Compressing Neural Networks using the Variational Information Bottleneck

Bin Dai[1]; Chen Zhu[2]; Baining Guo[3]; David Wipf[3]

[1]Tsinghua University, [2]University of Maryland, [3]Microsoft Research

Neural Networks are Often Over-Parameterized

(*Denil et al., 2013*)

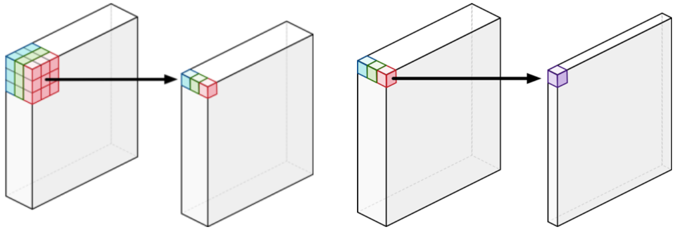
Consequences:

- Unnecessarily large model size
- Increased computational cost
- Extra run-time memory footprint

Network Compression Methods

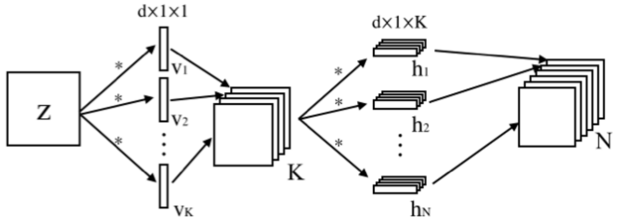
- Design more efficient network structure

  (*Howard et al., 2017; Dong et al., 2017; Iandola et al., 2016*)



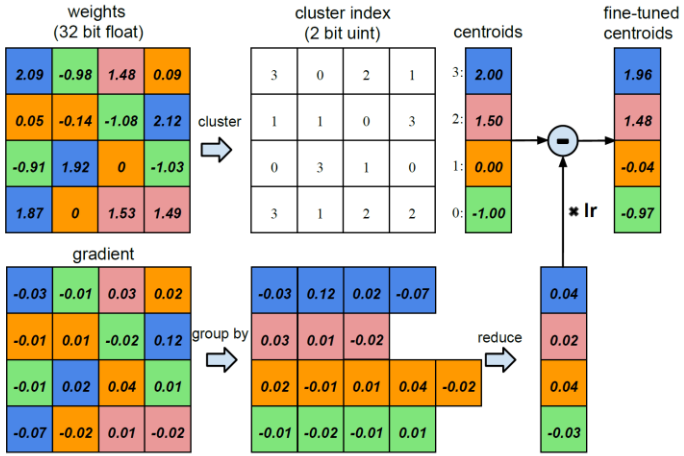
- Tensor/matrix decomposition

  (*Jaderberg et al., 2014; Zhang et al., 2016; Yu et al., 2017*)



- Weight Quantization

  (*Courbariaux et al., 2016; 2015; Han et al., 2015a; Mellempudi et al., 2017; Rastegari et al., 2016*)

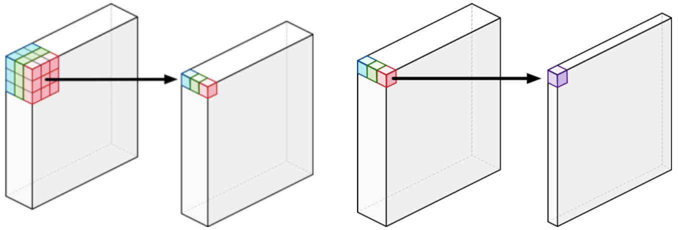


Images from http://machinethink.net/blog/googles-mobile-net-architecture-on-iphone/, Jaderberg et al., 2014 and Han et al. 2016

Network Compression Methods

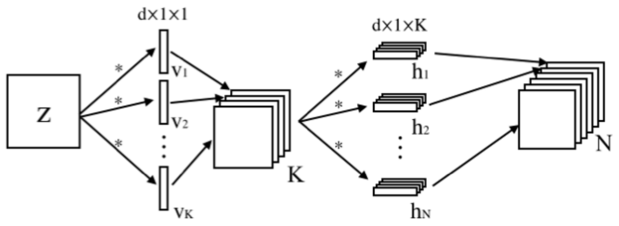
- Design more efficient network structure

  (*Howard et al., 2017; Dong et al., 2017; Iandola et al., 2016*)



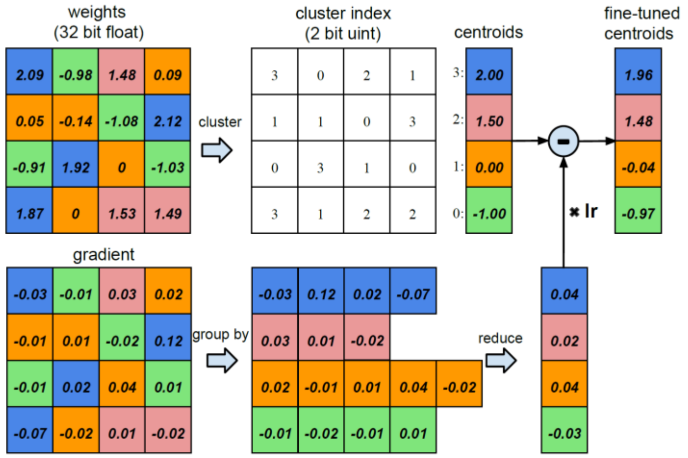
- Tensor/matrix decomposition

  (*Jaderberg et al., 2014; Zhang et al., 2016; Yu et al., 2017*)



- Weight Quantization

  (*Courbariaux et al., 2016; 2015; Han et al., 2015a; Mellempudi et al., 2017; Rastegari et al., 2016*)



- **Prune existing network structure …**

Images from http://machinethink.net/blog/googles-mobile-net-architecture-on-iphone/, Jaderberg et al., 2014 and Han et al. 2016

Pruning Methods

- Prune weight connections

  (*Han et al., 2015b; Guo et al., 2016; LeCun et al., 1990*)

- Prune weight groups / activations
  - Group lasso

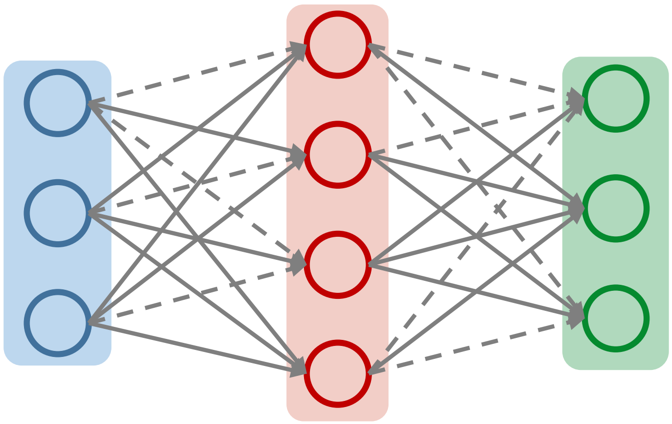
    (*Liu et al., 2017; Pan et al., 2016; Wen et al., 2016*)

  - Bayesian approach

    (*Louizos et al., 2017a; Neklyudov et al., 2017*)

  - Smoothed l_0 approach

    (*Louizos et al., 2017b*)



Prune weight connections

Pruning Methods

- Prune weight connections

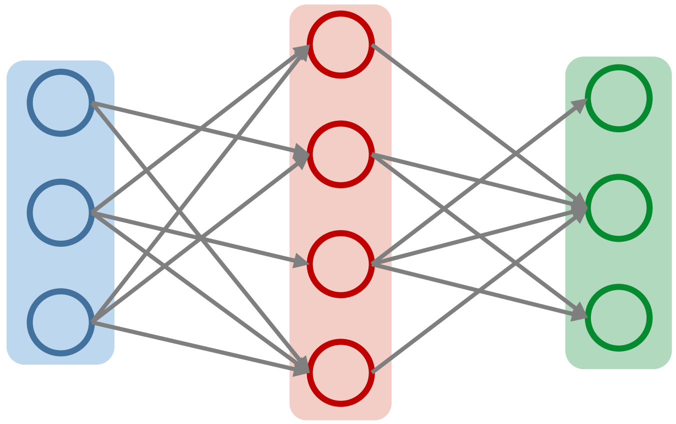
  (*Han et al., 2015b; Guo et al., 2016; LeCun et al., 1990*)

- Prune weight groups / activations
  - Group lasso

    (*Liu et al., 2017; Pan et al., 2016; Wen et al., 2016*)
  - Bayesian approach

    (*Louizos et al., 2017a; Neklyudov et al., 2017*)
  - Smoothed l_0 approach

    (*Louizos et al., 2017b*)

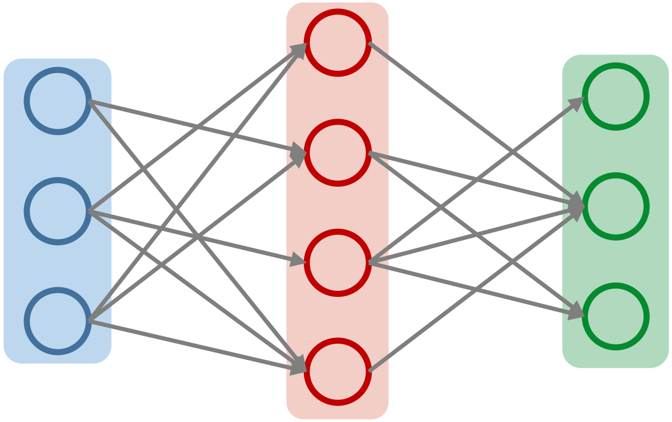


Prune weight connections

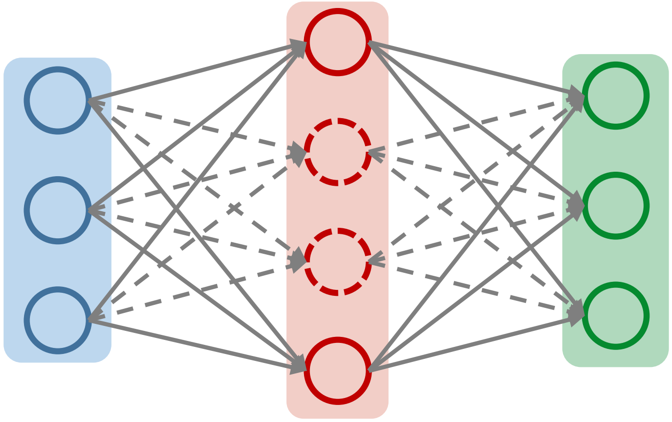
Pruning Methods

- Prune weight connections
  (*Han et al., 2015b; Guo et al., 2016; LeCun et al., 1990*)

- Prune weight groups / activations
  - Group lasso
    (*Liu et al., 2017; Pan et al., 2016; Wen et al., 2016*)
  - Bayesian approach
    (*Louizos et al., 2017a; Neklyudov et al., 2017*)
  - Smoothed l_0 approach
    (*Louizos et al., 2017b*)



Prune weight connections

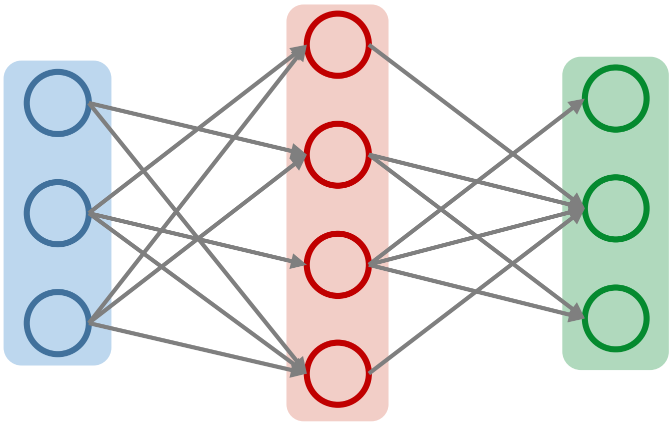


Prune weight groups / activations

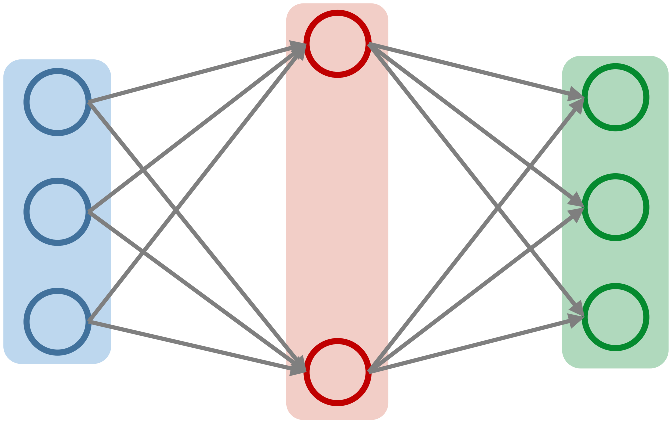
Pruning Methods

- Prune weight connections
  (*Han et al., 2015b; Guo et al., 2016; LeCun et al., 1990*)

- Prune weight groups / activations
  - Group lasso
    (*Liu et al., 2017; Pan et al., 2016; Wen et al., 2016*)
  - Bayesian approach
    (*Louizos et al., 2017a; Neklyudov et al., 2017*)
  - Smoothed l_0 approach
    (*Louizos et al., 2017b*)



Prune weight connections



Prune weight groups / activations

Markov Chain Interpretation of Network

$$y \rightarrow x(h_0) \rightarrow h_1 \rightarrow \cdots \rightarrow h_{i-1} \rightarrow h_i \rightarrow \cdots \rightarrow h_L \rightarrow \hat{y}$$

(*Tishby & Zaslavsky, 2015*)

Defines $p(h_i|h_{i-1})$

Degenerates to a Dirac-delta function if the network is deterministic

We consider non-degenerate distribution in our model

Markov Chain Interpretation of Network

$$y \rightarrow x(h_0) \rightarrow h_1 \rightarrow \cdots \rightarrow h_{i-1} \rightarrow h_i \rightarrow \cdots \rightarrow h_L \rightarrow \hat{y}$$

(*Tishby & Zaslavsky, 2015*)

Approximates $p(y|h_L)$ via some tractable alternative $p(\hat{y}\,|h_L)$

Intuition

$$y \rightarrow x(h_0) \rightarrow h_1 \rightarrow \cdots \rightarrow h_{i-1} \rightarrow h_i \rightarrow \cdots \rightarrow h_L \rightarrow \hat{y}$$

(*Tishby & Zaslavsky, 2015*)

- Maximize the mutual information $I(h_i; y)$ between h_i and y

For high-accuracy prediction

- Minimize the mutual information $I(h_i; h_{i-1})$ between h_i and h_{i-1}

For compression

Information Bottleneck

Layer-wise Energy

- Minimize the mutual information $I(h_i; h_{i-1})$ between h_i and h_{i-1}
- Maximize the mutual information $I(h_i; y)$ between h_i and y

$$\mathcal{L}_i = \gamma_i I(h_i; h_{i-1}) - I(h_i; y)$$

γ_i determines the strength of the bottleneck

Upper Bound

$$\mathcal{L}_i = \gamma_i I(h_i; h_{i-1}) - I(h_i; y)$$

$$\gamma_i \mathbb{E}_{\{x,y\}\sim\mathcal{D},h_{1:i-1}\sim p(h_{1:i-1}|x)}\left[\mathbb{KL}[p(h_i|h_{i-1})||q(h_i)]\right] \geq \gamma_i I(h_i; h_{i-1})$$

$$-\mathbb{E}_{\{x,y\}\sim\mathcal{D},h\sim p(h|x)}[\log q(y|h_L)] \geq -I(h_i; y)$$

$$\tilde{\mathcal{L}}_i = \gamma_i \mathbb{E}_{\{x,y\}\sim\mathcal{D},h_{1:i-1}\sim p(h_{1:i-1}|x)}\left[\mathbb{KL}[p(h_i|h_{i-1})||q(h_i)]\right] - \mathbb{E}_{\{x,y\}\sim\mathcal{D},h\sim p(h|x)}[\log q(y|h_L)]$$

$\mathcal{D}$: data distribution

$h_{1:i}$: $\{h_j\}_{j=1}^{i}$

h: $h_{1:L}$

$q(h_i)$: variational approximation of $p(h_i)$

$q(y|h_L)$: variational approximation of $p(y|h_L)$

Parameterization of $q(y|h_L)$

$$\tilde{\mathcal{L}}_i = \gamma_i \mathbb{E}_{\{x,y\}\sim\mathcal{D},h_{1:i-1}\sim p(h_{1:i-1}|x)}\left[\mathbb{KL}[p(h_i|h_{i-1})||q(h_i)]\right]$$
$$-\mathbb{E}_{\{x,y\}\sim\mathcal{D},h\sim p(h|x)}[\log q(y|h_L)]$$

$q(y|h_L)$: multinomial distribution for classification task

Gaussian distribution for regression task

Parameterization of $q(h_i)$

$$\tilde{\mathcal{L}}_i = \gamma_i \mathbb{E}_{\{x,y\}\sim\mathcal{D}, h_{1:i-1}\sim p(h_{1:i-1}|x)}\left[\mathbb{KL}[p(h_i|h_{i-1})||q(h_i)]\right]$$
$$-\mathbb{E}_{\{x,y\}\sim\mathcal{D}, h\sim p(h|x)}[\log q(y|h_L)]$$

$$q(h_i) = \mathcal{N}(h_i; 0, diag[\xi_i])$$

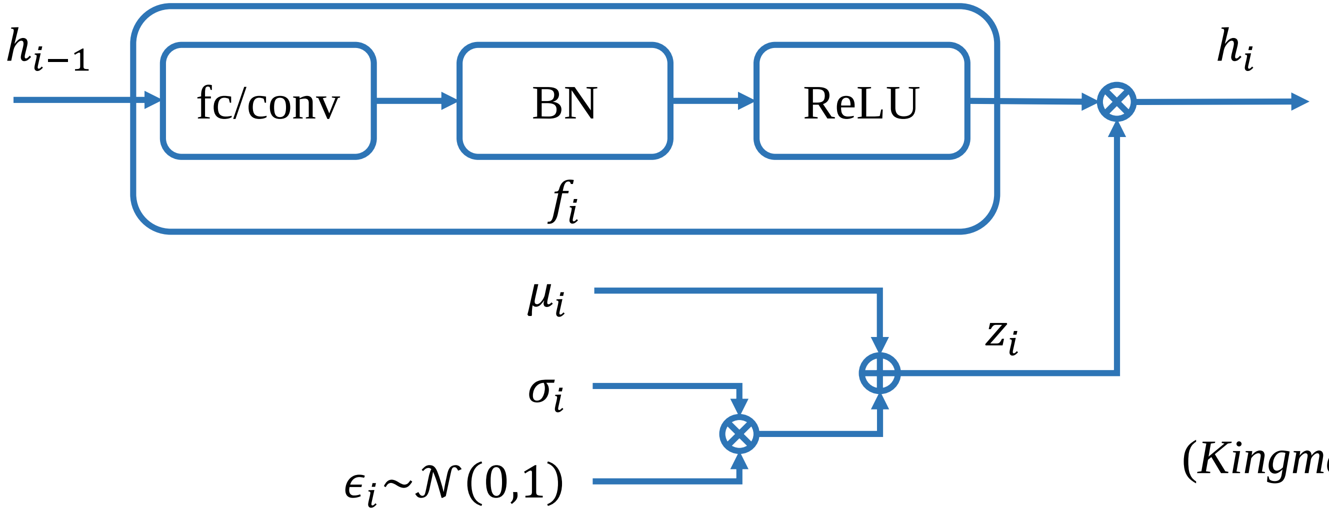
ξ_i: unknown vector of variances learned from data

(*Tipping 2001*)

Parameterization of $p(h_i|h_{i-1})$

$$\tilde{\mathcal{L}}_i = \gamma_i \mathbb{E}_{\{x,y\}\sim\mathcal{D},h_{1:i-1}\sim p(h_{1:i-1}|x)}\left[\mathbb{KL}[p(h_i|h_{i-1})||q(h_i)]\right]$$
$$-\mathbb{E}_{\{x,y\}\sim\mathcal{D},h\sim p(h|x)}[\log q(y|h_L)]$$

$$p(h_i|h_{i-1}) = \mathcal{N}(h_i; f_i(h_{i-1})\odot\mu_i, diag[f_i(h_{i-1})^2\odot{\sigma_i}^2])$$



$z_i = \mu_i + \epsilon_i\odot\sigma_i$

Reparameterization trick

(*Kingma and Welling, 2014; Rezende et al., 2014*)

Final Objective Function

$$\inf_{\xi_i \succ 0} 2\mathbb{E}_{\{x,y\}\sim\mathcal{D},h_{1:i-1}\sim p(h_{1:i-1}|x)}\left[\mathbb{KL}[p(h_i|h_{i-1})||q(h_i)]\right]$$

$$\equiv \sum_{j=1}^{r_i}\left[\log\left(1+\frac{\mu_{i,j}^2}{\sigma_{i,j}^2}\right)+\psi_{i,j}\right]$$

$$\psi_{i,j} \triangleq \log \mathbb{E}_{h_{i-1}\sim p(h_{i-1})}\left[f_{i,j}(h_{i-1})^2\right] - \mathbb{E}_{h_{i-1}\sim p(h_{i-1})}\left[\log f_{i,j}(h_{i-1})^2\right]$$

Final Objective Function

$$\tilde{\mathcal{L}} = \sum_{i=1}^{L} \gamma_i \sum_{j=1}^{r_i} \log\left(1 + \frac{\mu_{i,j}^2}{\sigma_{i,j}^2}\right) - L\mathbb{E}_{\{x,y\}\sim\mathcal{D},h\sim p(h|x)}[\log q(y|h_L)] \quad (*)$$

Regularization term — Data-fit term

r_i: Number of neurons in the i-th layer.

Define $\alpha_{i,j} = \frac{\mu_{i,j}^2}{\sigma_{i,j}^2}$.

Final Objective Function

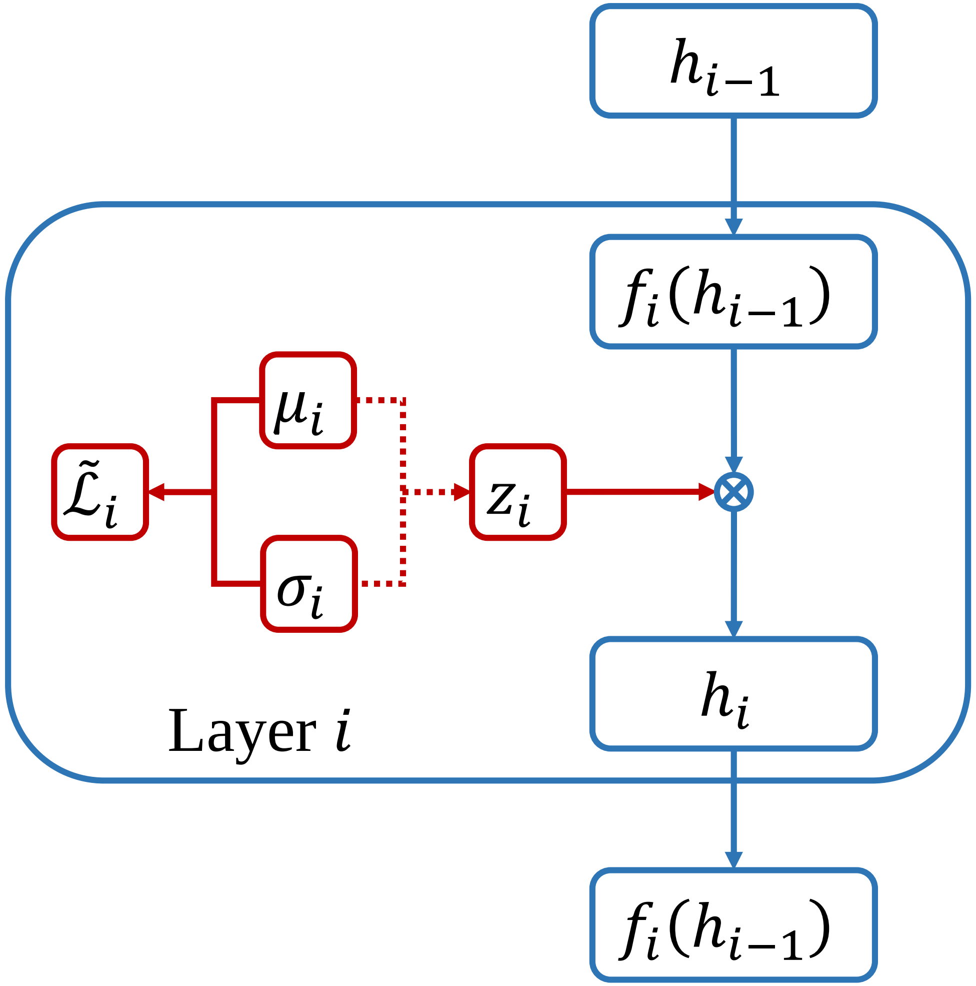
$$\tilde{\mathcal{L}} = \underbrace{\sum_{i=1}^{L} \gamma_i \sum_{j=1}^{r_i} \log\left(1 + \frac{\mu_{i,j}^2}{\sigma_{i,j}^2}\right)}_{\text{Regularization term}} - \underbrace{L\mathbb{E}_{\{x,y\}\sim\mathcal{D},h\sim p(h|x)}[\log q(y|h_L)]}_{\text{Data-fit term}} \quad (*)$$

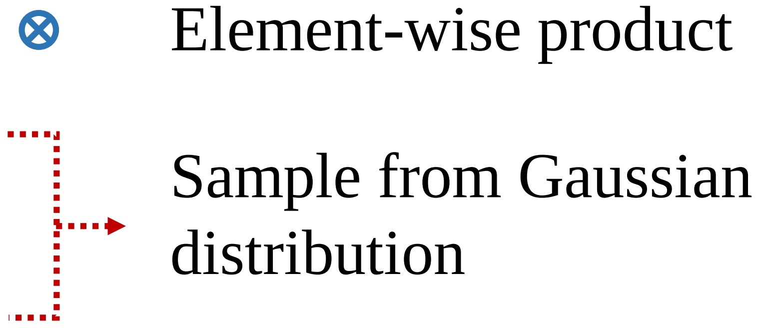
r_i: Number of neurons in the i-th layer.

Define $\alpha_{i,j} = \frac{\mu_{i,j}^2}{\sigma_{i,j}^2}$.

$$\tilde{\mathcal{L}} = \sum_{i=1}^{L} \gamma_i \sum_{j=1}^{r_i} \log(1 + \alpha_{i,j}) - L\mathbb{E}_{\{x,y\}\sim\mathcal{D},h\sim p(h|x)}[\log q(y|h_L)] \quad (*)$$

Network Structure





Variational Information Bottleneck Network (VIBNet)

Reduce Redundancy via Intrinsic Sparsity

$$\tilde{\mathcal{L}} = \sum_{i=1}^{L} \gamma_i \sum_{j=1}^{r_i} \log(1 + \alpha_{i,j}) - L\mathbb{E}_{\{x,y\}\sim\mathcal{D},h\sim p(h|x)}[\log q(y|h_L)] \quad (*)$$

Regularization term — Data-fit term

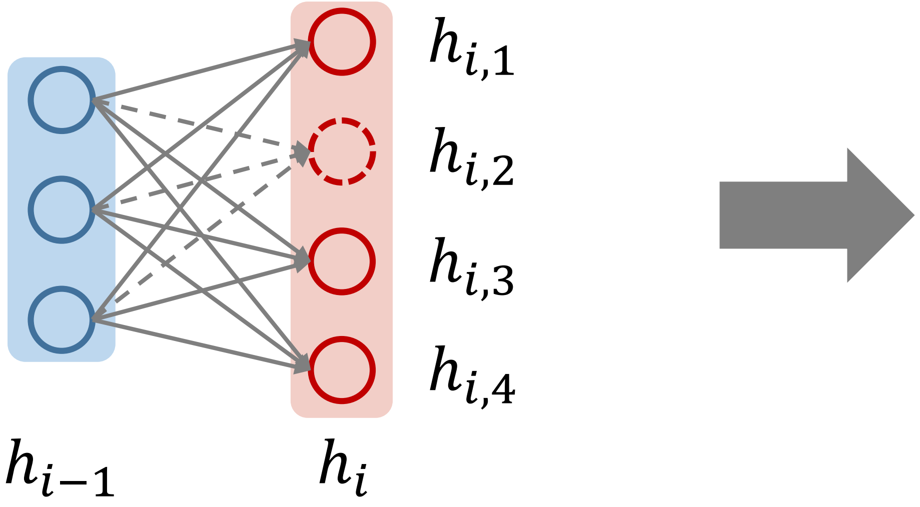
$\log(1 + \alpha_{i,j})$: concave non-decreasing function on $[0, \infty)$

- ✓ Push some $\alpha_{i,j}$ to exactly 0 and leaving others mostly unchanged
- ✗ Push all $\alpha_{i,j}$ towards smaller values

(*Chen et al., 2017*)

Relationship Between $\alpha_{i,j}$ and $I(h_{i,j}, h_{i-1})$

Proposition 1. At any minimum of $(*)$



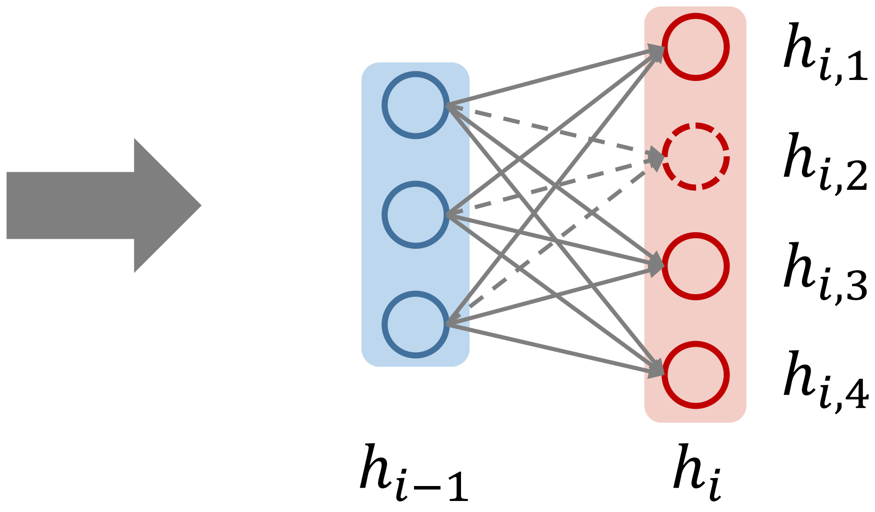
$$I(h_{i,2}, h_{i-1}) = 0$$

$$\alpha_{i,2} = 0$$

Relationship Between $\alpha_{i,j}$ and $I\left(h_{i,j}, h_{i-1}\right)$

Proposition 1. At any minimum of $(*)$…

$\alpha_{i,2} = 0$

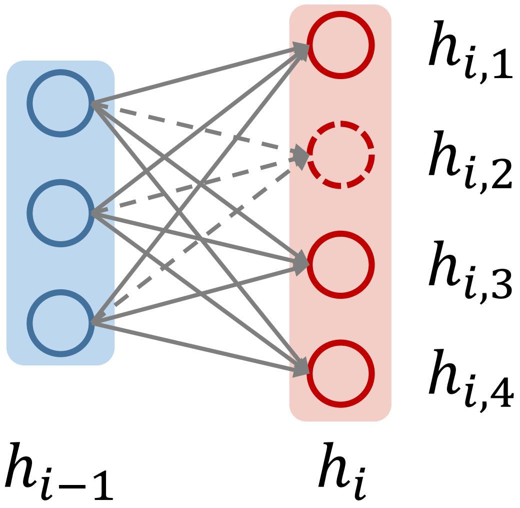


$$I\left(h_{i,2}, h_{i-1}\right) < \psi_{i,2}$$

Relationship Between $\alpha_{i,j}$ and $I\left(h_{i,j}, h_{i-1}\right)$

Proposition 1. At any minimum of $(*)$

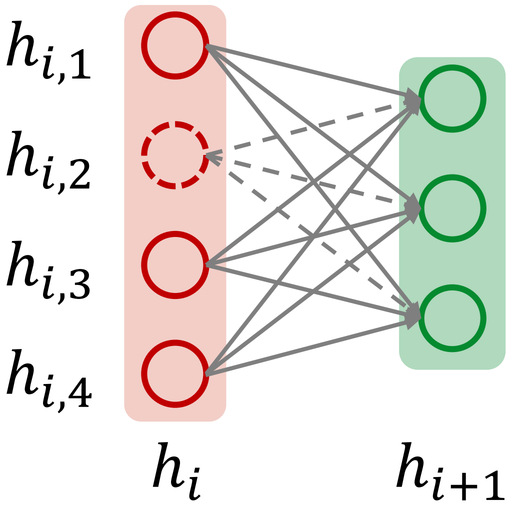
$\alpha_{i,2} = 0$



$I\left(h_{i,2}, h_{i-1}\right) = 0$ With added assumptions.

Relationship Between $\alpha_{i,j}$ and $W_{i+1,\cdot j}$

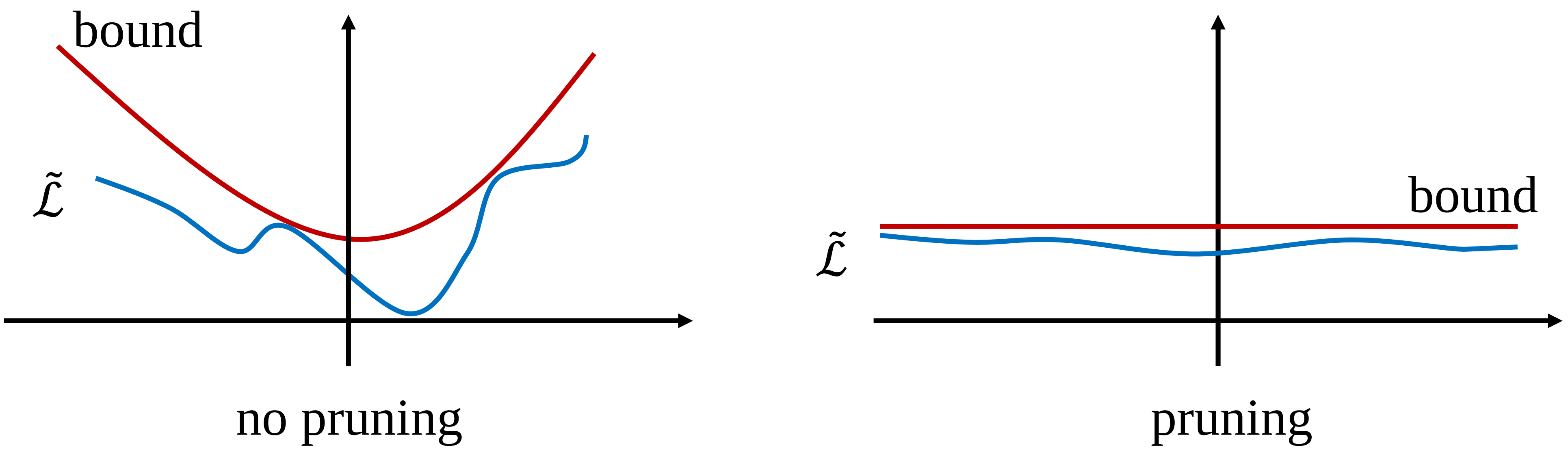
Proposition 2. At any minimum of $(*)$



$W_{i+1,\cdot 2} = 0$ $\Rightarrow$ $\alpha_{i,2} = 0$

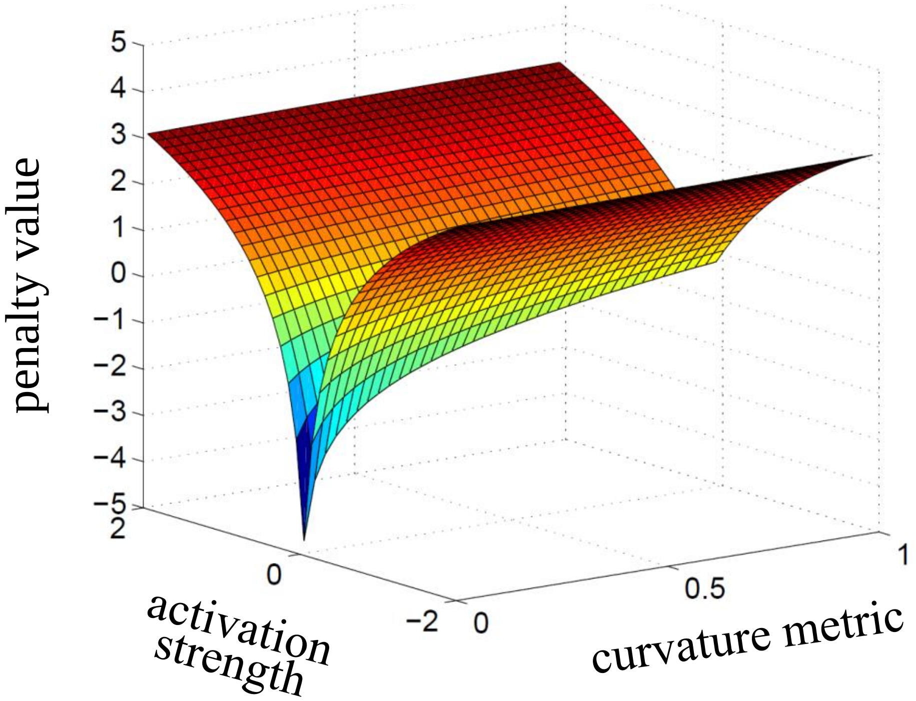
Analysis of Tractable Upper Bound

At any minimum, redundant activations will provably be pruned



Analysis of Tractable Upper Bound

Effective sparsity penalty shape is adaptive relative to the curvature of data fitting terms:



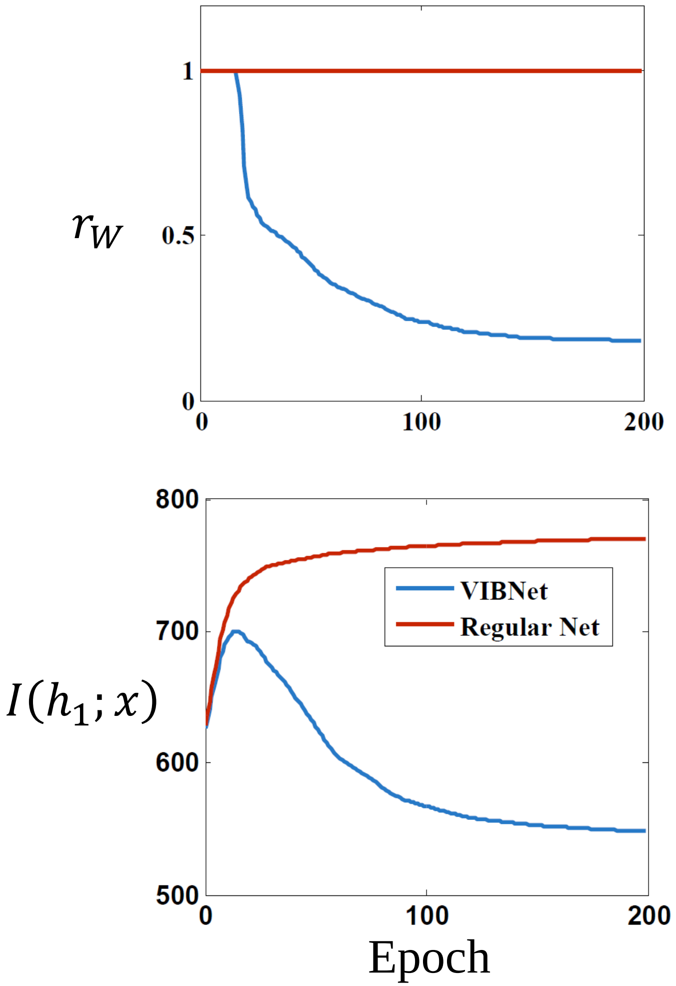
Training and Testing

- Training
  - Bottleneck parameters set with simple heuristic to roughly match accuracy of existing methods
- Pruning
  - Small values of $\alpha_{i,j}$ are set to zero (SGD will not push to exactly zero)
- Testing
  - Use the mean value of $p(h_i|h_{i-1})$ rather than sampling
  - Use multiple samples can further improve the accuracy (at the cost of computation time)
  - If desired, can finetune the pruned model to further boost the accuracy

(*Louizos et al., 2017a*)

LeNet300-100 on MNIST

Method	r_W(%)	r_N(%)	Error(%)	Pruned Model
VD (*Molchanov et al., 2017*)	25.28	58.95	1.8	512-114-72
BC-GNJ (*Louizos et al., 2017a*)	10.76	32.85	1.8	278-98-13
BC-GHS (*Louizos et al., 2017a*)	10.55	34.71	1.8	311-86-14
L0 (*Louizos et al., 2017b*)	26.02	45.02	**1.4**	219-214-100
L0-sep (*Louizos et al., 2017b*)	10.01	32.69	1.8	266-88-33
DN (*Pan et al., 2016*)	23.05	57.94	1.8	542-83-61
VIBNet	**3.59**	**16.98**	1.6	**97-71-33**



$$r_W = \frac{\#\text{ params left}}{\#\text{ params total}} \qquad r_N = \frac{\text{Memory footprint of the pruned model}}{\text{Memory footprint of the original model}}$$

VGG16 on CIFAR10

Method	r_W(%)	r_N(%)	Error(%)	FLOP(Mil)
BC-GNJ (*Louizos et al., 2017a*)	6.57	81.68	8.6	141.5
BC-GHS (*Louizos et al., 2017a*)	5.40	74.82	9.0	121.9
VIBNet	**5.30**	**49.57**	8.8 (**8.5**)	**70.63**
PF (*Li et al., 2017b*)	35.99	83.97	6.6	206.3
SBP (*Neklyudov et al., 2017*)	7.01	80.72	7.5	136.0
SBPa (*Neklyudov et al., 2017*)	5.78	66.46	9.0	99.20
VIBNet	**5.45**	**57.86**	6.5 (**6.1**)	**86.82**
NS-Single (*Liu et al., 2017*)	11.50	-	6.2	195.5
NS-Best (*Liu et al., 2017*)	8.60	-	5.9	147.0
VIBNet	**5.79**	**59.60**	6.2 (**5.8**)	**116.0**

Previous works used three different modifications of VGG16 on CIFAR10.

VGG16 on CIFAR100

Method	r_W(%)	r_N(%)	Error(%)	FLOP(Mil)
RNP (*Lin et al., 2017*)	-	-	38.0	160
VIBNet	**22.75**	**59.80**	37.6 (**37.4**)	**133.6**
NS-Single (*Liu et al., 2017*)	24.90	-	26.5	250.5
NS-Best (*Liu et al., 2017*)	20.80	-	26.0	214.8
VIBNet	**15.08**	**73.80**	25.9 (**25.7**)	**203.1**

Conclusion

- We proposed a network compression model inspired by the information bottleneck principle
- Theoretical analysis shows that the objective tends to accumulate useful information in a sparse set of neurons
- The adaptive sparsity penalty that emerges from our model has advantages over traditional fixed sparsity penalties
- Empirical results show that our model can produce better performance than previous compression models

Q & A

https://github.com/zhuchen03/VIBNet

Poster #128