Institución Universitaria
Reacreditada en Alta Calidad

Compresión de NNs - Pruning

INTEGRACIÓN DE ML EN EMBEBIDOS Y EDGE COMPUTING

Somos Innovación Tecnológica con Sentido Humano

Alcaldía de Medellín

Contenido

1. Técnicas para compresión de redes neuronales
2. Pruning

Técnicas

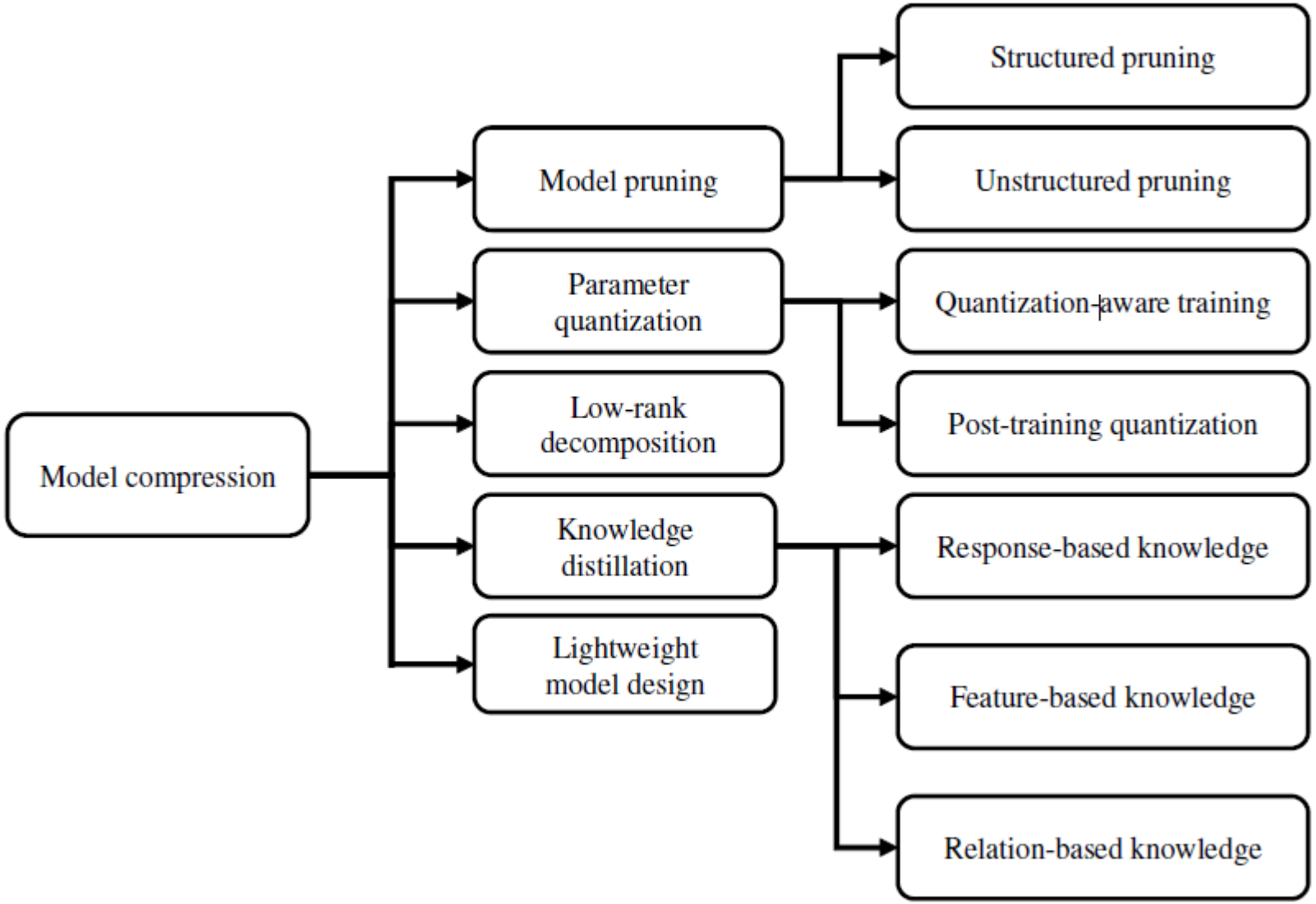
Podado del modelo (Model pruning): Diseño de criterios de evaluación de parámetros (pesos) para medir la importancia del parámetro. Remueve parámetros poco relevantes. Se aplica a capas Conv y FC.

Quantización: Convierte los calculos de punto flotante a calculos con enteros baja tasa de bits. Se aplica a capas Conv y FC.

Descomposiciones de rango bajo: Descompone los tensores originales en varios tensores de rango bajo. Se aplica a capas Conv.

Knowledge distillation: Usa una red grande de alta complejidad como maestra para enseñar a redes estudiantes de baja complejidad. . Se aplica a capas Conv y FC.

Técnicas



Zhuo Li et al, Model Compression for Deep Neural Networks: A Survey, Computers 2023, https://doi.org/10.3390/computers12030060

Contenido

1. Técnicas para compresión de redes neuronales
2. Pruning

Pruning – Motivación

Age	Number of Connections	Stage
at birth	50 Trillion	newly formed
1 year old	1000 Trillion	peak
10 year old	500 Trillion	pruned and stabilized

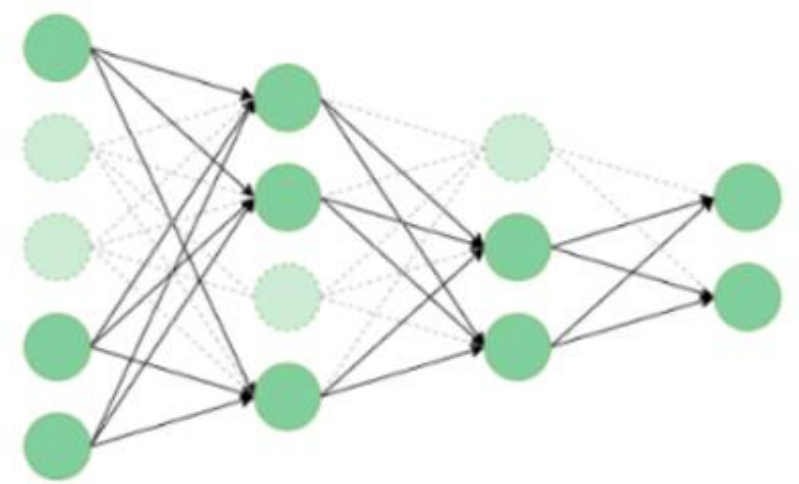
Table 1: The synapses pruning mechanism in human brain development

El mecanismo de podado remueve conexiones del cerebro que son redundantes.

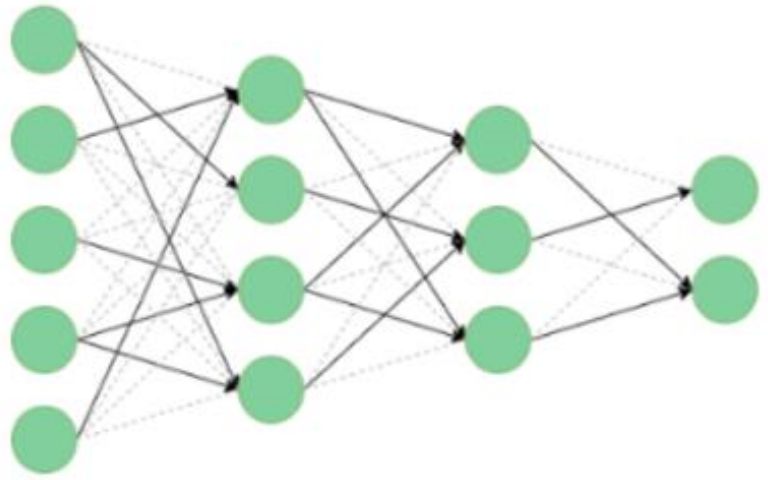
https://web.stanford.edu/class/ee380/Abstracts/160106-slides.pdf

Pruning – Tipos

Structured Pruning: involucre la remoción selectiva de una parte más grande de la red, como una capa o un canal.



Unstructured Pruning: encuentra y remueve las conexiones menos destacadas del modelo donde quiera que estén. No considera cualquier relación entre los pesos podados.

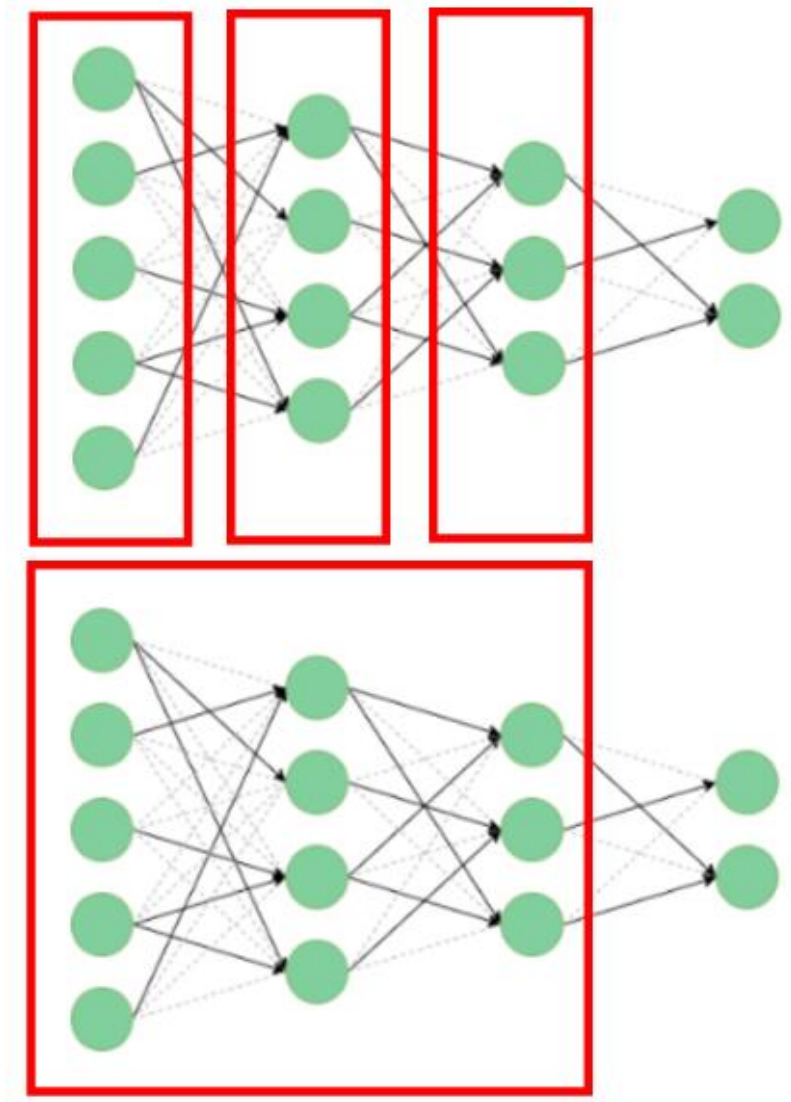


https://web.stanford.edu/class/ee380/Abstracts/160106-slides.pdf

Pruning – Local o Global

Local Pruning: Consiste en remover un porcentaje fijo de unidades/conexiones de cada capa por comparación en la capa.

Global Pruning: toma todos los parámetros juntos a través de las capas y selecciona una porción global de ellos a ser podados.



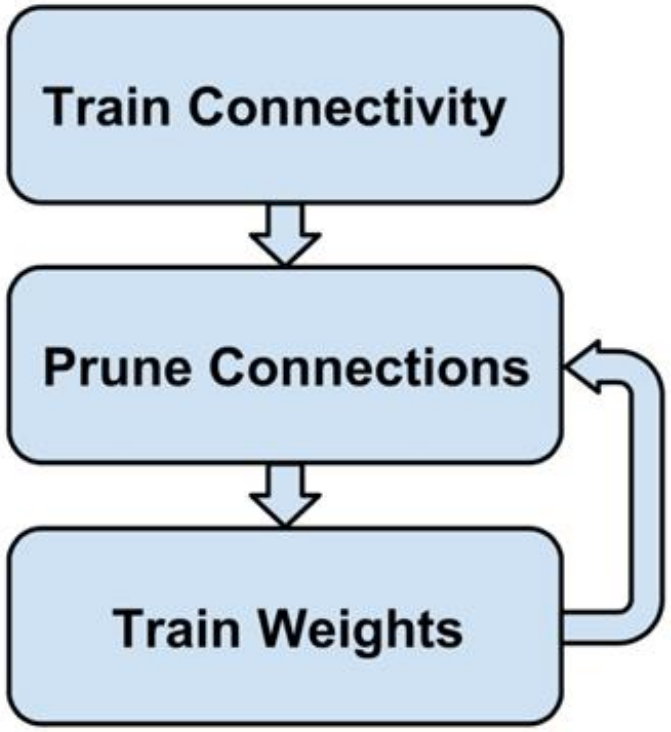
https://web.stanford.edu/class/ee380/Abstracts/160106-slides.pdf

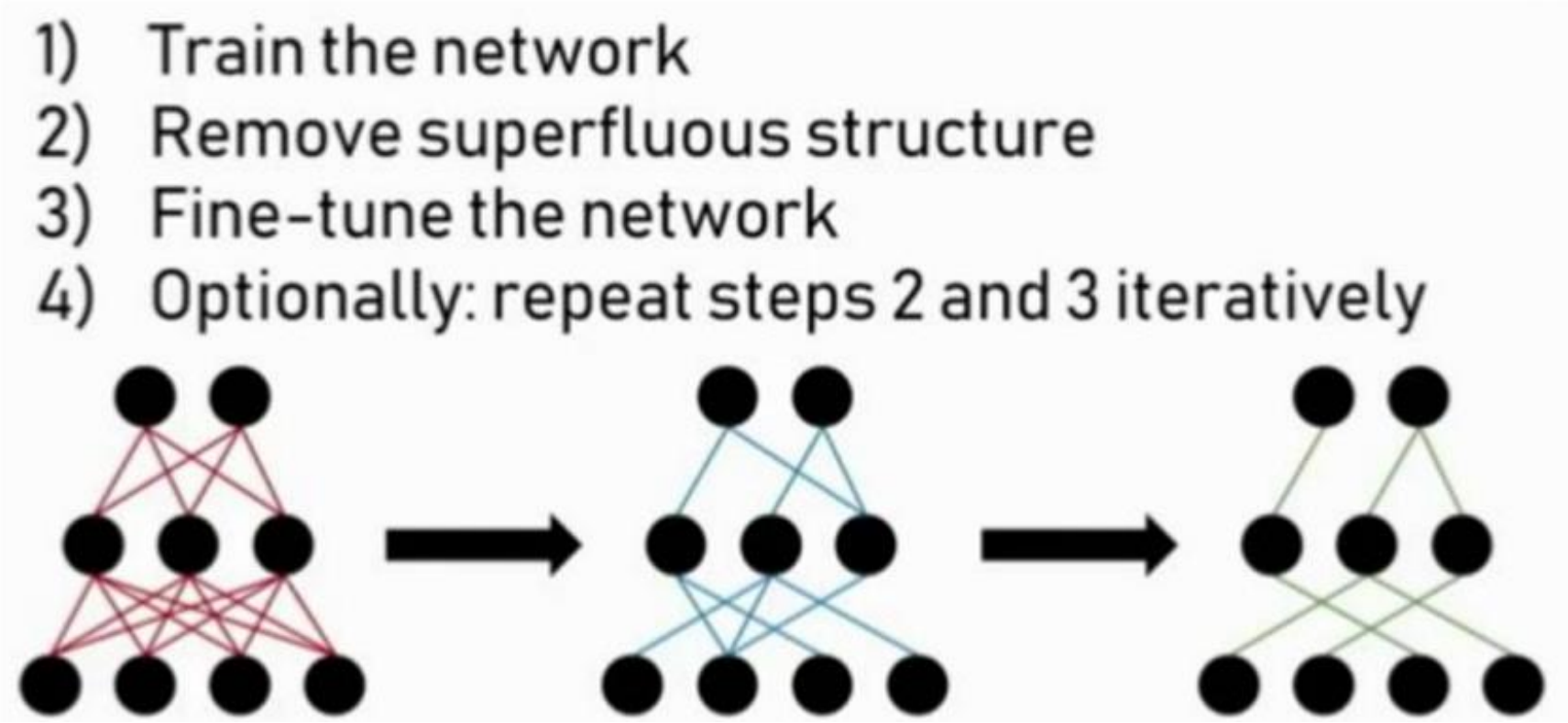
Pruning rate

- $p\%$ es la tasa de podado (pruning rate)
- P_m es la dispersion de la red podada.

Ejemplo: $P_m = 25\%$ cuando $p\% = 75\%$ de los pesos son podados. En este caso, la tasa de compresión es $\frac{1}{P_m} = 4$.

Método basado en magnitud: podado iterativo y re-entrenar





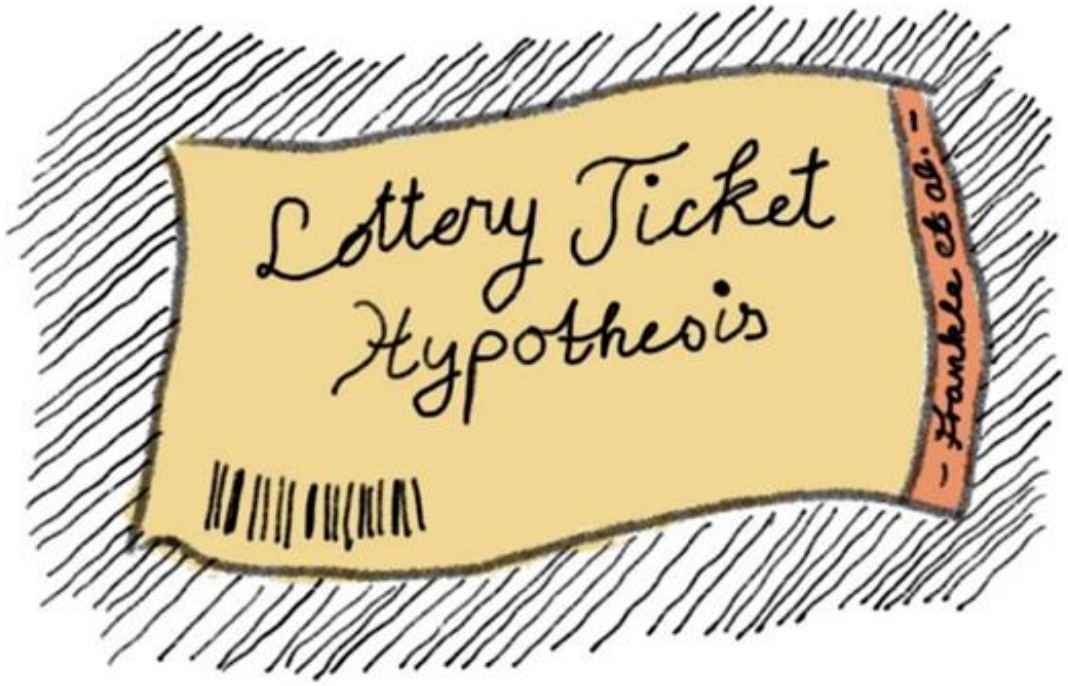
Han, Song, et al. "Learning both weights and connections for efficient neural network." NIPS. 2015.

Método basado en magnitud: podado iterativo y re-entrenar

Network	Top-1 Error	Top-5 Error	Parameters	Compression Rate
LeNet-300-100 Ref	1.64%	-	267K	12X
LeNet-300-100 Pruned	1.59%	-	22K	
LeNet-5 Ref	0.80%	-	431K	12X
LeNet-5 Pruned	0.77%	-	36K	
AlexNet Ref	42.78%	19.73%	61M	9X
AlexNet Pruned	42.77%	19.67%	6.7M	
VGG-16 Ref	31.50%	11.32%	138M	13X
VGG-16 Pruned	31.34%	10.88%	10.3M	

Han, Song, et al. "Learning both weights and connections for efficient neural network." NIPS. 2015.

Lottery ticket hypothesis

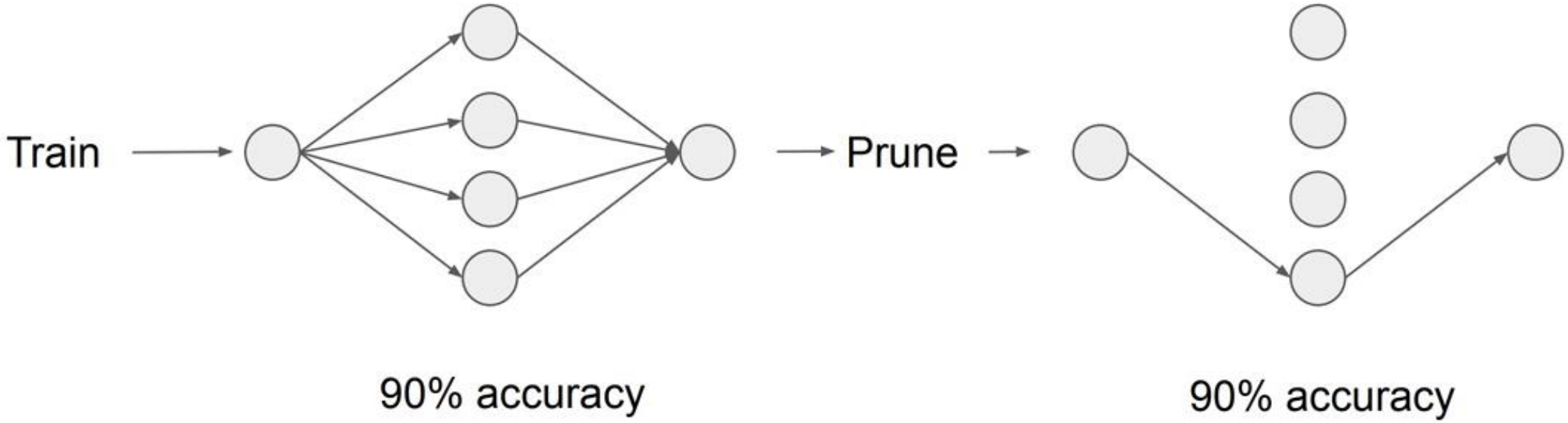


Jonathan Frankle
MIT CSAIL
jfrankle@csail.mit.edu

Michael Carbin
MIT CSAIL
mcarbin@csail.mit.edu

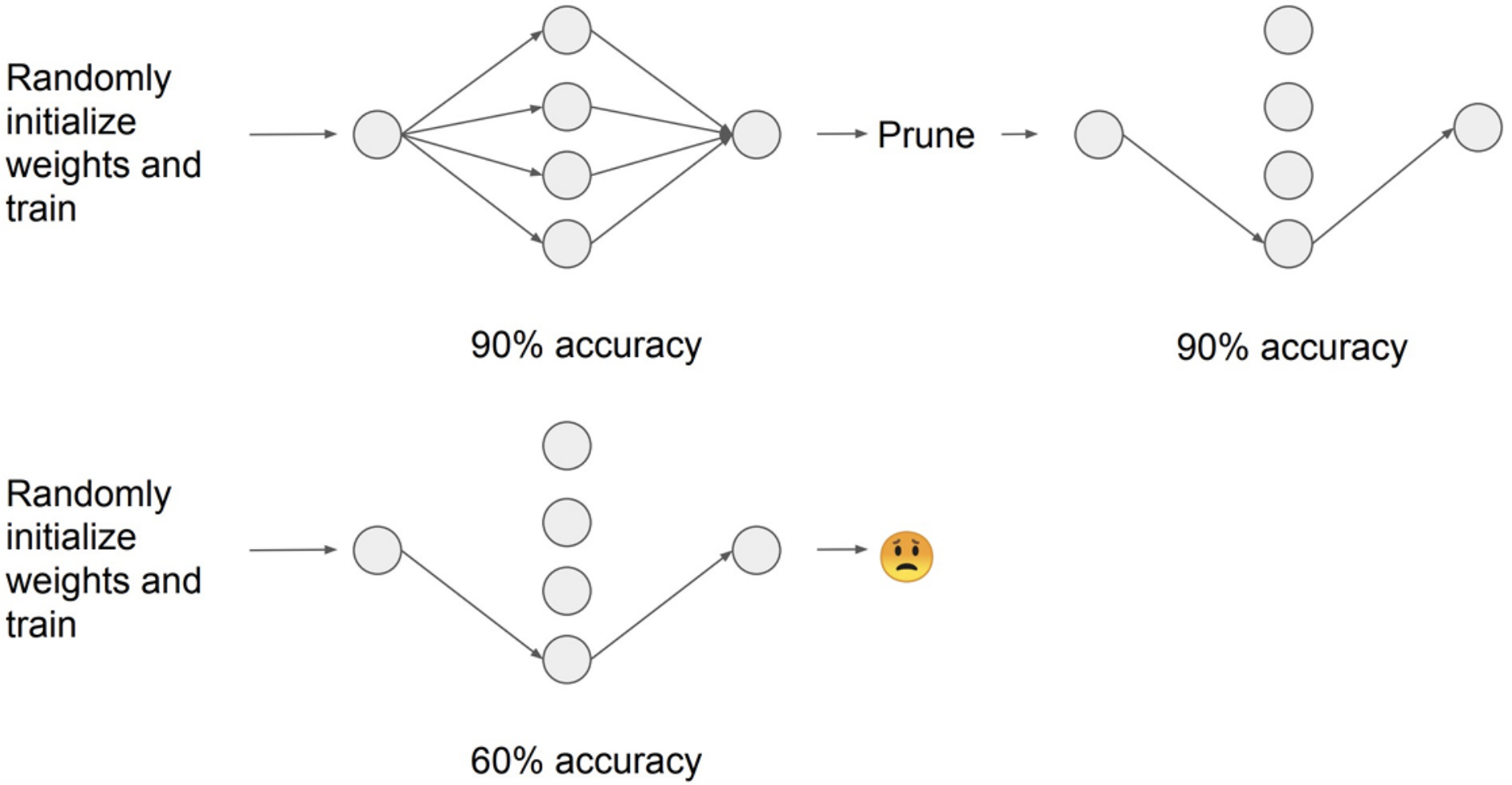
Lottery ticket hypothesis - motivación

Las técnicas de podado pueden reducir la cantidad de parámetros en un 90% sin perjudicar el acierto.



https://ndey96.github.io/deep-learning-paper-club/slides/Lottery%20Ticket%20Hypothesis%20slides.pdf

Lottery ticket hypothesis - motivación



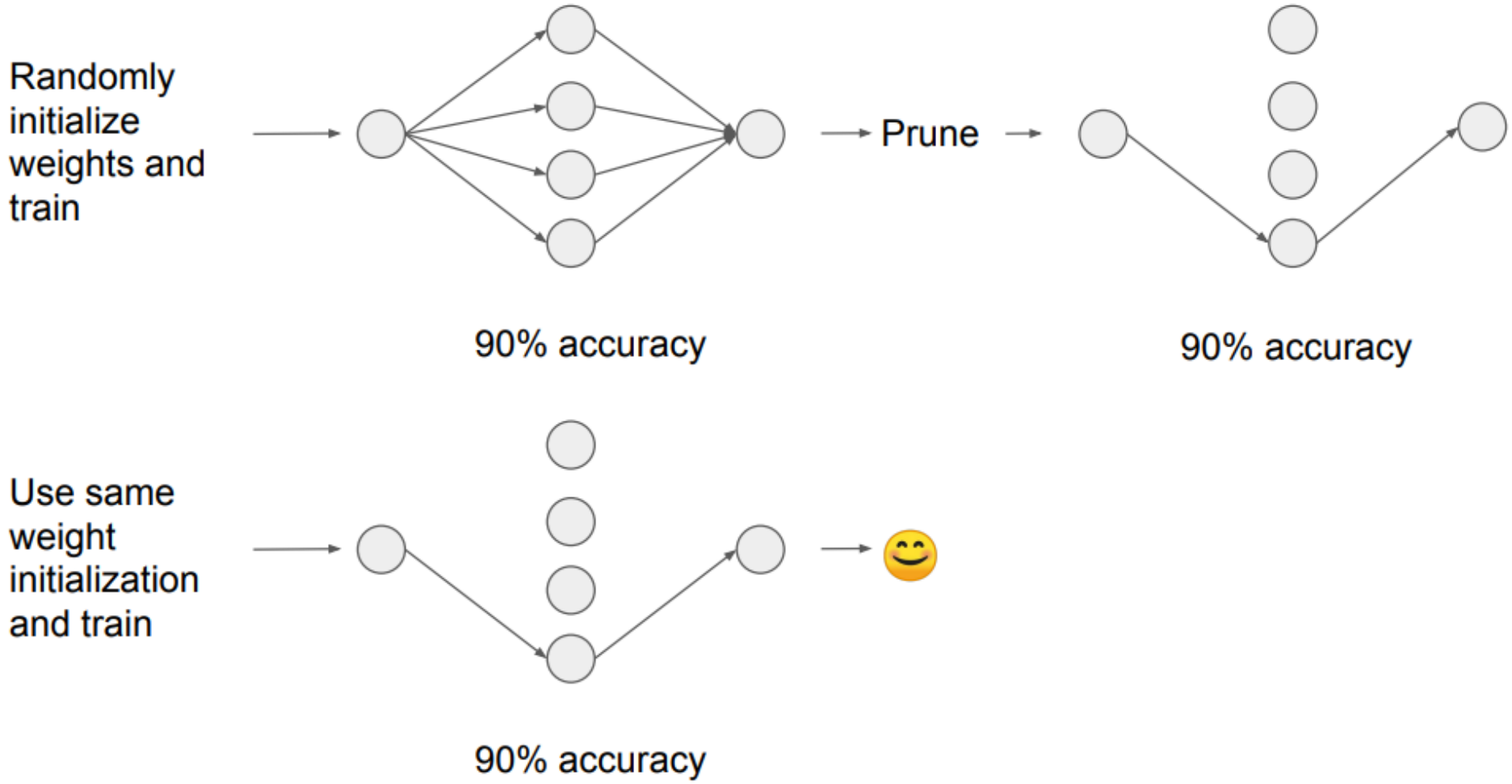
https://ndey96.github.io/deep-learning-paper-club/slides/Lottery%20Ticket%20Hypothesis%20slides.pdf

Lottery ticket hypothesis - motivación

A randomly-initialized, dense neural network contains a subnetwork that is initialized such that—when trained in isolation—it can match the test accuracy of the original network after training for at most the same number of iterations.

https://ndey96.github.io/deep-learning-paper-club/slides/Lottery%20Ticket%20Hypothesis%20slides.pdf

Lottery ticket hypothesis - motivación



https://ndey96.github.io/deep-learning-paper-club/slides/Lottery%20Ticket%20Hypothesis%20slides.pdf

Lottery ticket hypothesis - motivación

- Si quieres ganar la loteria, solo debes comprar un monton de tiquetes y algunos probablemente ganaran.
- Comprar un monto de tiquetes = tener una red neuronal sobre-parametrizada para la tarea.
- Ganar la loteria = entrenar una red con un alto acierto.
- Tiquete ganador = subred podada que alcanza un alto acierto.

https://ndey96.github.io/deep-learning-paper-club/slides/Lottery%20Ticket%20Hypothesis%20slides.pdf

Métodos de podado

- One-shot pruning:
  1. Randomly initialize a neural network $f(x; \theta_0)$, with initial parameters θ_0
  2. Train the network for j iterations, arriving at parameters θ_j
  3. Prune $p\%$ of the parameters in θ_j, creating a mask m
  4. Reset the remaining parameters to their value in θ_0, creating the winning ticket $f(x; m \odot \theta_0)$.
- Iterative pruning:
  1. Randomly initialize a neural network $f(x; \theta_0)$, with initial parameters θ_0
  2. Train the netowork for j iterations, arriving at parameters θ_j
  3. Prune $p^{1/n}\%$ of the parameters in θ_j, creating a mask m
  4. Reset the remaining parameters to their value in θ_0, creating network $f(x; m \odot \theta_0)$
  5. Repeat n times from 2
  6. Final network is a winning ticket $f(x; m \odot \theta_0)$.

https://ndey96.github.io/deep-learning-paper-club/slides/Lottery%20Ticket%20Hypothesis%20slides.pdf

Experimentos

- MLP for MNIST
- CNN for CIFAR10
- Ablation Study (dropout, weight decay, optimizer …)

Network	Lenet	Conv-2	Conv-4	Conv-6	Resnet-18	VGG-19
Convolutions		64, 64, pool	64, 64, pool 128, 128, pool	64, 64, pool 128, 128, pool 256, 256, pool	16, 3x[16, 16] 3x[32, 32] 3x[64, 64]	2x64 pool 2x128 pool, 4x256, pool 4x512, pool, 4x512
FC Layers	300, 100, 10	256, 256, 10	256, 256, 10	256, 256, 10	avg-pool, 10	avg-pool, 10
All/Conv Weights	266K	4.3M / 38K	2.4M / 260K	1.7M / 1.1M	274K / 270K	20.0M
Iterations/Batch	50K / 60	20K / 60	25K / 60	30K / 60	30K / 128	112K / 64
Optimizer	Adam 1.2e-3	Adam 2e-4	Adam 3e-4	Adam 3e-4	← SGD 0.1-0.01-0.001 Momentum 0.9 →	
Pruning Rate	fc20%	conv10% fc20%	conv10% fc20%	conv15% fc20%	conv20% fc0%	conv20% fc0%

Resultados MLP – (LeNet)

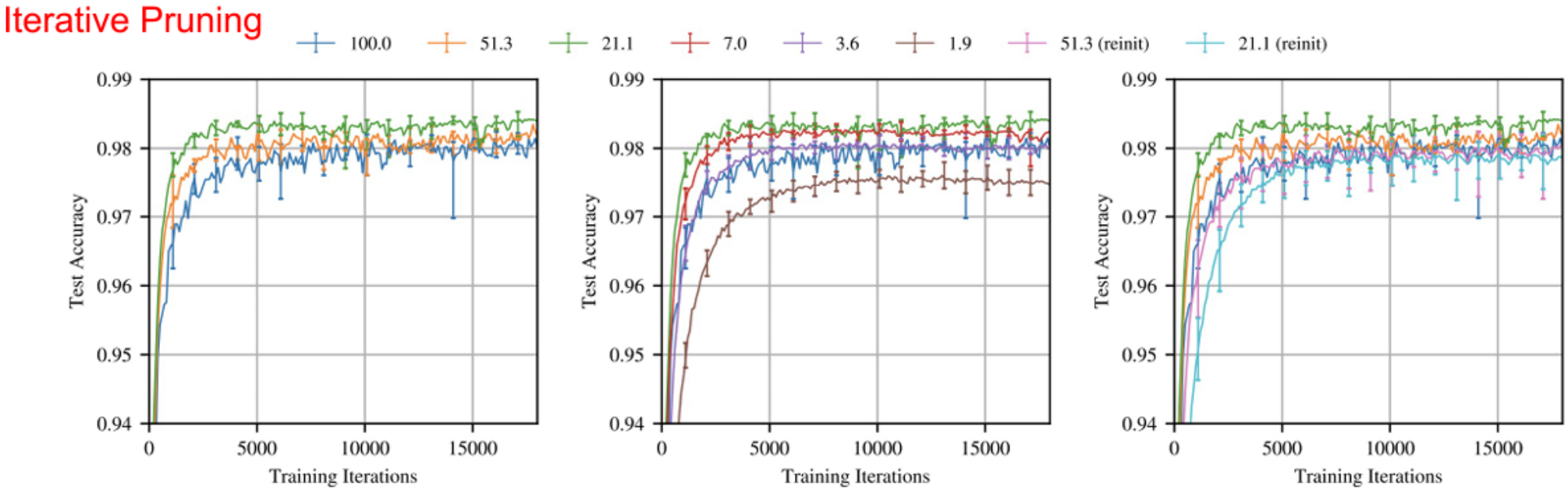


Figure 3: Test accuracy on Lenet (iterative pruning) as training proceeds. Each curve is the average of five trials. Labels are P_m—the fraction of weights remaining in the network after pruning. Error bars are the minimum and maximum of any trial.

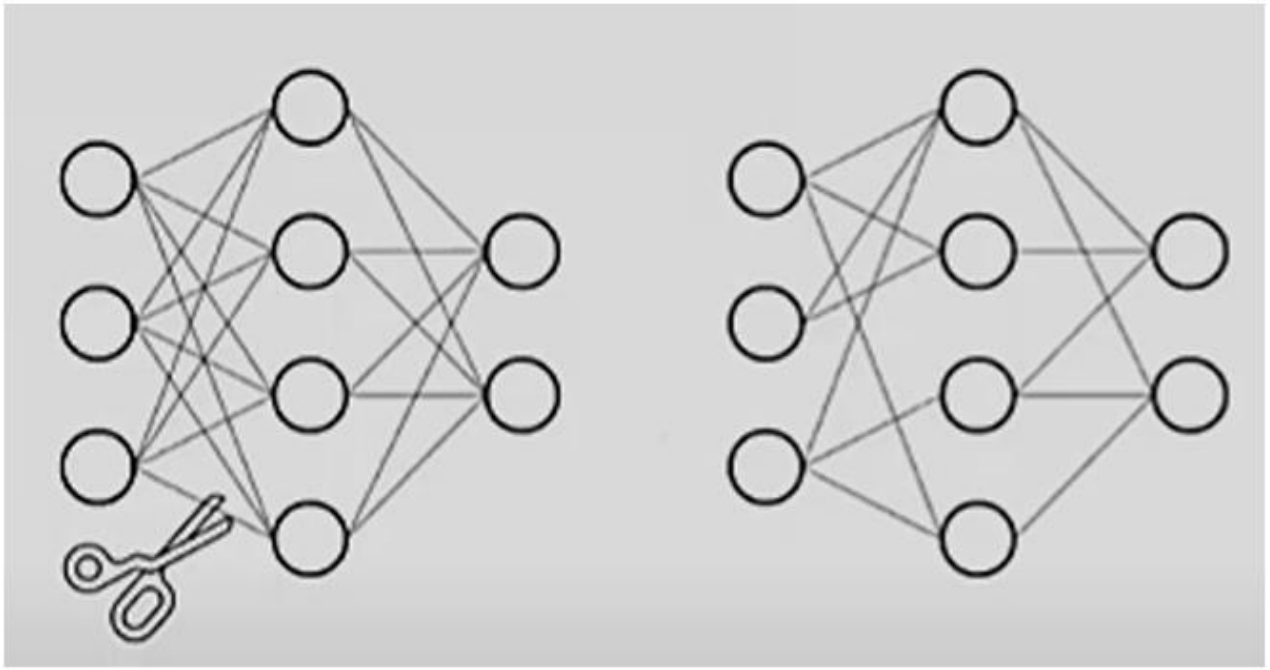
- 51.3%, 21.12% es major que 100%, 3.6% es comparable con 100%.
- Winning ticket es mejor que la reinicializacion.

Lottery ticket hypothesis - limitaciones

- Solo se probaron bases de datos pequeñas.
- El podado iterativo es intensivo computacionalmente (alrededor de 15x).
- Para los métodos Structured pruning y non-magnitude pruning.
- Faltan estudios sobre las propiedades de las inicializaciones.

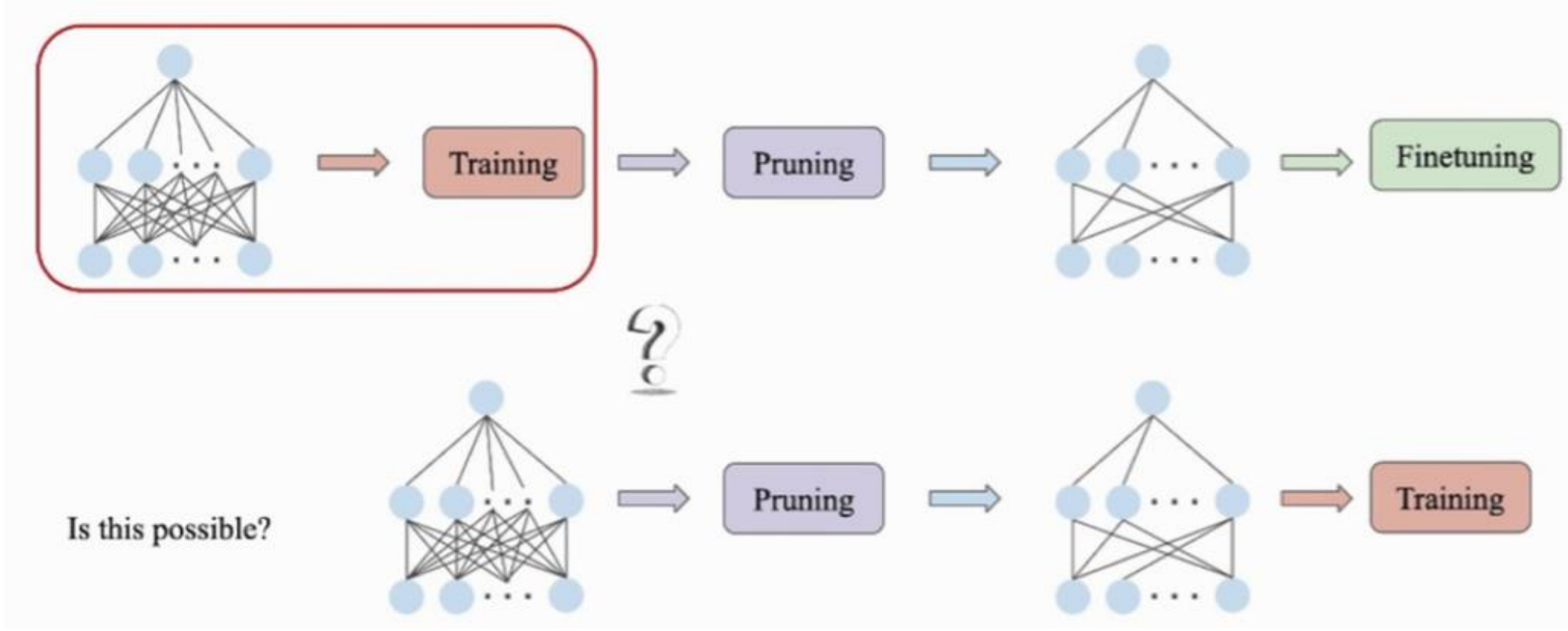
https://ndey96.github.io/deep-learning-paper-club/slides/Lottery%20Ticket%20Hypothesis%20slides.pdf

SNIP: Single-Shot Network Pruning Based On Connection Sensitivity



Namhoon Lee, Thalaiyasingam Ajanthan & Philip H. S. Torr
University of Oxford

SNIP: Single-Shot Network Pruning Based On Connection Sensitivity



- Metodos basados en el destacamiento: remover parámetros (conexiones) redundates de la red neuronal de manera selectiva.

Sensibilidad de la conexión

- Esperamos podar la red antes del entrenamiento.
- Denotamos $c_j \in \{0,1\}$ como el indicador de la conexión

$$\min_{\mathbf{c},\mathbf{w}} L(\mathbf{c} \odot \mathbf{w}; \mathcal{D}) = \min_{\mathbf{c},\mathbf{w}} \frac{1}{n} \sum_{i=1}^{n} \ell(\mathbf{c} \odot \mathbf{w}; (\mathbf{x}_i, \mathbf{y}_i)) \,,$$
$$\text{s.t.} \quad \mathbf{w} \in \mathbb{R}^m \,,$$
$$\mathbf{c} \in \{0,1\}^m, \quad \|\mathbf{c}\|_0 \leq \kappa \,,$$

Sensibilidad de la conexión

- Nos enfocamos en la diferencia de la funcion objetivo cuando c_j se cambia, podemos determinar la importancia de la conexión.
- La conexion j esta activa cuando $c_j = 1$; esta podada cuando $c_j = 0$.
- Podemos medir es efecto de forma precisa con

$$\Delta L_j(\mathbf{w}; \mathcal{D}) = L(\mathbf{1} \odot \mathbf{w}; \mathcal{D}) - L((\mathbf{1} - \mathbf{e}_j) \odot \mathbf{w}; \mathcal{D})$$

- $\boldsymbol{e}_j$ es un vector de ceros en todas partes, excepto en el indice j donde es 1.

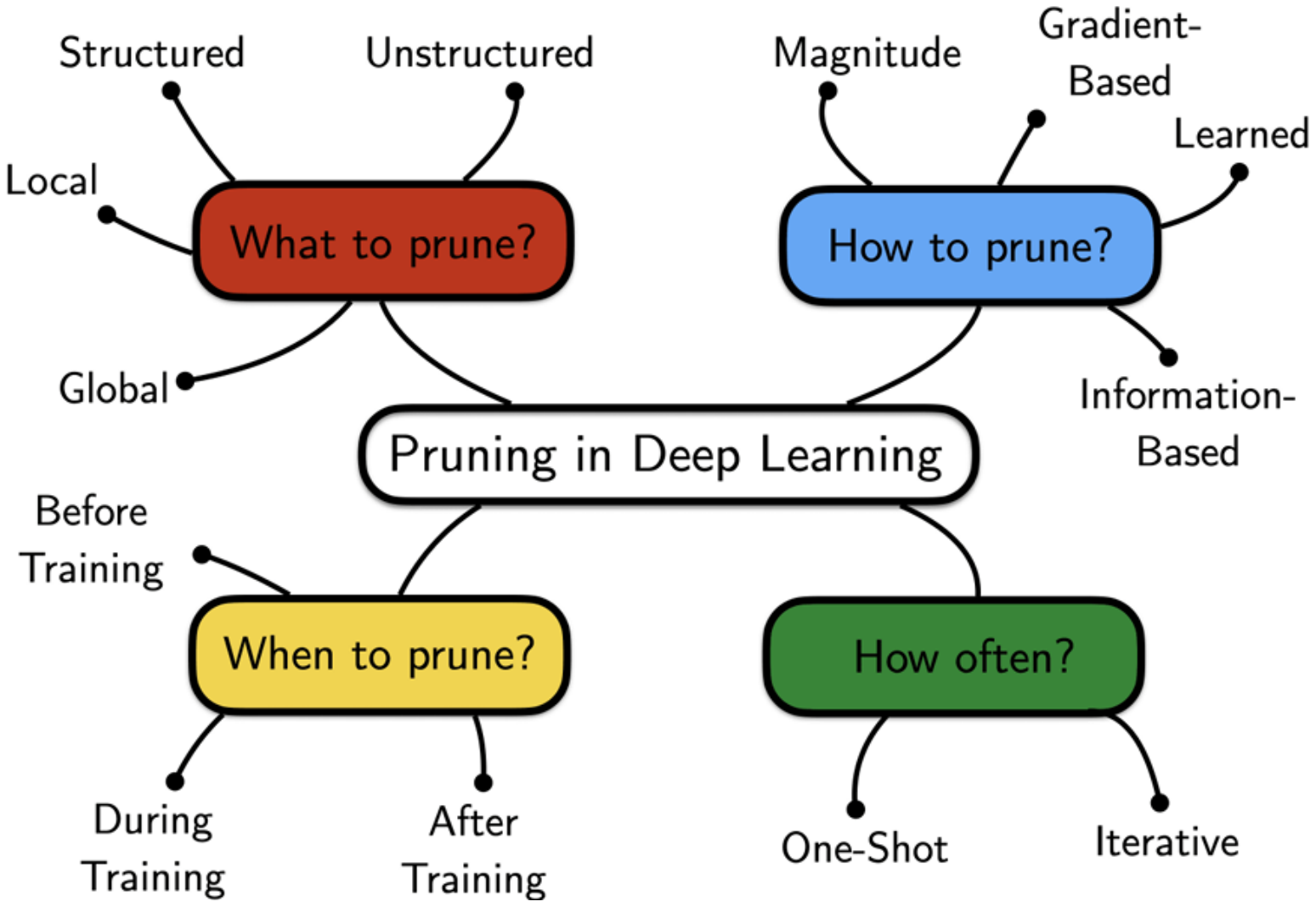
Sensibilidad de la conexión

$$\min_{\mathbf{c},\mathbf{w}} L(\mathbf{c} \odot \mathbf{w}; \mathcal{D}) = \min_{\mathbf{c},\mathbf{w}} \frac{1}{n} \sum_{i=1}^{n} \ell(\mathbf{c} \odot \mathbf{w}; (\mathbf{x}_i, \mathbf{y}_i)) \,,$$
$$\text{s.t.} \quad \mathbf{w} \in \mathbb{R}^m \,,$$
$$\mathbf{c} \in \{0,1\}^m, \quad \|\mathbf{c}\|_0 \leq \kappa \,,$$

- Debido que c_j es binario, no se puede diferenciar y es dificil de optimizar, podemos solucionar el problema relajandolo.

$$\Delta L_j(\mathbf{w}; \mathcal{D}) \approx g_j(\mathbf{w}; \mathcal{D}) = \left. \frac{\partial L(\mathbf{c} \odot \mathbf{w}; \mathcal{D})}{\partial c_j} \right|_{\mathbf{c}=\mathbf{1}} = \lim_{\delta \to 0} \left. \frac{L(\mathbf{c} \odot \mathbf{w}; \mathcal{D}) - L((\mathbf{c} - \delta\, \mathbf{e}_j) \odot \mathbf{w}; \mathcal{D})}{\delta} \right|_{\mathbf{c}=\mathbf{1}}$$

Resumen



Documentación y ejemplos

https://pytorch.org/tutorials/intermediate/pruning_tutorial.html

https://intellabs.github.io/distiller/pruning.html

https://colab.research.google.com/github/pytorch/tutorials/blob/gh-pages/_downloads/f40ae04715cdb214ecba048c12f8dddf/pruning_tutorial.ipynb

https://colab.research.google.com/github/tensorflow/model-optimization/blob/master/tensorflow_model_optimization/g3doc/guide/pruning/pruning_with_keras.ipynb

https://colab.research.google.com/github/matthew-mcateer/Keras_pruning/blob/master/Model_pruning_exploration.ipynb

https://colab.research.google.com/github/sayakpaul/Adventures-in-TensorFlow-Lite/blob/master/Model_Pruning_in_Deep_Learning_with_tfmot.ipynb

Institución Universitaria
Reacreditada en Alta Calidad

¡Gracias!

Somos Innovación Tecnológica con Sentido Humano

Alcaldía de Medellín

Sensibilidad de la conexión

$$\Delta L_j(\mathbf{w};\mathcal{D}) \approx g_j(\mathbf{w};\mathcal{D}) = \left.\frac{\partial L(\mathbf{c}\odot\mathbf{w};\mathcal{D})}{\partial c_j}\right|_{\mathbf{c}=\mathbf{1}}$$

- Debido que c_j es binario, no se puede diferenciar y es dificil de optimizar, podemos solucionar el problema relajandolo.

Bibliografía

1. Schizas, N.; Karras, A.; Karras, C.; Sioutas, S. *TinyML for Ultra-Low Power AI and Large Scale IoT Deployments: A Systematic Review*. *Future Internet* **2022**, *14*, 363. https://doi.org/10.3390/fi14120363
2. Atul K. Gupta and Dr. Siva P. Nandyala. *Deep Learning on Microcontrollers: Learn how to develop embedded AI applications using TinyML*. 2023. ISBN 978-93-55518-057.