Phase-field Models for Lightweight Graph Convolutional Networks

Hichem Sahbi
Sorbonne University, CNRS, LIP6, F-75005, Paris, France
hichem.sahbi@sorbonne-universite.fr

Abstract

In this paper, we design lightweight graph convolutional networks (GCNs) using a particular class of regularizers, dubbed as phase-field models (PFMs). PFMs exhibit a bi-phase behavior using a particular ultra-local term that allows training both the topology and the weight parameters of GCNs as a part of a single "end-to-end" optimization problem. Our proposed solution also relies on a reparametrization that pushes the mask of the topology towards binary values leading to effective topology selection and high generalization while implementing any targeted pruning rate. Both masks and weights share the same set of latent variables and this further enhances the generalization power of the resulting lightweight GCNs. Extensive experiments conducted on the challenging task of skeleton-based recognition show the outperformance of PFMs against other staple regularizers as well as related lightweight design methods.

1. Introduction

Deep convolutional networks are nowadays becoming mainstream in solving many pattern classification tasks including visual recognition [2–6]. Their principle consists in training convolutional filters together with pooling and attention mechanisms that maximize classification performances. Many existing convolutional networks were initially dedicated to grid-like data, including images [9, 10]. However, data sitting on top of irregular domains (such as skeleton graphs in action recognition [47, 52, 66]) require extending convolutional networks to general graph structures, and these extensions are known as graph convolutional networks (GCNs) [7, 8]. Two families of GCNs exist in the literature: spectral and spatial. The former achieve convolutions using Fourier [11–16] whilst the latter are based on message passing, via attention matrices, prior to convolution [17–22]. Whereas spatial GCNs have been relatively more effective compared to spectral ones, their precision is highly reliant on the accuracy of the attention matrices that capture context and node-to-node relationships [48]. With multi-head attention, GCNs are more accurate but overparametrized and computationally overwhelming.

Many solutions are proposed in the literature in order to reduce time and memory footprint of convolutional networks including GCNs. Some of them pretrain oversized networks prior to reduce their computational complexity (using distillation [23–29, 61], tensor decomposition [39, 40], quantization [30, 41–45] and pruning [31–37, 46]), while others build efficient networks from scratch using neural architecture search [58]. In particular, pruning methods, either unstructured or structured are currently becoming mainstream. Their principle consists in removing connections whose impact on the classification performance is the least noticeable. Structured pruning [33, 36, 38] consists in removing groups of connections, entire filters, etc., and this makes the class of learnable subnetworks highly rigid. In contrast, unstructured pruning [30, 34] is more flexible and proceeds by dropping-out connections individually using different proxy criteria, such as weight magnitude [34] or using more sophisticated variational methods [50, 54, 55].

The general recipe of variational pruning consists in learning both the weights and the binary masks that capture the topology of the pruned subnetworks. This is achieved by minimizing an objective function that combines (via a mixing hyperparameter) a classification loss and a regularizer which controls the sparsity of the resulting masks [36, 49, 51]. However, these methods are powerless to implement any given targeted pruning rate (cost) without overtrying multiple settings of the mixing hyperparameters. Alternative variational methods model explicitly the cost, using ℓ_0-based criteria [51, 53], in order to minimize the discrepancy between the observed cost and the targeted one. Nonetheless, the underlying optimization problems are highly combinatorial and existing solutions usually rely on sampling heuristics. Existing more tractable relaxation (such as ℓ_1/ℓ_2-based, etc. [56, 57, 59]) promote sparsity, but are powerless to implement any given target cost *exactly*, and also result into overpruning effects leading to disconnected subnetworks, with weak generalization, especially at very high pruning regimes. Besides, most of the existing pruning solutions decouple the training of network topology

(masks) from weights, and this doubles the number of training parameters and increases the risk of overfitting. Finally, the mainstream magnitude pruning [34] allows reaching any targeted cost, but relies on a tedious fine-tuning step and also decouples the training of topology from weights and this makes training clearly suboptimal.

Considering all these issues, we introduce in this paper a new lightweight network design based on the phase-field model (PFM). The latter gathers the upsides of the aforementioned regularization methods while discarding their downsides at some extent. PFM is based on an ultra-local term with two local minima around 0 and 1; when composed with a particular mask reparametrization, PFM promotes sparsity by pushing the values of this reparametrization towards crisp (binary) values without any temperature annealing. In other words, the proposed method allows generating only feasible solutions (i.e., binary masks) while implementing any targeted pruning rate without overtrying multiple mixing hyperparameters. The proposed solution also avoids the decoupling of weights and masks, and this reduces the number of training parameters, and also the risk of overfitting. Experiments conducted on the challenging task of action and hand-gesture recognition show a consistent gain of the proposed PFM-based approach against staple regularizers and cost-sensitive variational methods as well as the related work including magnitude pruning.

2. A Glimpse on GCNs

Let $\{\mathcal{G}_i = (\mathcal{V}_i, \mathcal{E}_i)\}_i$ denote a collection of graphs with $\mathcal{V}_i$, $\mathcal{E}_i$ being respectively the nodes and edges of $\mathcal{G}_i$. Each graph $\mathcal{G}_i$ (denoted for short as $\mathcal{G} = (\mathcal{V}, \mathcal{E})$) is endowed with a signal $\{\phi(u) \in \mathbb{R}^s : u \in \mathcal{V}\}$ and associated with an adjacency matrix $\mathbf{A}$. GCNs aim at learning a set of C filters $\mathcal{F}$ that define convolution on n nodes of $\mathcal{G}$ (with $n = |\mathcal{V}|$) as $(\mathcal{G} \star \mathcal{F})_{\mathcal{V}} = f(\mathbf{A}\,\mathbf{U}^\top\,\mathbf{W})$, here $^\top$ stands for transpose, $\mathbf{U} \in \mathbb{R}^{s\times n}$ is the graph signal, $\mathbf{W} \in \mathbb{R}^{s\times C}$ is the matrix of convolutional parameters corresponding to the C filters and $f(.)$ is a nonlinear activation applied entry-wise. In this convolution, the input signal $\mathbf{U}$ is projected using $\mathbf{A}$ and this provides for each node u, the aggregate set of its neighbors. Entries of $\mathbf{A}$ could be handcrafted or learned so $(\mathcal{G} \star \mathcal{F})_{\mathcal{V}}$ implements a convolutional block with two layers; the first one aggregates signals in $\mathcal{N}(\mathcal{V})$ (sets of node neighbors) by multiplying $\mathbf{U}$ with $\mathbf{A}$ while the second layer achieves convolution by multiplying the resulting aggregates with the C filters in $\mathbf{W}$. Learning multiple adjacency (also referred to as attention) matrices (denoted as $\{\mathbf{A}^k\}_{k=1}^K$) allows us to capture different contexts and graph topologies when achieving aggregation and convolution. With multiple matrices $\{\mathbf{A}^k\}_k$ (and associated convolutional filter parameters $\{\mathbf{W}^k\}_k$), $(\mathcal{G} \star \mathcal{F})_{\mathcal{V}}$ is updated as $f\big(\sum_{k=1}^K \mathbf{A}^k\mathbf{U}^\top\mathbf{W}^k\big)$. Stacking aggregation and convolutional layers, with multiple matrices $\{\mathbf{A}^k\}_k$, makes GCNs accurate but heavy. We propose, in what follows, a method that makes our networks lightweight and still effective.

3. Lightweight Design

In the rest of this paper, a given GCN is subsumed as a multi-layered neural network g_θ whose weights defined as $\theta = \big\{\mathbf{W}^1, \ldots, \mathbf{W}^L\big\}$, with L being its depth, $\mathbf{W}^\ell \in \mathbb{R}^{d_{\ell-1}\times d_\ell}$ its ℓ^{th} layer weight tensor, and d_ℓ the dimension of ℓ. The output of a given layer ℓ is defined as $\phi^\ell = f_\ell(\mathbf{W}^{\ell^\top}\,\phi^{\ell-1})$, $\ell \in \{2, \ldots, L\}$, being f_ℓ an activation function; without a loss of generality, we omit the bias in the definition of ϕ^ℓ. Pruning consists in zeroing-out a subset of weights in θ by multiplying $\mathbf{W}^\ell$ with a binary mask $\mathbf{M}^\ell \in \{0,1\}^{d_{\ell-1}\times d_\ell}$. The binary entries of $\mathbf{M}^\ell$ are set depending on whether the underlying layer connections are kept or removed, so $\phi^\ell = f_\ell((\mathbf{M}^\ell \odot \mathbf{W}^\ell)^\top\,\phi^{\ell-1})$, here $\odot$ stands for the element-wise matrix product. In this definition, entries of the tensor $\{\mathbf{M}^\ell\}_\ell$ are set depending on the prominence of the underlying connections in g_θ. However, such pruning suffers from several drawbacks. On the one hand, optimizing the discrete set of variable $\{\mathbf{M}^\ell\}_\ell$ is known to be highly combinatorial and intractable especially on large networks. On the other hand, the total number of parameters $\{\mathbf{M}^\ell\}_\ell$, $\{\mathbf{W}^\ell\}_\ell$ is twice the number of connections in g_θ and this increases training complexity and may also lead to overfitting. In order to circumvent these issues, we consider an alternative *reparametrization* that allows finding both the topology of the pruned networks together with their weights, without doubling the size of the training parameters, while making learning still effective.

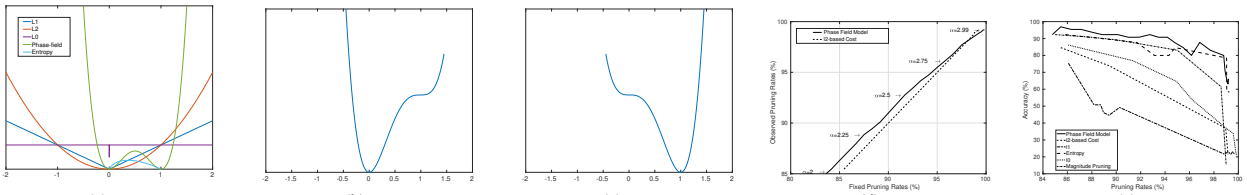
3.1. Weight Reparametrization

We consider an alternative parametrization of the network related to magnitude pruning. This reparametrization corresponds to the Hadamard product involving a weight tensor and a function applied entry-wise to the same tensor

$$\mathbf{W}^\ell = \hat{\mathbf{W}}^\ell \odot \psi(\hat{\mathbf{W}}^\ell). \tag{1}$$

In the above equation, $\hat{\mathbf{W}}^\ell$ is a latent tensor and $\psi(\hat{\mathbf{W}}^\ell)$ is a continuous relaxation of $\mathbf{M}^\ell$ which enforces the prior that smallest weights should be removed from the network. In order to achieve this goal, ψ must be (i) bounded in $[0, 1]$, (ii) differentiable, (iii) symmetric, and (iv) $\psi(\omega) \rightsquigarrow 1$ when $|\omega|$ is sufficiently large and $\psi(\omega) \rightsquigarrow 0$ otherwise. The first and the fourth properties ensure that the reparametrization is neither acting as a scaling factor greater than one nor changing the sign of the latent weight, and also acts as the identity for sufficiently large weights, and as a contraction factor for small ones. The second property is necessary to ensure that ψ has computable gradient while the third condition guarantees that only the magnitudes of the latent weights matter[1].

[1] A possible choice, used in practice, that satisfies these four conditions (when combined with PFM) is $\psi(\omega) = 2\sigma(\omega^2) - 1$ with σ being the sigmoid function.



(a) (b) (c) (d) (e)

Figure 1. (a) This figure shows a comparison of different regularizers, namely ℓ_0, ℓ_1, ℓ_2 and entropy as well as the balanced PFM (i.e., $\alpha = 0$). From these curves, it is clear that our PFM gathers the advantages of all these regularizers: (i) it strongly penalizes large mask values (as ℓ_2), (ii) it (not only) pushes mask values near 0 (as all the regularizers), but also near 1 (as ℓ_0 and entropy) which allows counting non-zeros while (iii) behaving as a relaxed differentiable function (as ℓ_1), (iv) finally, similarly to entropy, PFM has also two local minima between 0 and 1, however entropy does not penalize large mask values and does not allow reaching any (a priori) fixed pruning rate. (b and c) These figures show the two imbalanced versions of PFM (corresponding to $\alpha > 0$ and $\alpha < 0$) that allow implementing over and under-pruning respectively. In all experiments, β is arbitrarily fixed to 3 while α is accordingly chosen depending on the targeted pruning rate tpr $= \frac{\alpha+\beta}{2\beta}$ (which corresponds to the local maximum of the ultra-local term). (d) This figure shows the alignment between the fixed and the observed pruning rates when using cost-sensitive pruning and PFMs on the SBU dataset. (e) The underlying accuracy w.r.t. different pruning rates. Note that $\beta = 3$ in all the PFMs. **(Better to zoom the PDF version).**

Note that the fourth property is implemented without any ill-posed temperature annealing, but instead using a phase-field model – presented subsequently – which controls the smoothness of ψ around the support of the latent weights. Put differently, the asymptotic behavior of ψ – that allows selecting the topology of the pruned subnetworks – is obtained using the phase-field energy as described below.

3.2. Phase-field Model

A phase-field is a real-valued function defined on an input domain $\Omega \subset \mathbb{R}$ [64]. A phase-field determines a region by the map $\xi_z(\psi) = \{\omega \in \Omega : \psi(\omega) > z\}$ (where z is a given threshold) and a phase-field energy as

$$E_P(\psi) = \int_\Omega V(\psi(\omega))\, d\omega, \tag{2}$$

here $V(t)$ - referred to as the ultra-local term - is given by

$$\beta\left(\frac{(2t-1)^4}{4} - \frac{(2t-1)^2}{2}\right) + \alpha\left(2t - 1 - \frac{(2t-1)^3}{3}\right). \tag{3}$$

If one minimizes (2) subject to $\xi_z(\psi) = \mathcal{R}$ for a fixed region $\mathcal{R}$, then away from the boundary $\partial\mathcal{R}$, the minimizing function (denoted as $\psi_\mathcal{R}$) assumes approximately the value +1 inside, and 0 outside $\mathcal{R}$ thanks to the ultra-local term, and it varies smoothly (depending on β) across the interface near $\partial\mathcal{R}$. Considering a discrete approximation of the integral on a finite set of parameters $\{\omega_i\}_i$, one may rewrite $E_P(\psi)$ as $E_P(\psi(\{\omega_i\}_i)) = \sum_i V(\psi(\omega_i))$. In order to guarantee two energy minima at 0 and +1 associated to the two classes (pruned/unpruned), the inequality $\beta > |\alpha|$ must be satisfied, so $V'(1) = V'(0) = 0$ and $V''(1) = V''(0) > 0$ where $'$ and $''$ denote the first and second derivatives respectively. Notice that the formulation of our PFM yields to choose the threshold z to be at the maximum $(\beta + \alpha)/2\beta$ of V which also corresponds to the fixed pruning rate (see Fig. 1-abc). When setting $\alpha = 0$ and $\beta > 0$, we get the particular case of PFM; this leads to $V(1) = V(0)$ corresponding to equiprobable phases $\{0, 1\}$, and hence E_P is suitable for balanced pruning. In contrast, for significantly imbalanced pruning (which is the main scope of this paper), one should select $\alpha \neq 0$ so that the two phases $\{0, 1\}$ will have different energies: $V(1) - V(0) = 4\alpha/3 \neq 0$. In other words, a strictly positive α allows implementing imbalanced over-pruning and vice-versa (see again Fig. 1-abc).

3.3. Variational Pruning

Pruning is achieved using a global loss as a combination of cross-entropy $\mathcal{L}_e$, and phase-field energy E_P (which controls the cost and aims at zeroing as much mask entries as possible depending on the setting of α) resulting into

$$\min_{\{\hat{\mathbf{W}}^\ell\}_\ell} \mathcal{L}_e\big(\{\hat{\mathbf{W}}^\ell \odot \psi(\hat{\mathbf{W}}^\ell)\}_\ell\big) \;+\; \lambda\, E_P(\{\psi(\hat{\mathbf{W}}^\ell)\}_\ell), \tag{4}$$

here λ is sufficiently large (overestimated in practice), so Eq. (4) focuses on binarizing $\{\psi(\hat{\mathbf{W}}^\ell)\}_\ell$ using the phase-field energy, and also constraining the pruning rate to reach $\frac{\alpha+\beta}{2\beta}$. As training evolves, E_P reaches its minimum and stabilizes while the gradient of the global loss becomes dominated by the gradient of $\mathcal{L}_e$, and this maximizes further the classification performances.

4. Experiments

In this section, we evaluate the performances of our pruned GCNs on skeleton-based recognition using two challenging datasets, namely SBU [60] and FPHA [62]. SBU is an interaction dataset acquired using the Microsoft Kinect sensor; it includes in total 282 moving skeleton sequences (performed by two interacting individuals) belonging to 8 categories. Each pair of interacting individuals corresponds to two 15 joint skeletons and each joint is characterized with a sequence of its 3D coordinates across video frames. In this dataset, we consider the same evaluation protocol as the one suggested in the original dataset release [60] (i.e., train-test split). The FPHA dataset includes 1175 skeletons belonging to 45 action categories with high inter and intra subject variability. Each skeleton includes 21 hand joints and each joint is again characterized with a

sequence of its 3D coordinates across video frames. We evaluate the performance of our method following the protocol in [62]. In all these experiments, we report the average accuracy over all the classes of actions.

Implementation details. We trained the GCNs end-to-end using the Adam optimizer [1] for 2,700 epochs with a batch size equal to 200 for SBU and 600 for FPHA, a momentum of 0.9 and a global learning rate (denoted as $\nu(t)$) inversely proportional to the speed of change of the loss used to train our networks. When this speed increases (resp. decreases), $\nu(t)$ decreases as $\nu(t) \leftarrow \nu(t-1) \times 0.99$ (resp. increases as $\nu(t) \leftarrow \nu(t-1)/0.99$). In all these experiments, we use a GeForce GTX 1070 GPU (with 8 GB memory). The architecture of our baseline GCN includes an attention layer of 1 head on SBU (resp. 16 heads on FPHA) applied to skeleton graphs whose nodes are encoded with 8-channels (resp. 32 for FPHA), followed by a convolutional layer of 32 filters for SBU (resp. 128 filters for FPHA), and a dense fully connected layer and a softmax layer. The initial network for SBU is not heavy, its number of parameters does not exceed 15,320, and this makes its pruning challenging as many connections will be isolated (not contributing in the evaluation of the network output). In contrast, the initial network for FPHA is relatively heavy (for a GCN) and its number of parameters reaches 2 millions. As shown subsequently, both GCNs are accurate compared to the related work on the SBU/FPHA benchmarks. Considering these GCN baselines, our goal is to make them highly lightweight and as accurate as possible.

Model analysis and comparison. Fig 1-d shows the alignment between the fixed/targeted pruning rates (tpr) and the observed ones when using PFM and its comparison against cost-sensitive pruning. In these experiments, PFM acts not only as a regularizer (and binarizer) but also as a rebalancing function which allows implementing any tpr by choosing α that satisfies $\frac{\alpha+\beta}{2\beta} = \text{tpr}$ or equivalently $\alpha = 2\beta \times \text{tpr} - \beta$. Fig. 1-e shows the accuracy of our lightweight GCNs w.r.t. the underlying pruning rates. In these results, PFM is compared against different *alternative* regularizers (plugged in Eq. 4 instead of PFM), namely ℓ_0 [51], ℓ_1 [65], entropy [67] and ℓ_2-based cost-sensitive pruning. From these results, the impact of PFM is substantial on highly pruned GCNs while relatively smaller pruning regimes provide equivalent performances. Note that when alternative regularizers are used, multiple settings (trials) of the underlying hyperparameter λ (in Eq. 4) are necessary prior to reach any targeted pruning rate, and this makes the whole pruning process overwhelming. While cost-sensitive pruning makes training more tractable, its downside resides in the collapse of trained masks, and this degrades performances significantly at high pruning rates; a similar behavior is observed with magnitude pruning (see again Fig. 1-e).

Table 1 shows an ablation study (and extra comparisons) of our PFM when used individually and jointly with the other regularizers as well as cost-sensitive pruning. From these results, we first observe that when training is achieved with weight reparametrization, performances are equivalent and sometimes overtake the initial heavy GCN, with less parameters (pruning rate does not exceed 70% as no control on tpr is achieved) as this produces a regularization effect similar to [63]. Second, we observe a positive impact of PFM when jointly combined with the aforementioned regularizers and cost-sensitive loss; note that when PFM is jointly used, α is set to 0, so tpr $= 0.5$ and this makes the rebalancing effect of PFM null, and only the other regularizers allow implementing the targeted pruning rates when λ is appropriately tuned. Finally, extra comparison against magnitude pruning [30] shows the substantial gain of our PFM at very high pruning regimes.

Datasets	Methods	Pruning rates (%)	# Parameters	Accuracy (%)
SBU	Initial Model	0.0	15320	90.76
	Weight Reparametrization (WR)	70.66	4494	**93.84**
	Magnitude Pruning	98.58	216	61.53
	WR+ℓ_0	99.00	152	36.92
	WR+ℓ_0+PFM ($\alpha = 0$)	99.05	144	**55.38**
	WR+ℓ_1	98.87	171	21.53
	WR+ℓ_1+PFM ($\alpha = 0$)	98.94	161	**73.84**
	WR+Entropy	98.97	157	60.00
	WR+Entropy+PFM ($\alpha = 0$)	98.96	158	**61.53**
	WR+ℓ_2-based Cost	98.96	158	36.92
	WR+PFM ($\alpha = 2\beta\text{tpr} - \beta$)	98.98	154	67.69
	WR+ℓ_2-based Cost+PFM ($\alpha = 0$)	98.96	158	**75.38**
FPHA	Initial Model	0.0	1967616	86.08
	Weight Reparametrization (WR)	50.38	976268	85.56
	Magnitude Pruning	98.83	22892	52.69
	WR+ℓ_0	99.24	14858	8.34
	WR+ℓ_0+PFM ($\alpha = 0$)	99.43	11203	**64.69**
	WR+ℓ_1	99.26	14460	2.78
	WR+ℓ_1+PFM ($\alpha = 0$)	99.26	14460	**70.78**
	WR+Entropy	99.09	17788	31.13
	WR+Entropy+PFM ($\alpha = 0$)	99.25	14683	**67.47**
	WR+ℓ_2-based Cost	99.49	9945	5.56
	WR+PFM ($\alpha = 2\beta\text{tpr} - \beta$)	99.68	6156	65.91
	WR+ℓ_2-based Cost+PFM ($\alpha = 0$)	99.49	10034	**69.91**

Table 1. Ablation study of our pruning method (w and w/o PFMs). When PFMs are combined with other regularizers, α is necessarily equal to 0, so only the regularization effect is considered (as the other regularizers indirectly control the pruning rate). When PFMs are individually used, $\alpha = 2\beta\text{tpr} - \beta$ where tpr corresponds to the targeted pruning rate. In these results, PFMs are also compared against weight reparametrization and magnitude pruning. It's worth noticing that low accuracies result from the disconnected pruned networks obtained at high pruning regimes.

5. Conclusion

In this paper, we introduce a novel pruning method based on phase-field models (PFMs) which allow training very lightweight GCNs at very high pruning regimes. The strength of PFMs resides in their ability to leverage the advantage of different regularizers used in variational pruning while discarding their inconveniences at some extent. Indeed, the proposed PFMs allow training highly overpruned networks, binarizing the underlying masks while implementing any targeted pruning rate and improving generalization. Extensive experiments conducted on the challenging task of skeleton-based recognition show the substantial gain of our pruned lightweight networks against different baselines as well as the related work. As a future work, we are currently investigating the extension of this method to other network architectures and datasets.

References

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

[2] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in NIPS, 2012.

[3] H. Sahbi and N. Boujemaa. "From coarse to fine skin and face detection." Proceedings of the eighth ACM international conference on Multimedia. 2000.

[4] M. Jiu and H. Sahbi. "Laplacian deep kernel learning for image annotation." 2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP). IEEE, 2016.

[5] M. Jiu and H. Sahbi. "Nonlinear deep kernel learning for image annotation." IEEE Transactions on Image Processing 26.4 (2017): 1820-1832.

[6] M. Jiu and H. Sahbi. "Deep representation design from deep kernel networks." Pattern Recognition 88 (2019): 447-457.

[7] J. Bruna, W. Zaremba, A. Szlam, Y. LeCun. Spectral networks and locally connected networks on graphs. arXiv:1312.6203 (2013).

[8] M. Henaff, J. Bruna, Y. LeCun. Deep convolutional networks on graph structured data. arXiv preprint arXiv:1506.05163 (2015).

[9] K. He, X. Zhang, S. Ren, and J. Sun. Deep Residual Learning for Image Recognition. In CVPR, pages 770-778, June 2016.

[10] R. Girshick. Fast R-CNN. In ICCV, pages 1440-1448, 2015

[11] A. Mazari and H. Sahbi. "MLGCN: Multi-Laplacian graph convolutional networks for human action recognition." The British Machine Vision Conference (BMVC). 2019.

[12] TN. Kipf, M. Welling. Semi-supervised classification with graph convolutional networks. In ICLR, 2017

[13] R. Levie, F. Monti, X. Bresson, M.M. Bronstein. Cayleynets: Graph convolutional neural networks with complex rational spectral filters. IEEE Transactions on Signal Processing 67(1), 97–109 (2018)

[14] R. Li, S. Wang, F. Zhu, J. Huang. Adaptive graph convolutional neural networks. In AAAI, 2018.

[15] H. Sahbi. "Learning laplacians in chebyshev graph convolutional networks." Proceedings of the IEEE/CVF International Conference on Computer Vision. 2021.

[16] M. Defferrard et al. Convolutional Neural Networks on graphs with Fast Localized Spectral Filtering. In NIPS, 2016

[17] H. Sahbi. "Learning connectivity with graph convolutional networks." 2020 25th International Conference on Pattern Recognition (ICPR). IEEE, 2021.

[18] M. Gori, G. Monfardini, F. Scarselli. A new model for learning in graph domains. In IEEE IJCNN, vol. 2, pp. 729–734, 2005.

[19] A. Micheli. Neural network for graphs: A contextual constructive approach. IEEE TNN 20(3), 498-511 (2009)

[20] Z. Wu, S. Pan, F. Chen, G. Long, C. Zhang, P.S. Yu. A comprehensive survey on graph neural networks. arXiv:1901.00596 (2019).

[21] W. Hamilton, Z. Ying, J. Leskovec. Inductive representation learning on large graphs. In NIPS. pp. 1024–1034 (2017).

[22] H. Sahbi. "Kernel-based graph convolutional networks." 2020 25th International Conference on Pattern Recognition (ICPR). IEEE, 2021.

[23] G. E. Hinton, O. Vinyals, and J. Dean, "Distilling the knowledge in a neural network," *CoRR*, vol. abs/1503.02531, 2015.

[24] S. Zagoruyko and N. Komodakis, "Paying more attention to attention: Improving the performance of convolutional neural networks via attention transfer," in *ICLR*, 2017.

[25] A. Romero, N. Ballas, S. Ebrahimi Kahou, A. Chassang, C. Gatta, and Y. Bengio, "Fitnets: Hints for thin deep nets," in *ICLR*, 2015.

[26] S.-I. Mirzadeh et al. "Improved knowledge distillation via teacher assistant," in *AAAI*, 2020.

[27] Y. Zhang, T. Xiang, T. M. Hospedales, and H. Lu, "Deep mutual learning," in *CVPR*, 2018.

[28] S. Ahn, S. X. Hu, A. C. Damianou, N. D. Lawrence, and Z. Dai, "Variational information distillation for knowledge transfer," in *CVPR*, 2019.

[29] Sahbi, Hichem. "Coarse-to-fine deep kernel networks." Proceedings of the IEEE International Conference on Computer Vision Workshops. 2017.

[30] S. Han, H. Mao, and W. J. Dally, "Deep compression: Compressing deep neural network with pruning, trained quantization and huffman coding," in *ICLR*, 2016.

[31] H. Sahbi. "Lightweight Connectivity In Graph Convolutional Networks For Skeleton-Based Recognition." 2021 IEEE International Conference on Image Processing (ICIP). IEEE, 2021.

[32] B. Hassibi and D. G. Stork, "Second order derivatives for network pruning: Optimal brain surgeon," in *NIPS*, 1992.

[33] H. Li, A. Kadav, I. Durdanovic, H. Samet, and H. P. Graf, "Pruning filters for efficient convnets," in *ICLR*, 2017.

[34] S. Han, J. Pool, J. Tran, and W. J. Dally, "Learning both weights and connections for efficient neural network," in *NIPS*, 2015.

[35] Y. LeCun, J. S. Denker, and S. A. Solla, "Optimal brain damage," in *NIPS*, 1989.

[36] Z. Liu, J. Li, Z. Shen, G. Huang, S. Yan, and C. Zhang, "Learning efficient convolutional networks through network slimming," in *ICCV*. 2017, IEEE Computer Society.

[37] H. Sahbi. "Topologically-Consistent Magnitude Pruning for Very Lightweight Graph Convolutional Networks." 2022 IEEE International Conference on Image Processing (ICIP). IEEE, 2022.

[38] Zhao, Chenglong, et al. "Variational convolutional neural network pruning." In IEEE/CVF CVPR, 2019.

[39] E-L. Denton, W. Zaremba, J. Bruna, Y. LeCun, and R. Fergus. Exploiting linear structure within convolutional networks for efficient evaluation. In NIPS, pages 1269–1277, 2014

[40] W. Wang, Y. Sun, B. Eriksson, W. Wang, and V. Aggarwal. Wide compression: Tensor ring nets. In IEEE CVPR, pages 9329–9338, 2018.

[41] M. Courbariaux, I. Hubara, D. Soudry, R. El-Yaniv, and Y. Bengio. Binarized neural networks: Training deep neural networks with weights and activations constrained to+ 1 or-1. arXiv preprint arXiv:1602.02830, 2016

[42] S. Gupta, A. Agrawal, K. Gopalakrishnan, and P. Narayanan. Deep learning with limited numerical precision. In International Conference on Machine Learning, pages 1737–1746, 2015

[43] W. Chen, J. Wilson, S. Tyree, K. Weinberger, and Y. Chen. Compressing neural networks with the hashing trick. In ICML, 2015

[44] M. Rastegari, V. Ordonez, J. Redmon, and A. Farhadi. Xnor-net: Imagenet classification using binary convolutional neural networks. In ECCV, pages 525–542. Springer, 2016

[45] E. Park, J. Ahn, and S. Yoo. Weighted entropy based quantization for deep neural networks. In IEEE CVPR, 2017

[46] F. Tung and G. Mori. Clip-q: Deep network compression learning by in-parallel pruning-quantization. In IEEE CVPR, pages 7873–7882, 2018

[47] L. Wang and H. Sahbi. "Nonlinear cross-view sample enrichment for action recognition." Computer Vision-ECCV 2014 Workshops: Zurich, Switzerland.

[48] P. Veličković, G. Cucurull, A. Casanova, A. Romero, P. Lio, and Y. Bengio (2017). Graph attention networks. arXiv preprint arXiv:1710.10903.

[49] W. Wen, C. Wu, Y. Wang, Y. Chen, and H. Li. Learning structured sparsity in deep neural networks. In proc of NIPS, 2016.

[50] D. Kingma, P. Durk, T. Salimans, and M. Welling. "Variational dropout and the local reparameterization trick." In NIPS 28 (2015).

[51] C. Louizos, M. Welling, and D. Kingma. Learning sparse neural networks through l0 regularization. In proc. of ICLR, 2018

[52] L. Wang and H. Sahbi. "Bags-of-daglets for action recognition." 2014 IEEE International Conference on Image Processing (ICIP). IEEE, 2014.

[53] W. Pan, H. Dong, and Y. Guo. Dropneuron: Simplifying the structure of deep neural networks. In arXiv preprint arXiv:1606.07326, 2016

[54] P. David, Wipf, B. Dai, C. Zhu. Compressing neural networks using the variational information bottleneck. proc. of ICML, 2018.

[55] D. Molchanov, A. Ashukha, and D. Vetrov. Variational dropout sparsifies deep neural networks. proc of ICML, 2017.

[56] A. Gordon, E. Eban, O. Nachum, B. Chen, H. Wu, T.J. Yang, and E. Choi. Morphnet: Fast and simple resource constrained structure learning of deep networks. In Proc. of CVPR, 2018

[57] M. A Carreira-Perpin and Y. Idelbayev. Learning compression algorithms for neural net pruning. In Proc. of CVPR,pages 8532–8541, 2018

[58] H. Pham, M-Y. Guan, B. Zoph, Q-V. Le, and J. Dean. Efficient neural architecture search via parameter sharing. arXiv preprint arXiv:1802.03268, 2018

[59] C. Lemaire, A. Achkar, and P-M. Jodoin. "Structured pruning of neural networks with budget-aware regularization." Proceedings of the IEEE/CVF CVPR, 2019.

[60] K. Yun, J. Honorio, D. Chattopadhyay, T-L. Berg, and D. Samaras, CVPR Workshop, 2012.

[61] H. Sahbi. Coarse-to-fine support vector machines for hierarchical face detection. Diss. PhD thesis, Versailles University, 2003.

[62] G. Garcia-Hernando, S. Yuan, S. Baek, and T.K. Kim. First Person Hand Action Benchmark with RGB-D Videos and 3D Hand Pose Annotations. In CVPR, 2018.

[63] Wan, Li, et al. "Regularization of neural networks using dropconnect." International conference on machine learning. PMLR, 2013.

[64] P-C. Hohenberg and B-I. Halperin. "Theory of dynamic critical phenomena." Reviews of Modern Physics 49.3 (1977): 435.

[65] B. Koneru, N. Girish, and V. Vasudevan. "Sparse artificial neural networks using a novel smoothed LASSO penalization." IEEE TCS II: Express Briefs 66.5 (2019): 848-852.

[66] A. Mazari and H. Sahbi. "Deep temporal pyramid design for action recognition." ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP). IEEE, 2019.

[67] Wiedemann, Simon, et al. "Entropy-constrained training of deep neural networks." In IEEE IJCNN, 2019.