- Network pruning arised to address the need to reduce large networks and make possible better use of available resources.
- Generally pruning algorithm consists of 3 steps : training, pruning, and fine-tuning remaining network. While pruning, the most redundant weights are removed to retain maximum accuracy.
- Observations state that fine-tuning a pruned network leads to equal or worse performance, so it is better to train the target network directly with randomly initialised weights. Also, the pruning can be seen as finding better network architecture instead of saving most important weights.
- Mainly pruning is done based on two **beliefs** :
    - Training larger network with stronger representation and optimization power and removing redundant weights may not hurt performance much.
    - Fine-tuning pruned network with (so called) important weights is better than training pruned network from scratch.
- Two types of pruning :
    - Structured : Pruning larger part of the network like channel or layer.
        - Predefined target network : user fixed fraction of each layer is pruned. Training this network from scratch could achieve the same performance.
        - Auto-discovered target network : algorithm decides fraction for each layer. Training this network from scratch sometimes gave better performance.
    - Unstructured : Pruning less important connections between neurons. Training this network from scratch could not perform good with large datasets.
- Using important (by pruning algorithm) weights may be sub optimal, but using the algorithm to find better networks implicitly.
- Problem with unstructured pruning is that they form sparse weight matrices which are difficult to compress and put GPUs to better use, whereas structured pruning preserves the network architecture and can be compressed.
- Channel pruning (most popular) can be done using sparsity constraints on channels, feature reconstruction error of channels, channels' influence on final loss etc.
- Studies show that random channel pruning can perform as good as other complicated channel pruning methods.
- Lottery ticket hypothesis : Training sub network in isolation can reach performance of original network within same number of epochs.
    - Training sub network with randomly initialised weights gave poor performance.
    - Training it with initial weights of unpruned network gave great performance. So training large no. of weights (buying lot of tickets) increases chance of finding optimal weights for sub network (winning ticket).
    - So, training large sparse model is better than training small model from scratch.
    - The fact that pruned network trained from scratch can perform as good as finetuned model shows that the **pruned network** deserves more credit.

- Training smaller models from scratch:
    - Training smaller model for same number of epochs is not good as it leads to less computation for pruned model. So, pruned model is trained until it reaches the same number of FLOPs as required by the original model.
    - Generally it is believed that smaller model converges quickly, but training it for more epochs within a reasonable range does not harm.
- Predetermined target model pruning methods :
    - L1-norm based Filter pruning : Removing some fraction filters with least L1 norm.
    - ThiNet : Greedily pruning filters which affect next layer activations least calculated by adding noise to input layer.
    - Regression based Feature Reconstruction : Lasso regression is used to minimize feature reconstruction error of next layer.
- Automatic structured pruning methods :
    - Network slimming : It applies L1 sparsing based on scaling factors of different BatchNorm layers' channels and removes the ones with least scaling factor.
    - Sparse Structure Selection : It generalises network slimming and removes some residual blocks or layers.
- In the structured pruning, pruned model from scratch outperformed 3-stage pipeline. Training the model for same no. of epochs gave worse results than for same no. of FLOPs.
- Unstructured pruning could not outperform the fine tuned model for large datasets. It could be due to complexity of the data or the observation that unstructured pruning significantly changes weights distribution.
- Experiments show that network slimming is way more parameter efficient (no. of params Vs accuracy) than predefined target pruning. Also unstructured pruning is more parameter efficient than uniform sparsification.
- Network slimming could outperform uniform pruning in the case of VGG, but not for ResNets and DenseNets. Analysis shows that the sparsity patterns are nearly uniform for ResNets and DenseNets but different for VGG. So, VGG's redundancy is imbalanced across the layers and network slimming may help more in such cases.
- Guided pruning (using average sparsity patterns to design network) and Transferred Guided pruning (using sparsity patterns of a network on some dataset with different dataset) also gave parameter efficient results.
- Lottery ticket hypothesis and this paper contradict but there are some differences in evaluation settings :
    - LTH was on unstructured pruning but this was on both pruning methods.
    - LTH used shallow nets but this paper used modern deep nets.
    - LTH used Adam with low LR but this paper used SGM+Momentum with high LR.
    - LTH was only on MNIST and CIFAR, but this paper was also on Imagenet.
- It was analysed that the differences in both theories arises due to different LRs (0.1 and 0.01). LTH gives better results for 0.01 LR and RNP gives better for 0.1 LR. But 0.01 LR performance is inferior to that of 0.1 LR.

- LTH may give better results for low LR with original initialisation as the final weights may not be far away from original initialisation due to small update stepsize.
- Comparing LTH and RNP results :
    - Structured pruning : For both LRs, RNP is at par with LTH.
    - Unstructured : RNP is better for larger LR (better optimum), and LTH is better on lower LR.
- Soft Filter Pruning (SFP) : It is a predefined target method. It prunes new weights after every epoch and also adds some pruned weights to the network. It outperforms the pruned models in almost all prune ratios.
- Transfer Learning for detection : Generally 2 ways for transfer learning : a) prune model, fine-tune for classification, then transfer to detection; b) transfer to detection, prune model, then fine-tune for detection. Training pruned model (L1 norm based) and transferring to detection outperformed both a and b ways.
    - Interestingly a outperforms b, which is hypothesised to be because of removing weights from an earlier stage (classification as in a) prevents the model to stop in bad optima. This hypothesis is in line with pruned models, where redundant weights are removed from beginning itself.
- Aggressive pruning : pruned models outperformed fine-tuned models by larger margin with large prune ratios.
- Fine-tuning for more epochs was also tried, but it gave negligible increase in accuracy but could not beat pruned models. Even on extending the standard training schedule, pruned models performed atleast at par with fine-tuned models.
- Analysis on weight distributions show that unpruned models have more weights close to zero. In unstructured pruning, fine-tuned models have no close to zero values, maybe due to which they outperform pruned models sometimes.