DAMP: distribution-aware magnitude pruning for budget-sensitive graph convolutional networks

Hichem Sahbi

DAMP: DISTRIBUTION-AWARE MAGNITUDE PRUNING FOR BUDGET-SENSITIVE GRAPH CONVOLUTIONAL NETWORKS

Hichem Sahbi

Sorbonne University, CNRS, LIP6, F-75005, Paris, France

ABSTRACT

Graph convolutional networks (GCNs) are nowadays becoming mainstream in solving many image processing tasks including skeleton-based recognition. Their general recipe consists in learning convolutional and attention layers that maximize classification performances. With multi-head attention, GCNs are highly accurate but oversized, and their deployment on edge devices requires their pruning. Among existing methods, magnitude pruning (MP) is relatively effective but its design is clearly suboptimal as network topology selection and weight retraining are achieved independently.
In this paper, we devise a novel lightweight GCN design dubbed as Distribution-Aware Magnitude Pruning (DAMP). The latter is variational and proceeds by aligning the weight distribution of the learned networks with an a priori distribution. This allows implementing any targeted pruning rate while maintaining high generalization of the designed lightweight GCNs particularly at the highest (most interesting) pruning regimes. Extensive experiments conducted on the challenging task of skeleton-based recognition show a substantial gain of our DAMP compared to MP as well as related methods.

***Index Terms*—** Graph convolutional networks, lightweight design, magnitude pruning, skeleton-based recognition

1. INTRODUCTION

With the resurgence of deep neural networks [18], many image processing and pattern recognition tasks [5, 11] have been successfully revisited during the last decade [19–23, 25]. These tasks have been approached with increasingly accurate but *oversized* networks, and this makes their deployment on cheap devices, endowed with limited hardware resources, highly challenging. Among existing models, graph convolutional networks (GCNs) are known to be effective particularly on non-euclidean domains including point-clouds and skeletons [8, 24, 66]. Two categories of GCNs are known in the literature; spatial and spectral. Spectral methods [26–33] first project graph signals from the input to the Fourier domain in order to achieve convolution [40], and then back-project the convolved signals in the input domain. Spatial methods [34–39] proceed differently by aggregating node signals using message passing and attention mechanisms prior to apply convolutions on the resulting node aggregates [41]. Spatial GCNs are deemed more effective compared to spectral ones, but their main downside resides in the high computational complexity especially when using multi-head attention.

A major challenge is how to make these attention-based networks lightweight and frugal while maintaining their high accuracy [43–46]. In this regard, many existing works tackle the issue of lightweight network design including tensor decomposition [61], quantization [57], distillation [47–53] and pruning [54–56]. In particular, pruning methods are highly effective. Their principle consists in removing connections whose impact on the classification performances is the least noticeable. Two major categories of pruning techniques exist in the literature; structured [58, 60] and unstructured [56, 57]. The former consists in zeroing-out weights of entire filters or channels whilst the latter seeks to remove weights individually and independently. Whereas structured methods produce computationally more efficient networks, they are less effective compared to unstructured techniques; indeed, the latter provide more flexible (and thereby more accurate) networks which are computationally still efficient.

Magnitude pruning (MP) is one of the mainstream methods that proceeds by removing the smallest weight connections in a given heavy network, prior to retrain the resulting pruned (lightweight) network. While being able to reach any targeted pruning rate, MP is clearly suboptimal as its design *decouples* the training of network topology from weights. Therefore, any removed connection cannot be recovered when retraining the pruned network, and this usually leads to a significant drop in classification performances. In this paper, we investigate a novel alternative for magnitude pruning referred to as DAMP (Distribution-Aware Magnitude Pruning) that allows *coupling end-to-end* the training of network topology and weights by constraining these weights to match a targeted distribution. This allows, *via a band-stop mechanism*, to filter out all the connections up to a given targeted pruning rate. Hence, the advantage of the proposed contribution is twofold; (i) it allows reaching any targeted pruning rate almost exactly by constraining the learned weights to fit a targeted distribution and (ii) this also leads to better generalization, compared to MP, particularly at the highest (most interesting) pruning regimes as reported later in experiments.

2. GRAPH CONVNETS AT A GLANCE

Let $\mathcal{S} = \{\mathcal{G}_i = (\mathcal{V}_i, \mathcal{E}_i)\}_i$ denote a collection of graphs with $\mathcal{V}_i$, $\mathcal{E}_i$ being respectively the nodes and the 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 $(\mathcal{G} \star \mathcal{F})_{\mathcal{V}} = f\big(\mathbf{A}\ \mathbf{U}^\top\ \mathbf{W}\big)$, 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 $(\mathcal{G} \star \mathcal{F})_{\mathcal{V}}$, 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 $(\mathcal{G} \star \mathcal{F})_{\mathcal{V}}$ makes it possible to implement 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 GCN DESIGN

In the remainder of this paper, we subsume a given GCN as a multi-layered neural network g_θ whose weights are 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 variables $\{\mathbf{M}^\ell\}_\ell$ 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.

3.1. Band-stop Weight Parametrization

In order to circumvent the above issues, we consider an alternative *parametrization*, related to magnitude pruning, 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. This parametrization corresponds to the Hadamard product involving a weight tensor and a function applied entry-wise to the same tensor as $\mathbf{W}^\ell = \hat{\mathbf{W}}^\ell \odot \psi(\hat{\mathbf{W}}^\ell)$, here $\hat{\mathbf{W}}^\ell$ 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 parametrization 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. A possible choice, used in practice, that satisfies these four conditions is $\psi_{a,\sigma}(\hat{\mathbf{w}}) = (1 + \sigma \exp(a^2 - \hat{\mathbf{w}}^2))^{-1}$ with σ being a scaling factor and a threshold. As shown in Fig. 1, (σ, a) control the smoothness of $\psi_{a,\sigma}$ around the support $\Omega \subseteq \mathbb{R}$ of the latent weights. A linear increase of σ (w.r.t. training epochs) allows implementing an annealed (soft-to-hard) thresholding function that cuts-off all the connections in smooth and differentiable manner as training of the latent parameters evolves. Put differently, the asymptotic behavior of $\psi_{a,\sigma}$ — that allows selecting the topology of the pruned subnetworks — is obtained as training reaches the latest epochs.

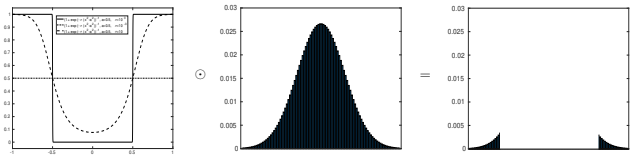

Fig. 1: This figure shows a Band-stop function $\psi_{a,\sigma}$ and its application to a given (gaussian) weight distribution. Depending on the setting of a, only large magnitude weights are kept and correspond to the targeted pruning rate. (Better to zoom the file).

3.2. Distribution-Aware Magnitude Pruning

The aforementioned parametrization — while being effective (see later experiments) — it does not allow implementing any targeted pruning rate as the dynamic of learned latent weights $\{\hat{\mathbf{W}}^\ell\}_\ell$ is not known a priori. Hence, pruning rates could only be observed a posteriori or implemented after training using a two stage process (e.g., magnitude pruning + retraining). In order to implement any a priori targeted pruning rate as a part of a single training process, we constrain the

distribution of latent weights to fit an arbitrary probability distribution, so one may fix a in $\psi_{a,\sigma}$ and thereby achieve the targeted pruning rate. Let $\hat{W} \in \Omega$ denote a random variable standing for the latent weights in the pruned network g_θ; $\hat{W}$ is assumed drawn from a given distribution P (uniform, gaussian, laplace, etc) *possibly the closest to the distribution of unpruned network* (see Fig. 2). Fixing appropriately P *not only* allows implementing any targeted pruning rate, but has also a regularization effect which controls the dynamic of the learned weights and thereby the generalization properties of the pruned network (by making its weight distribution close to the unpruned network) as shown later in table 3.

Fitting a targeted distribution. Considering Q as the observed distribution of the latent weights $\{\hat{\mathbf{W}}^\ell\}_\ell$, and P the targeted one, our goal is to reduce the discrepancy between P and Q using a Kullback-Leibler Divergence (KLD) loss

$$D_{KL}(P||Q) = \int_\Omega P(\hat{W})(\log P(\hat{W}) - \log Q(\hat{W}))\, d\hat{W}. \quad (1)$$

Note that the analytic form of the above equation is known on the widely used probability density functions (PDFs), whilst for general (arbitrary) probability distributions, the exact form is not always known and requires sampling. Hence, we consider instead a discrete variant of this loss w.r.t. P and Q; examples of targeted distributions P are given in Fig. 2 while the observed (and also differentiable) one Q is based on a relaxed variant of histogram estimation. Let $\{q_1, \ldots, q_K\}$ denote a K-bin quantization of Ω (in practice $K = 100$), the k-th entry of Q is defined as

$$Q(\hat{W} = q_k) \propto \sum_{\ell=1}^{L-1} \sum_{i=1}^{n_\ell} \sum_{j=1}^{n_{\ell+1}} \exp\Big\{ -(\hat{\mathbf{W}}^\ell_{i,j} - q_k)^2 / \beta_k^2 \Big\}, \quad (2)$$

here β_k is a scaling factor that controls the smoothness of the exponential function; larger values of β_k result into over-smoothed histogram estimation while a sufficiently (not very) small β_k leads to a surrogate histogram estimation close to the actual discrete distribution of Q. In practice, β_k is set to $(q_{k+1} - q_k)/2$; with this setting, one may replace $\propto$ (in Eq. 2) with an equality as the partition function of Q — i.e., $\sum_{k=1}^{K} Q(\hat{W} = q_k)$ — reaches almost one in practice.

Budget-aware pruning. Let $F_{\hat{W}}(a) = P(\hat{W} \leq a)$ be the cumulative distribution function (CDF) of $P(\hat{W})$. For any given pruning rate r, one may find the threshold a of the parametrization $\psi_{a,\sigma}$ as $a = F^{-1}_{\hat{W}}(r)$. This function, known as the quantile, defines the pruning threshold a on the targeted distribution P (and equivalently on the observed one Q thanks to the KLD loss) which guarantees that only a fraction $(1 - r)$ of the total weights are kept (i.e, nonzero) when applying the band-stop reparametrization in section 3.1. Note that the quantile at any given pruning rate r, can either be empirically evaluated on discrete random variables or can be analytically derived on the widely used PDFs (see table. 1).

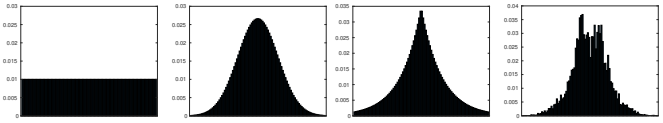

Fig. 2: The first 3 figures correspond to targeted (uniform, gaussian and laplace) distributions. The 4th figure shows the actual weight distribution of the heavy/unpruned GCN which resembles to gaussian/laplacian. This may explain the best performances when these targets are used including at the highest pruning regimes (see table 3).

Considering the above budget implementation, pruning is achieved using a global loss as a combination of a cross-entropy term $\mathcal{L}_e$, and the KLD loss D_{KL} (which controls weight distribution and hence *implicitly* guarantees the targeted pruning rate/budget depending on the setting of the quantile a in $\psi_{a,\sigma}$) resulting into

$$\min_{\{\hat{\mathbf{W}}^\ell\}_\ell} \mathcal{L}_e\big(\{\hat{\mathbf{W}}^\ell \odot \psi(\hat{\mathbf{W}}^\ell)\}_\ell\big) \;+\; \lambda\, D_{KL}(P||Q), \quad (3)$$

here λ is sufficiently large (overestimated to $\lambda = 10$ in practice), so Eq. 3 focuses on implementing the budget and also constraining the pruning rate to reach r. As training evolves, D_{KL} 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.

Distributions	PDF $P(\hat{W})$	Quantile $a = F^{-1}_{\hat{W}}(r)$
Uniform	$\frac{1}{T}$	$a = \frac{r}{T}$
Gaussian	$\frac{1}{\sigma\sqrt{2\pi}} \exp\big\{ -\frac{1}{2}\big(\frac{\hat{W}-\mu}{\sigma}\big)^2 \big\}$	$a = \mu + \sigma\sqrt{2}\mathbf{erf}^{-1}(2r-1)$
Laplace	$\frac{1}{2b} \exp\big\{ -\frac{\lvert\hat{W}-b\rvert}{b} \big\}$	$a = \begin{cases} \mu + b\,\log(2r) & \text{if } r \leq \frac{1}{2} \\ \mu - b\,\log(2-2r) & \text{otherwise} \end{cases}$

Table 1: Different standard PDFs and the underlying quantile functions.

Note that the impact of $D_{KL}(P||Q)$ in Eq. 3 has some similarities and differences w.r.t. the usual regularizers particularly ℓ_0, ℓ_1 and ℓ_2. Whilst these three regularizers favor respectively uniform, laplace and gaussian distributions in Q, there is no guarantee that these regularizers allow implementing any targeted pruning rate and require adding *explicit* (and difficult to solve) budget criteria or *overtrying* many λ values in Eq. 3. In our method, in contrast, as Q is constrained in $D_{KL}(P||Q)$, the Band-pass mechanism in section 3.1 makes reaching any targeted pruning rate easily feasible.

4. EXPERIMENTS

We benchmark our GCNs on the task of action recognition using the First-Person Hand Action (FPHA) dataset [2] which includes 1175 skeletons belonging to 45 action categories. Each sequence of skeletons (video) is initially described with a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ with each node $v_j \in \mathcal{V}$ corresponding to the j-th hand-joint trajectory (denoted as $\{\hat{p}^t_j\}_t$) and an edge

$(v_j, v_i) \in \mathcal{E}$ exists iff the j-th and the i-th trajectories are spatially connected. Each trajectory in $\mathcal{G}$ is processed using *temporal chunking* [59]: first, the total duration of a sequence is split into M evenly-sized temporal chunks ($M = 32$ in practice), then the trajectory coordinates $\{\hat{p}_j^t\}_t$ are assigned to the M chunks (depending on their time stamps) prior to concatenate the averages of these chunks. This produces the raw description (signal) of v_j.

Implementation details and baseline GCN. We trained the GCNs end-to-end using the Adam optimizer [1] for 2,700 epochs with a batch size equal to 600, a momentum of 0.9 and a global learning rate (denoted as $\nu(t)$) inversely proportional to the speed of change of our global 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 $\nu(t) \leftarrow \nu(t-1)/0.99$). We use in our experiments a GeForce GTX 1070 GPU (with 8 GB memory) and we evaluate the performances using the protocol proposed in [2] with 600 action sequences for training and 575 for testing, and we report the average accuracy over all the classes of actions. The architecture of our baseline GCN (taken from [59]; see also section 2) includes stacked 8-head attentions applied to skeleton graphs whose nodes are encoded with 16-channels, followed by convolutions of 32 filters, and a dense fully connected layer as well as a final classification layer. In total, this initial network is relatively heavy (for a GCN). Nevertheless, this GCN is accurate compared to the related work on the challenging FPHA benchmark as shown in Table. 2. Considering this GCN baseline, our goal is to make it lightweight while maintaining its high accuracy.

Lightweight GCNs (Comparison & Ablation). Table 3 shows the performances of our baseline and lightweight GCNs. From these results, we observe the positive impact for different PDFs and for increasing pruning rates r; for mid r values (i.e., 55%), DAMP (gaussian) overtakes all the other settings while for very high pruning regimes (i.e., $\geq$ 98%), DAMP (laplace) is the most performant. Note that lightweight GCNs with mid r overtake the baseline; indeed, mid r values produce subnetworks with already enough (a large number of) connections and having some of them removed from the baseline GCNs produces a well known regularization effect [42]. We also note that the targeted and the observed pruning rates are very similar; the quantile functions of the gaussian and laplace PDFs allow implementing *fine-grained* targeted pruning rates particularly when r is large. In contrast, the quantile functions of the gaussian and laplace PDFs are coarse around mid r values (i.e., 55%). Extra comparison against MP coupled with other regularizers (in Eq. 3 instead of KLD, namely ℓ_0 [62], ℓ_1 [63], entropy [64] and ℓ_2-based cost [65]) shows the substantial gain of DAMP at the highest pruning rate (namely 98%). Note that when alternative regularizers are used, multiple trials of the underlying hyperparameter λ (in Eq. 3) are considered prior to reach the targeted pruning rate, and this makes the whole training and pruning process overwhelming compared to DAMP.

Method	Color	Depth	Pose	Accuracy (%)
Two stream-color [3]	✓	✗	✗	61.56
Two stream-flow [3]	✓	✗	✗	69.91
Two stream-all [3]	✓	✗	✗	75.30
HOG2-depth [4]	✗	✓	✗	59.83
HOG2-depth+pose [4]	✗	✓	✓	66.78
HON4D [6]	✗	✓	✗	70.61
Novel View [7]	✗	✓	✗	69.21
1-layer LSTM [8]	✗	✗	✓	78.73
2-layer LSTM [8]	✗	✗	✓	80.14
Moving Pose [9]	✗	✗	✓	56.34
Lie Group [10]	✗	✗	✓	82.69
HBRNN [12]	✗	✗	✓	77.40
Gram Matrix [13]	✗	✗	✓	85.39
TF [14]	✗	✗	✓	80.69
JOULE-color [15]	✓	✗	✗	66.78
JOULE-depth [15]	✗	✓	✗	60.17
JOULE-pose [15]	✗	✗	✓	74.60
JOULE-all [15]	✓	✓	✓	78.78
Huang et al. [16]	✗	✗	✓	84.35
Huang et al. [17]	✗	✗	✓	77.57
Our GCN baseline	✗	✗	✓	**86.43**

Table 2: Comparison of our baseline GCN against related work on FPHA.

Targeted PR	Observed PR	Target PDFs.	Accuracy (%)	Observation
none	0.00	✗	**86.43**	Baseline GCN
55%	55.00	✗	87.82	MP
	55.10	✓	87.82	DAMP (Uniform)
	55.31	✓	**88.52**	DAMP (Gaussian)
	57.83	✓	87.65	DAMP (Laplace)
80%	80.00	✗	86.78	MP
	77.74	✓	85.91	DAMP (Uniform)
	80.71	✓	**87.47**	DAMP (Gaussian)
	80.11	✓	86.95	DAMP (Laplace)
98%	98.00	✗	60.34	MP
	97.98	✓	70.26	DAMP (Uniform)
	97.97	✓	70.60	DAMP (Gaussian)
	97.90	✓	**70.80**	DAMP (Laplace)
Comparative (reg-based) pruning on the highest (most interesting) pruning regime				
98%	98.00	✗	64.69	MP (+ℓ_0 reg)
	98.00	✗	70.78	MP (+ℓ_1 reg)
	98.00	✗	67.47	MP (+Entropy reg)
	98.00	✗	69.91	MP (+Cost-Aware reg)

Table 3: Detailed performances and ablation, for different targeted and observed pruning rates, and for different targeted probability distributions. Here "PR" stands for pruning rate and "reg" for regularization. For comparative methods, exact PRs are implemented after an overwhelming tuning of λ and retraining with the underlying regularizers (instead of D_{KL}) in Eq. 3; hence these methods are shown for the most interesting PR of 98%, and DAMP obtains the highest accuracy without overwhelming λ tuning.

5. CONCLUSION

We introduce in this paper a novel lightweight GCN design based on Distribution-Aware Magnitude Pruning (DAMP). The strength of DAMP resides in its ability to constrain the probability distribution of the learned GCNs to match an a priori distribution and this allows implementing any given targeted pruning rate while also enhancing the generalization performances of the resulting GCNs. Extensive experiments conducted on the challenging task of skeleton-based recognition shows a significant gain of DAMP against magnitude pruning as well as other regularization-based methods.

6. REFERENCES

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

[2] G. Garcia-Hernando et al. First- Person Hand Action Benchmark with RGB-D Videos and 3D Hand Pose Annotations. In CVPR, 2018

[3] C. Feichtenhofer, A. P., and A. Zisserman. Conv 2-Stream Network Fusion for Video Act Rec. CVPR, pages 1933-1941, 2016. 8

[4] E.Ohn-Barand, M.M.Trivedi. Hand Gesture Recognition in Real Time for Automotive Interfaces: A Multimodal Vision- Based Approach and Evaluations. IEEE TITS, 15(6):2368–2377, 2014.

[5] L. Wang and H. Sahbi. Bags-of-daglets for action recognition. In IEEE ICIP 2014.

[6] O. Oreifej and Z. Liu. HON4D: Hist of Orient 4D Norm for Act Recognition from Depth Sequences. In CVPR, pages 716-723, June 2013.

[7] H. Rahmani and A. Mian. 3D Action Recognition from Novel Viewpoints. In CVPR, pages 1506–1515, June 2016.

[8] W. Zhu et al. Co-occurrence feature learning for skeleton based action recognition using regularized deep LSTM networks In AAAI, 2016.

[9] M. Zanfir et al. The Moving Pose: An Efficient 3D Kinematics Descriptor for Low-Latency Action Recognition and Detection. In ICCV, 2013.

[10] R. Vemulapalli, F. Arrate, and R. Chellappa. Hum act rec by rep 3D skeletons as points in a Lie group. In IEEE CVPR, pages 588–595, 2014

[11] L. Wang and H. Sahbi. Nonlinear cross-view sample enrichment for action recognition. In Computer Vision-ECCV 2014 Workshops, 2014.

[12] Y. Du, W. Wang, and L. Wang. Hier rec neural net for skeleton based action recognition. In IEEE CVPR, pages 1110–1118, 2015.

[13] X. Zhang et al. Efficient Temporal Sequence Comparison and Classification Using Gram Matrix Embeddings on a Riemannian Manifold. In CVPR, pages 4498–4507, 2016

[14] G. Hernando et al. Transition Forests: Learning Discriminative Temporal Transitions for Action Recognition. In CVPR, pages 407–415, 2017.

[15] J. Hu, W. Zheng, J. Lai, and J. Zhang. Jointly Learning Heterogeneous Features for RGB-D Activity Recognition. In CVPR 2015

[16] Z. Huang and L. V. Gool. A Riemannian Network for SPD Matrix Learning. In AAAI, pages 2036–2042, 2017

[17] Z. Huang, J. Wu, and L. V. Gool. Building Deep Networks on Grassmann Manifolds. In AAAI, pages 3279–3286, 2018

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

[19] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *CVPR*, 2016, pp. 770–778.

[20] M. Jiu and H. Sahbi. Deep representation design from deep kernel networks. Pattern Recognition, 88, 447-457, 2019.

[21] G. Huang et al. "Densely connected conv networks," *CVPR*, 2017.

[22] He, Kaiming, et al. "Mask r-cnn." Proceedings of ICCV, 2017.

[23] M. Jiu and H. Sahbi. DHCN: Deep hierarchical context networks for image annotation. In IEEE ICASSP 2021.

[24] Zhang et al. "Deep learning on graphs: A survey." IEEE TKDE (2020).

[25] O. Ronneberger et al. "U-net: Conv net for biom image seg." Int Conf on Medical image computing and computer-assisted intervention, 2015.

[26] J. Bruna et al. Spectral networks and locally connected networks on graphs. arXiv:1312.6203 (2013)

[27] A. Mazari and H. Sahbi. MLGCN: Multi-Laplacian graph convolutional networks for human action recognition. In BMVC, 2019

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

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

[30] R. Levie et al. Cayleynets: Graph conv neural nets with complex rational spectral filters. IEEE TSP 67(1), 97–109 (2018)

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

[32] H. Sahbi. Learning laplacians in chebyshev graph convolutional networks. In Proceedings of the IEEE/CVF ICCV (pp. 2064-2075), 2021.

[33] M. Defferrard, X. Bresson, P. Vandergheynst. Conv Neural Net on graphs with Fast Localized Spectral Filtering. In NIPS, 2016

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

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

[36] H. Sahbi. Kernel-based graph convolutional networks. In ICPR, 2021.

[37] Z. Wu et al. A comprehensive survey on graph neural networks. arXiv:1901.00596 (2019).

[38] H. Sahbi. Lightweight Connectivity In Graph Convolutional Networks For Skeleton-Based Recognition. In IEEE ICIP, 2021.

[39] W. Hamilton et al. Ind rep learning on large graphs. In NIPS, 2017.

[40] Chung, Fan RK, and Fan Chung Graham. Spectral graph theory. No. 92. American Mathematical Soc., 1997.

[41] Knyazev et al. "Understanding attention and generalization in graph neural networks." In NIPS, 2019.

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

[43] Gao et al. "Condensenet: An efficient densenet using learned group convolutions," in *CVPR*, 2018.

[44] M. Sandler et al. "Mobilenetv2: Inverted residuals and linear bottlenecks," in *CVPR*, 2018.

[45] A. G. Howard et al. "Mobilenets: Efficient convolutional neural networks for mobile vision applications," *CoRR*, abs/1704.04861, 2017.

[46] M. Tan and Q. V. Le, "Efficientnet: Rethinking model scaling for convolutional neural networks," in *ICML*. 2019, vol. 97, PMLR.

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

[48] H. Sahbi and D. Geman. A Hierarchy of Support Vector Machines for Pattern Detection. Journal of Machine Learning Research, 7(10), 2006.

[49] S. Zagoruyko et al. "Paying more att to att: Improving the performance of convolutional neural networks via attention transfer," *ICLR*, 2017.

[50] A. Romero et al. "Fitnets: Hints for thin deep nets," in *ICLR*, 2015.

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

[52] Y. Zhang al al. "Deep mutual learning," in *CVPR*, 2018.

[53] S. Ahn et al. "Variational inf distillation for know trans," *CVPR*, 2019.

[54] Y. LeCun et al. "Optimal brain damage," in *NIPS*, 1989.

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

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

[57] S. Han, H. Mao, and W. J. Dally, "Deep compression: Comp deep neural net with pruning, trained quant and huff coding," in *ICLR*, 2016.

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

[59] H. Sahbi. "Learning Connectivity with Graph Convolutional Networks." 25th ICPR. IEEE, 2021.

[60] Z. Liu et al. "Learn efficient conv networks through network slimming," *ICCV*. 2017.

[61] Howard, Andrew, et al. "Searching for mobilenetv3." ICCV 2019.

[62] C. Louizos et al. Learning sparse neural networks through l0 regularization . In proc of ICLR,2018

[63] B. Koneru et al. "Sparse art neur net using a novel smoothed LASSO pen."IEEE TCS II: Express Briefs 66.5 (2019): 848-852.

[64] Wiedemann et al. "Entropy-constrained training of deep neural networks." IJCNN,2019.

[65] C. Lemaire et al. "Structured pruning of neural networks with budget-aware regularization." Proceedings of the IEEE/CVF CVPR, 2019.

[66] H. Sahbi, J-Y. Audibert and R. Keriven. Context-dependent kernels for object classification. IEEE PAMI, 33(4), 699-708, 2011.