14], [32] and focus on the label co-occurrence matrix of training data to build the correlation matrix, which is used to represent the structured graph of label relationships. Let N be the total number of classes, the generation process is as follows: using the training data ground-truth labels we compute the label co-occurrence matrix $M \in \mathbb{R}^{N\times N}$ whose elements $M_{i,j}$ counts the times of appearance of pairwise events (e_i, e_j); then the total occurrence of each label L_i in the training set is counted and the conditional probability matrix is calculated by:

$$M'_{i,j} = M_{i,j}/L_i.$$

Contrary to [12], [14], [32], we further remove diagonal elements from the resulting matrix, which correspond to self-loop connections. We found in our experiments that these degraded performance.

The labels co-occurrence graph is then constructed from this correlation matrix M, which defines the graph edges and the node embeddings. This graph is then fed to a GNN which transforms such initial graph representation and outputs final learned node embeddings $\boldsymbol{H} \in \mathbb{R}^{N\times D}$.

Multilabel class logits $\boldsymbol{c}$ are obtained by multiplying these learned node embeddings $\boldsymbol{H}$ with the audio embedding module feature vector $\boldsymbol{f}$:

$$\boldsymbol{c} = \boldsymbol{H}\boldsymbol{f}^\top \tag{8}$$

IV. Experimental Setup and Results

In this study, we consider four different strategies for deriving the node embeddings of the label co-occurrence module. In detail, we compare one-hot vectors as used in [18], pre-trained GloVe word embeddings as used in [12], [14], [32], fastText word embeddings, as used in [12] and node2vec embeddings. These latter have not yet been explored for SEC tasks. We will compare the performance of various configurations for such node embeddings within the framework described in Sec. III.

A. *Dataset*

We perform our SEC experiments using FSD50K [21], a recently introduced dataset of sound event clips with over 100 hours of manually labeled audio and 200 classes, drawn from the larger AudioSet ontology [33]. More in detail, FSD50K contains 37134 audio clips for training, 4170 audio clips for validation, and 10231 audio clips for evaluation. The clips have variable length from 0.3s to 30s. We used the provided training, validation and test splits to respectively train, tune and test the models employed in our experiments.

B. *Feature Extraction*

Following [21], we resample FSD50K from the original 44.1 kHz samplerate to 22.050 kHz and extract 96-band, LFBEs to use as input features for the CRNN branch. To deal with variable-length clips, we feed to the CRNN LFBEs extracted from audio chunks of 1-second length with 50% overlap. LFBEs are computed with a Short-Time Fourier transform window size of 30 ms with 10 ms stride. With these settings, input CRNN features have shape $(f, t) = 96 \times 101$. The label associated with each chunk is inherited from the source clip label as the dataset is weakly labeled.

TABLE I
CLASSIFICATION PERFORMANCE FOR THE CONSIDERED ARCHITECTURE WITH DIFFERENT NODE EMBEDDINGS. FOR EACH MODEL, WE REPORT THE NUMBER OF TRAINABLE PARAMETERS. THE METRICS USED ARE DESCRIBED IN DETAIL IN SECTION IV-D

Model	node embedding	trainable param.	mAP	mAUC	d'	$l\omega lrap$
CRNN baseline	-	0.923M	0.3676	0.9307	2.0950	0.5113
CRNN-GGNN	200-dim one-hot	1.285M	0.3532	0.9264	2.0501	0.5252
CRNN-GGNN	300-dim GloVe	1.748M	0.3644	0.9266	2.0517	0.5114
CRNN-GGNN	300-dim fastText	1.748M	0.3696	0.9283	2.0599	0.5248
CRNN-GGNN	64-dim node2vec	0.943M	0.3752	0.9336	2.1254	0.5434
CRNN-GGNN	128-dim node2vec	1.062M	**0.4035**	**0.9340**	2.1310	0.5513
CRNN-GGNN	200-dim node2vec	1.285M	0.3551	0.9255	2.0990	0.5373
CRNN-GGNN	300-dim node2vec	1.748M	0.3725	0.9310	**2.1668**	**0.5516**

C. Networks structure and Node embedding

In the CRNN model we use 128 filters, $(5, 5)$ kernel size and unitary stride for all convolutional layers. The pooling sizes for the max-pooling layers are $(f, t) = (5, 2)$, $(4, 2)$ and $(3, 2)$. The bidirectional GRU block has an hidden layer with size 64. Regarding the graph co-occurrence branch, we employ a GGNN model composed of three hidden layers with node embeddings of the same size as input. We investigated several parameters involved in the *node2vec* algorithm, and found the best configuration to be as follows: $walk\_length = 20$, $num\_walks = 20$, return parameter $p = 0.1$ and in-out parameter $q = 1$.

D. Evaluation metrics

We use four evaluation metric scores, which are widely used in SEC [21]: mean Average Precision (mAP), mean Area Under the Curve (mAUC), and sensitivity index d-prime (d'); Mean Average Precision is an approximation of the area under the Precision-Recall curve, which is more informative of performance when dealing with imbalanced datasets. Similarly, mean Area Under the Curve (mAUC) metric is defined as the area under the ROC curve, averaged over all sound classes; d-prime index is closely connected to AUC; it is defined as the difference between z-scores of True Positive Rate (TPR) and False Positive Rate (FPR):

$$d' = z(\text{TPR}) - z(\text{FPR}). \tag{9}$$

Finally, the plain label-ranking average precision ($lrap$) measures the average precision of retrieving a ranked list of relevant labels for each test clip: the system ranks all the available labels, then the precisions of the ranked lists down to each true label are averaged. Here we employ the "label-weighted" variant which calculates the precision for each label in the test set and gives them all equal contribution to the final metric:

$$l\omega lrap = \frac{1}{\sum_s |C(s)|} \sum_s \sum_{c \in C(s)} lrap(c, s), \tag{10}$$

where $|C(s)|$ is the number of true classes for sample s.

E. Model Training

Models were trained up to 60 epochs with a random weight initialization for networks on both modules; learning rate was initially set to the value of $5 \cdot 10^{-4}$, and then halved if validation mAP is not improved within 5 epochs. Adam [34] was used as optimization algorithm, with L2 weight decay of $5 \cdot 10^{-4}$. Each model was trained with a binary cross-entropy loss with separate logits for each class, as we perform multi-label classification. We used a batch size of 256 to maximize the GPU utilization. Once the training is over, the model checkpoint with best validation mAP is selected and evaluated on the test set. For all graph-based models we tune the size of the node embedding dimension with respect to mAP obtained on the validation set. To improve generalization, during training, we employed *mixup* augmentation [35] with parameter λ drawn from a beta distribution $Beta(\alpha, \alpha)$ with $\alpha = 0.2$.

F. Results

We report our results in Table I. In detail we compare the CRNN classifier used alone (CRNN baseline) with the approach described in Section III, where we use a GGNN to learn co-occurrences prior knowledge as an additional information for SEC. In detail, for this latter approach, we compare different node embedding strategies while keeping the rest of the framework the same. Firstly we can see that using one-hot encoding, as in [18], improves only $l\omega lrap$ score with respect to the baseline and instead leads to a degradation of the other metrics. Secondly, we can observe that using GloVe pre-trained word embeddings for the nodes, as used in [17], leads to slighly lower performance for most metrics in our task. This could be due to the fact that FSD50K dataset is significantly more challenging than the dataset used in [17]. Instead, we found the proposed *node2vec* based approach to lead in general superior results both with respect to the CRNN baseline model used alone and with the other embeddings. The most noticeable increase in mAP and mAUC values are observed for the 128 *node2vec* embedding dimension (0.4035 and 0.9340 respectively) but a slight performance increase is observed already with a modest 64 embedding size model. Increasing further the embedding size leads to a degradation of mAP and mAUC scores and, in general, mixed results: the

200-dim model obtains worse results than the baseline model while the 300-dim model obtains the highest d' and $l\omega lrap$ figures. These are however only marginally better than ones obtained with 128 embedding size. In general we can conclude that using *node2vec* with 128 embedding size leads to the best trade-off between the number of trainable parameters and performance.

V. Conclusions

In this work we compared different node embedding strategies for ontology-aware SEC. Building from previous works, we adopt a SEC framework which is able to exploit information of sound events classes co-occurrence via a learned GNN-based module. This module is fed a graph whose nodes are labels embeddings and whose edges are defined by the various labels co-occurrences obtained from the training set. The output of this module is then combined with the output of a conventional CRNN architecture which is fed audio-related features. Using this framework, we compared different node embeddings strategies using the FSD50K dataset to perform our experiments. We show that *node2vec* node embeddings can outperform other embeddings strategies used in previous works on ontology-aware SEC. Our best node2vec-based method improves the SEC absolute performance up to 3.39% in terms of clip-level mAP score, 0.0711 points on sensitivity index (d') and 0.0265 points on lωlrap score, compared with the best competing embedding approach (fastText). Future works may include exploring different audio modules, as well as extending the technique to SED, by managing the event localization at frame-level and not at the audio clip-level.

References

[1] J. Salamon and J. P. Bello, "Deep convolutional neural networks and data augmentation for environmental sound classification," *IEEE Signal processing letters*, vol. 24, no. 3, pp. 279–283, 2017.

[2] H. Zhang, I. McLoughlin, and Y. Song, "Robust sound event recognition using convolutional neural networks," in *ICASSP*. IEEE, 2015, pp. 559–563.

[3] S. Abdoli, P. Cardinal, and A. L. Koerich, "End-to-end environmental sound classification using a 1d convolutional neural network," *Expert Systems with Applications*, vol. 136, pp. 252–263, 2019.

[4] Y. Wang, L. Neves, and F. Metze, "Audio-based multimedia event detection using deep recurrent neural networks," in *ICASSP*, 2016, pp. 2742–2746.

[5] T. H. Vu and J.-C. Wang, "Acoustic scene and event recognition using recurrent neural networks," *Detection and Classification of Acoustic Scenes and Events*, vol. 2016, pp. 1–3, 2016.

[6] A. Ö. Eren and M. Sert, "Audio captioning using gated recurrent units," *arXiv preprint arXiv:2006.03391*, 2020.

[7] Y. Xu, Q. Kong, W. Wang, and M. D. Plumbley, "Large-scale weakly supervised audio classification using gated convolutional neural network," in *ICASSP*, 2018, pp. 121–125.

[8] S. Adavanne, K. Drossos, E. Çakir, and T. Virtanen, "Stacked convolutional and recurrent neural networks for bird audio detection," in *EUSIPCO*, 2017, pp. 1729–1733.

[9] M. Malik, S. Adavanne, K. Drossos, T. Virtanen, D. Ticha, and R. Jarina, "Stacked convolutional and recurrent neural networks for music emotion recognition," *arXiv preprint arXiv:1706.02292*, 2017.

[10] C. Aironi, S. Cornell, E. Principi, and S. Squartini, "Graph-based representation of audio signals for sound event classification," in *EUSIPCO*. IEEE, 2021, pp. 566–570.

[11] X. Wang, Y. Ye, and A. Gupta, "Zero-shot recognition via semantic embeddings and knowledge graphs," in *IEEE conference on computer vision and pattern recognition*, 2018, pp. 6857–6866.

[12] Z.-M. Chen, X.-S. Wei, P. Wang, and Y. Guo, "Multi-label image recognition with graph convolutional networks," in *IEEE/CVF conference on computer vision and pattern recognition*, 2019, pp. 5177–5186.

[13] Y. Zhou, Y. Sun, and V. Honavar, "Improving image captioning by leveraging knowledge graphs," in *IEEE winter conference on applications of computer vision*, 2019, pp. 283–293.

[14] B. Chen, J. Li, G. Lu, H. Yu, and D. Zhang, "Label co-occurrence learning with graph convolutional networks for multi-label chest x-ray image classification," *IEEE journal of biomedical and health informatics*, vol. 24, no. 8, pp. 2292–2302, 2020.

[15] M. Allahyari, K. J. Kochut, and M. Janik, "Ontology-based text classification into dynamically defined topics," in *IEEE international conference on semantic computing*, 2014, pp. 273–278.

[16] Z. Wang, Q. Lv, X. Lan, and Y. Zhang, "Cross-lingual knowledge graph alignment via graph convolutional networks," in *Conference on empirical methods in natural language processing*, 2018, pp. 349–357.

[17] Y. Sun and S. Ghaffarzadegan, "An ontology-aware framework for audio event classification," in *ICASSP*. IEEE, 2020, pp. 321–325.

[18] H. Shrivastava, Y. Yin, R. R. Shah, and R. Zimmermann, "Mt-gcn for multi-label audio-tagging with noisy labels," in *ICASSP*, 2020, pp. 136–140.

[19] A. Grover and J. Leskovec, "node2vec: Scalable feature learning for networks," in *ACM SIGKDD international conference on Knowledge discovery and data mining*, 2016, pp. 855–864.

[20] P. Bojanowski, E. Grave, A. Joulin, and T. Mikolov, "Enriching word vectors with subword information," *arXiv preprint arXiv:1607.04606*, 2016.

[21] E. Fonseca, X. Favory, J. Pons, F. Font, and X. Serra, "Fsd50k: an open dataset of human-labeled sound events," *arXiv preprint arXiv:2010.00475*, 2020.

[22] M. Gori, G. Monfardini, and F. Scarselli, "A new model for learning in graph domains," in *International Joint Conference on Neural Networks*, vol. 2. IEEE, 2005, pp. 729–734.

[23] F. Scarselli, M. Gori, A. C. Tsoi, M. Hagenbuchner, and G. Monfardini, "The graph neural network model," *IEEE transactions on neural networks*, vol. 20, no. 1, pp. 61–80, 2008.

[24] A. Micheli, "Neural network for graphs: A contextual constructive approach," *IEEE Transactions on Neural Networks*, vol. 20, no. 3, pp. 498–511, 2009.

[25] M. Niepert, M. Ahmed, and K. Kutzkov, "Learning convolutional neural networks for graphs," in *ICML*. PMLR, 2016, pp. 2014–2023.

[26] J. Gilmer, S. S. Schoenholz, P. F. Riley, O. Vinyals, and G. E. Dahl, "Neural message passing for quantum chemistry," in *ICML*. PMLR, 2017, pp. 1263–1272.

[27] C. Gallicchio and A. Micheli, "Graph echo state networks," in *IJCNN*. IEEE, 2010, pp. 1–8.

[28] Y. Li, D. Tarlow, M. Brockschmidt, and R. Zemel, "Gated graph sequence neural networks," *arXiv preprint arXiv:1511.05493*, 2015.

[29] K. Cho, B. Van Merriënboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio, "Learning phrase representations using rnn encoder-decoder for statistical machine translation," *arXiv preprint arXiv:1406.1078*, 2014.

[30] H. Cai, V. W. Zheng, and K. C.-C. Chang, "A comprehensive survey of graph embedding: Problems, techniques, and applications," *IEEE Transactions on Knowledge and Data Engineering*, vol. 30, no. 9, pp. 1616–1637, 2018.

[31] E. Cakır, G. Parascandolo, T. Heittola, H. Huttunen, and T. Virtanen, "Convolutional recurrent neural networks for polyphonic sound event detection," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 25, no. 6, pp. 1291–1303, 2017.

[32] H. Wang, Y. Zou, D. Chong, and W. Wang, "Modeling label dependencies for audio tagging with graph convolutional network," *IEEE Signal Processing Letters*, vol. 27, pp. 1560–1564, 2020.

[33] J. F. Gemmeke, D. P. Ellis, D. Freedman, A. Jansen, W. Lawrence, R. C. Moore, M. Plakal, and M. Ritter, "Audio set: An ontology and human-labeled dataset for audio events," in *ICASSP*. IEEE, 2017, pp. 776–780.

[34] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[35] H. Zhang, M. Cisse, Y. N. Dauphin, and D. Lopez-Paz, "mixup: Beyond empirical risk minimization," *arXiv preprint arXiv:1710.09412*, 2017.