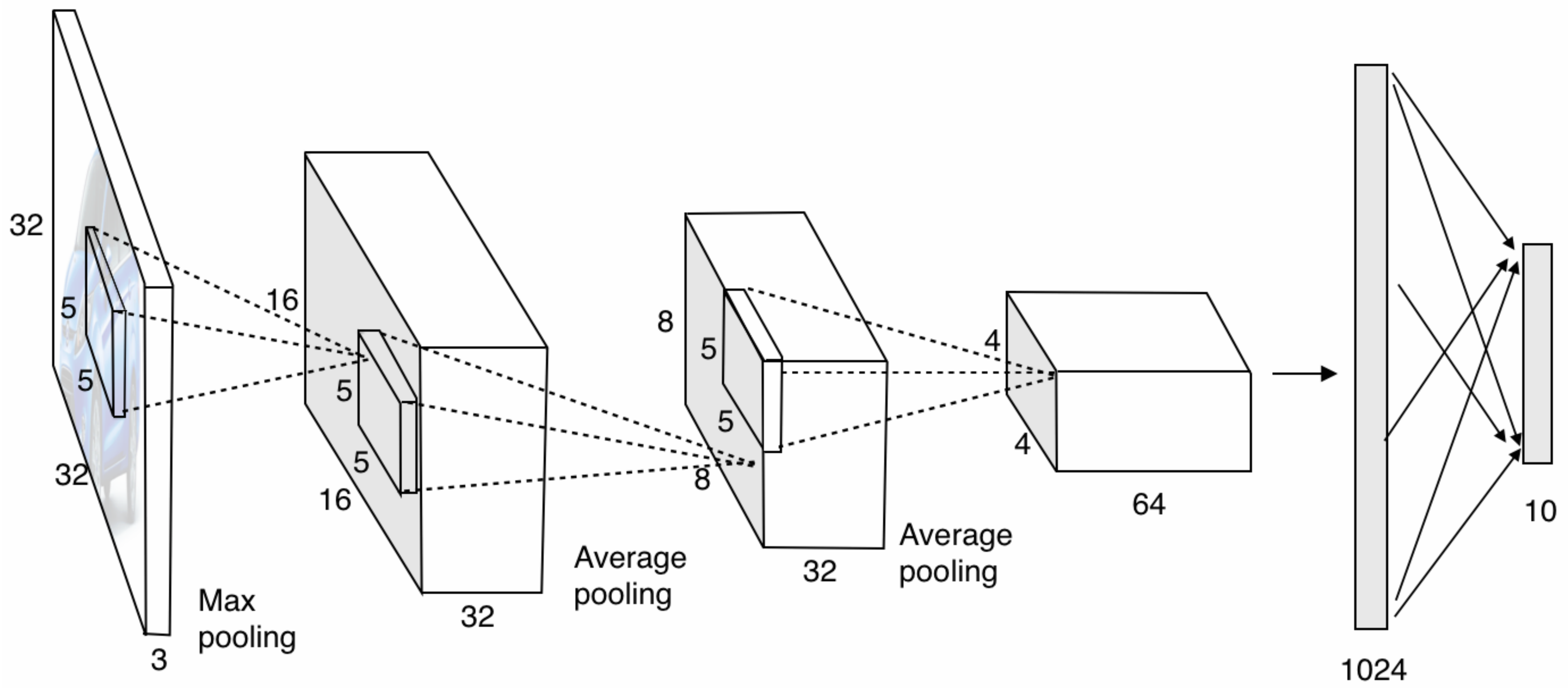
Convolutional Neural Network and Convex Optimization

Si Chen *and* Yufei Wang
{sic046, yuw176}@ucsd.edu

Convolutional neural network(CNN)

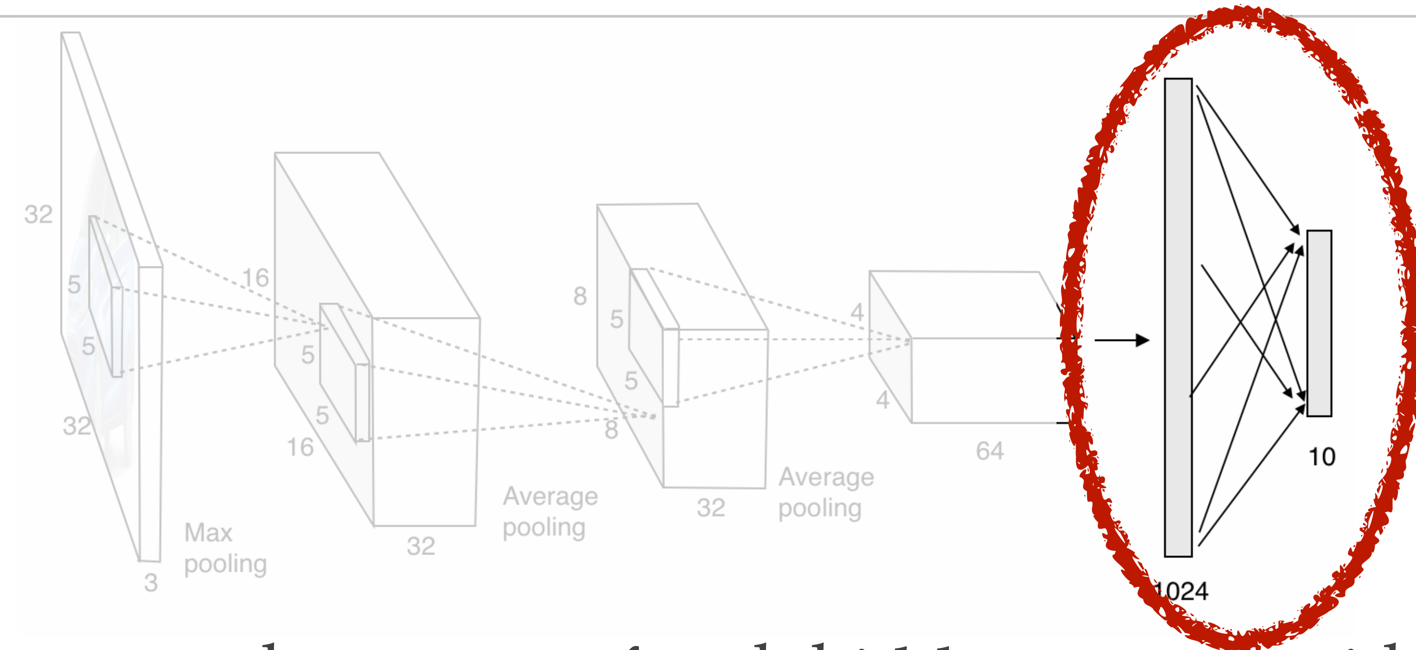
- Consists of multiple layers.
- Convolutional layer:
  - $h_{ij}^{k} = \mathrm{f}((W^{k} * x)_{ij} + b_{k})$, $f(x) = \max(0, x)$
- Pooling layer: max-pooling / average-pooling
- Last layer: logistic regression layer:
  - Output: $P(Y = i|x, W, b) = softmax_i(Wx + b) = \frac{e^{W_i x + b_i}}{\sum_j e^{W_j x + b_j}}$
  - Loss function: $L = \sum_{i=0}^{|D|} \log(P(Y = y^{(i)}|x^{(i)}, W, b))$
  - Prediction: $y_{pred} = \mathrm{argmax}_i P(Y = i|x, W, b)$

Overall architecture



- Dataset: CIFAR-10. 10 classes. 50,000 training. 10,000 test.
- Pooling information: each pooling unit spaced $s = 2$ pixels apart; summarizing a neighborhood of 3×3.

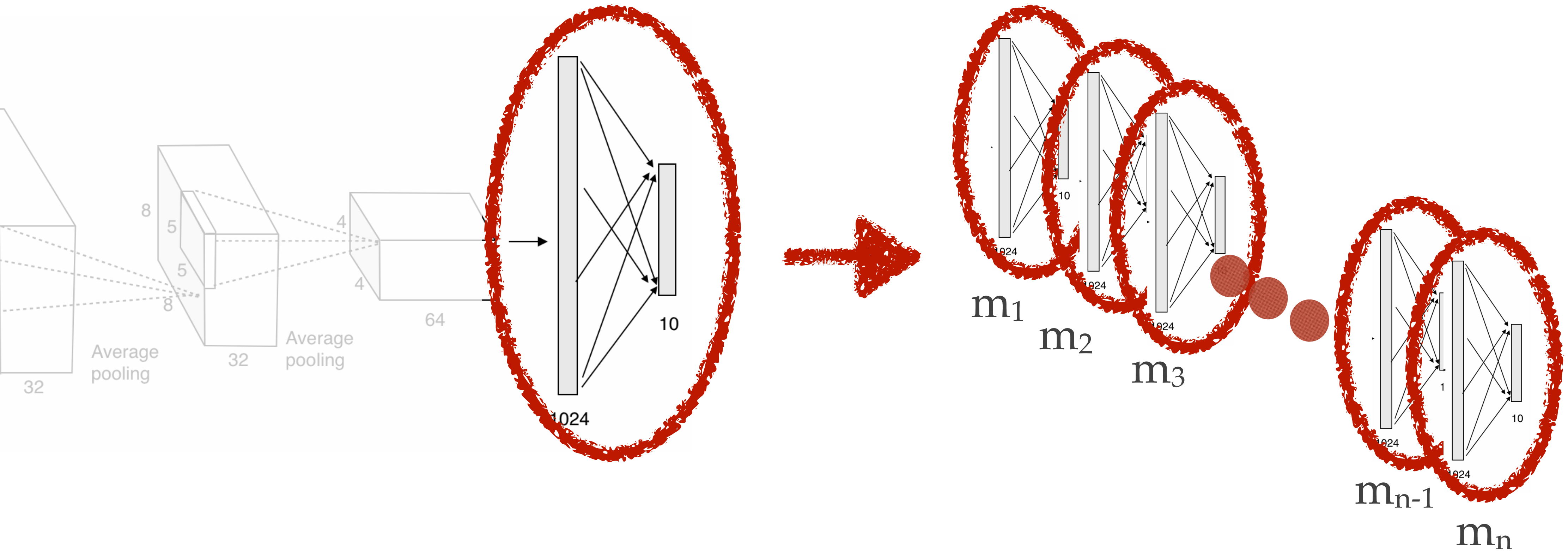
Dropout technique



- Training: setting to zero the output of each hidden neuron with probability of 0.5.

- Sub-model: for every training example, a different architecture
- Test: use all neurons but multiply their outputs by 0.5
  - an approximation to taking the geometric mean of the predictive distributions

From dropout to sub-model combination



- Sub-model: for every training example, a different architecture

- Linear combination of many sub-models:

$$P_{l.comb}(Y = i) = \sum_{k=1}^{n} l_k \times P_{m_k}(Y = i)$$

Sub-model combination: a convex problem

- Optimization problem:

$$\begin{array}{ll} \min\limits_{l} & \sum_{i=1}^{N} \|P_i \cdot l - y_i\|_2^2 \\ \text{s.t.} & l \geq 0 \end{array}$$

- Simplified objective function:

$$\begin{aligned} \sum_{i=1}^{N} \|P_i \cdot l - y_i\|_2^2 &= \sum_{i=1}^{N} (P_i \cdot l - y_i)^t (P_i \cdot l - y_i) \\ &= (P \cdot l - y)^t (P \cdot l - y) \\ &= l^t P^t P l - 2y^t P l + y^t y \end{aligned}$$

$$* \; y = \left[y_1^t, y_2^t, \ldots y_N^t\right]^t$$

$$P = \left[P_1^t, P_2^t, \ldots, P_n^t\right]^t$$

- QP problem:

$$\begin{array}{ll} \min\limits_{l} & l^t P^t P l - 2y^t P l + y^t y \\ \text{s.t.} & l \geq 0 \end{array}$$

Sub-model combination for non-dropout network

- Sub-model in non-dropout network:

$$\Pr(h^j_{m_i} = 2h^j_{orig}) = \Pr(h^j_{m_i} = 0) = \frac{1}{2}.$$

- Same objective function:

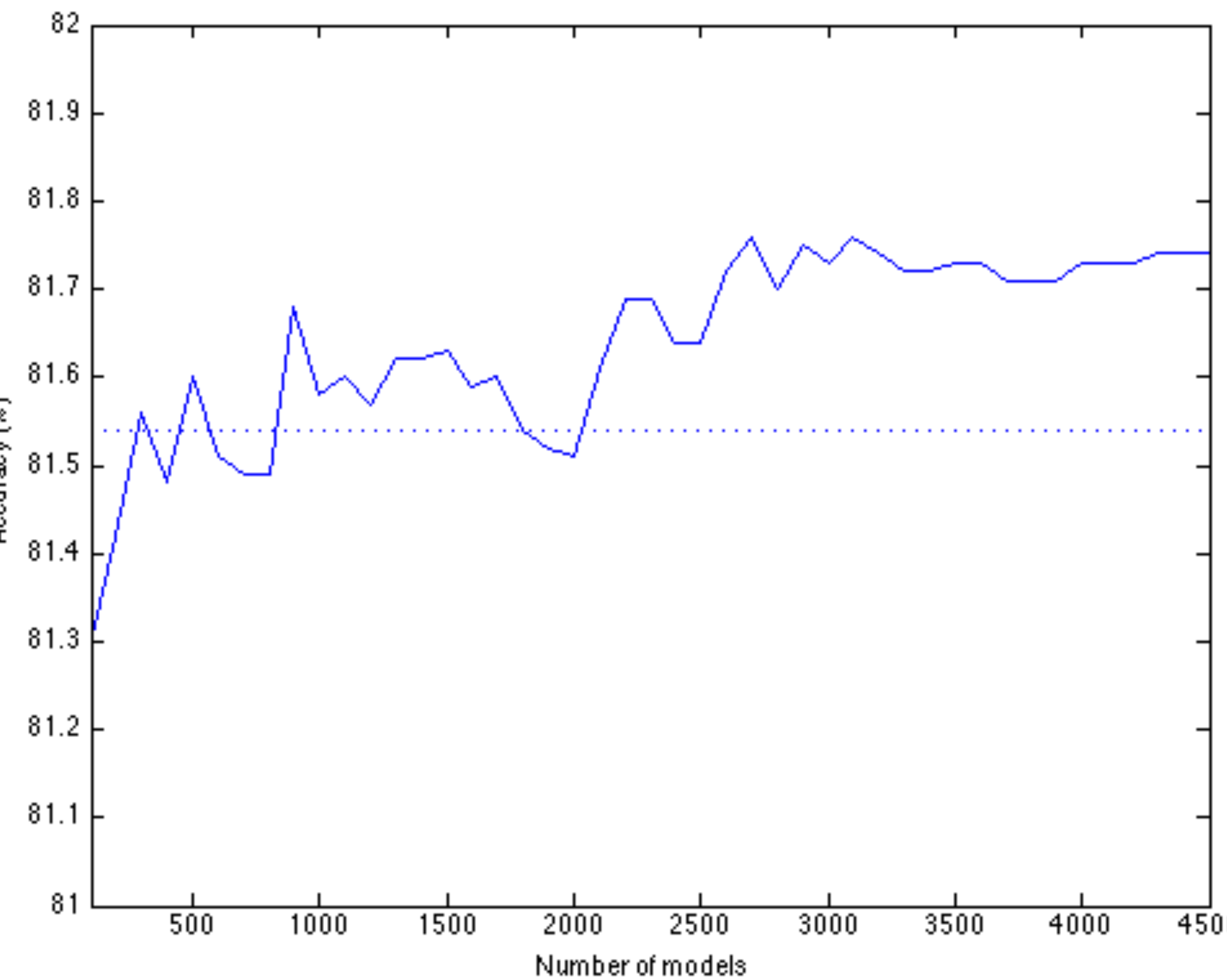
$$\begin{aligned} \min_{l} \quad & l^t P^t P l - 2y^t P l + y^t y \\ \text{s.t.} \quad & l \geq 0 \end{aligned}$$

* h_{m_i} : penultimate layer vector of m-th sub-model

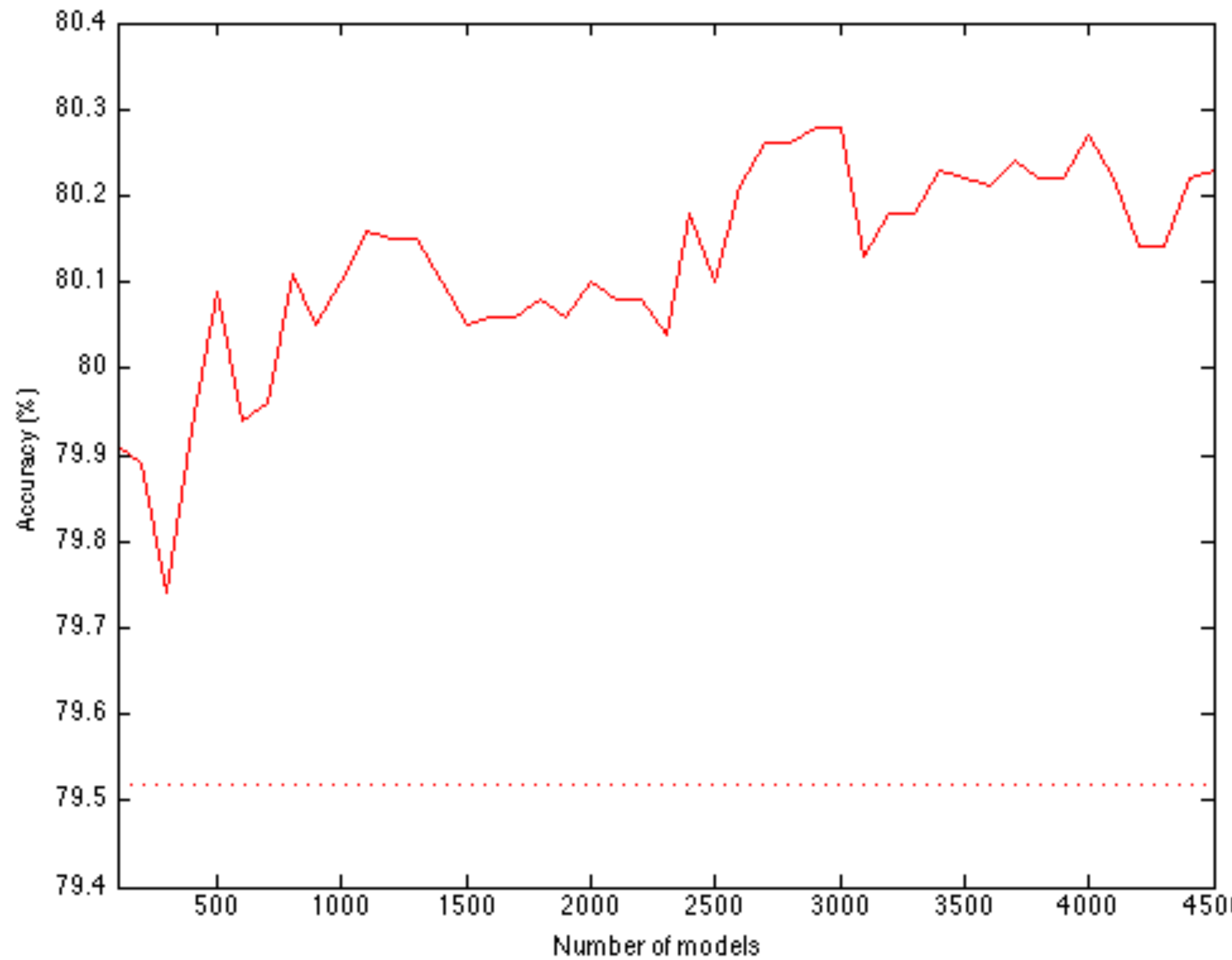
h_{orig}: penultimate layer vector of the trained CNN

Results - the effect of sub-model combination

- Dropout network & non-dropout network
- Randomly chosen 4500 sub-models for each network



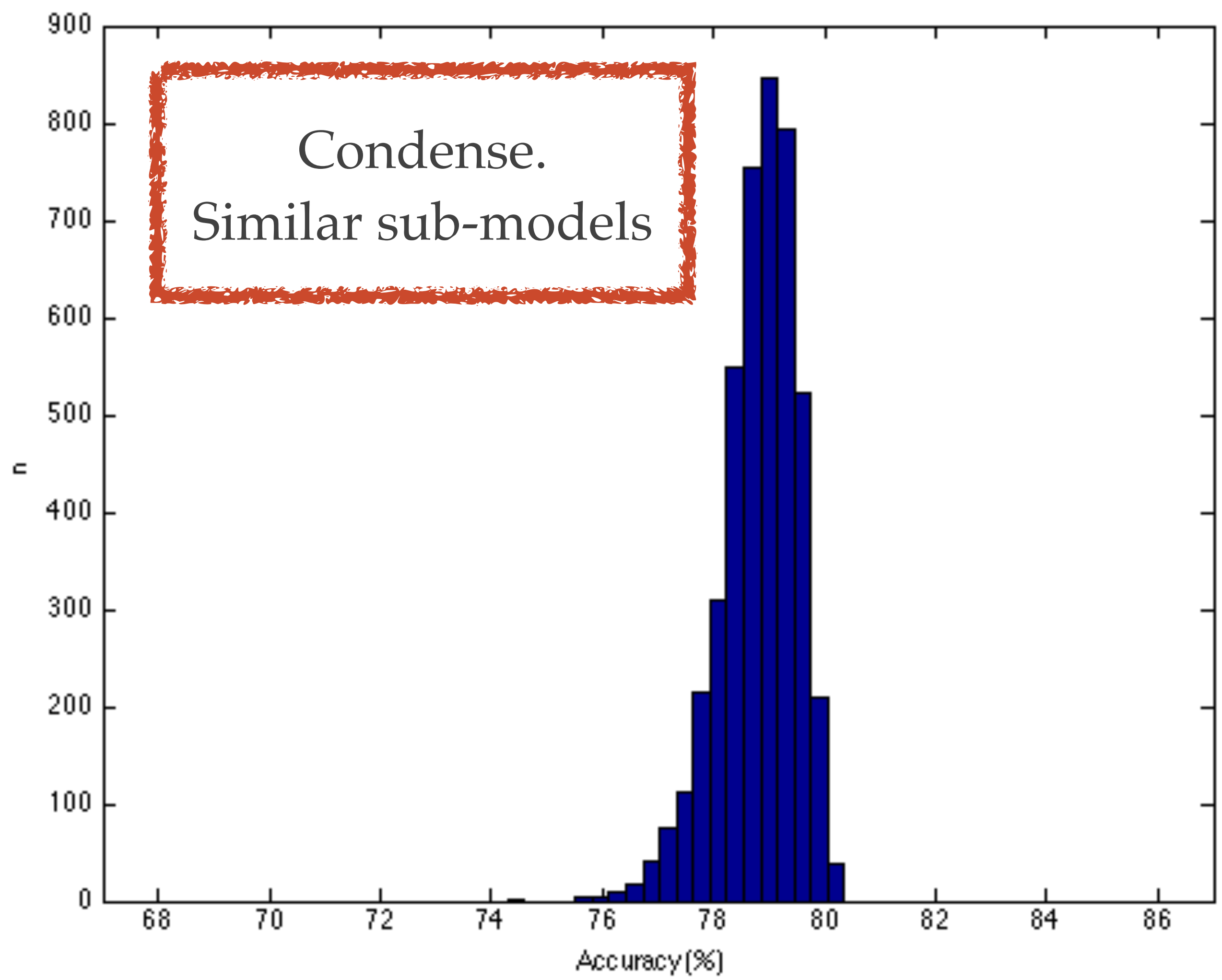
Sub-model combination for dropout network



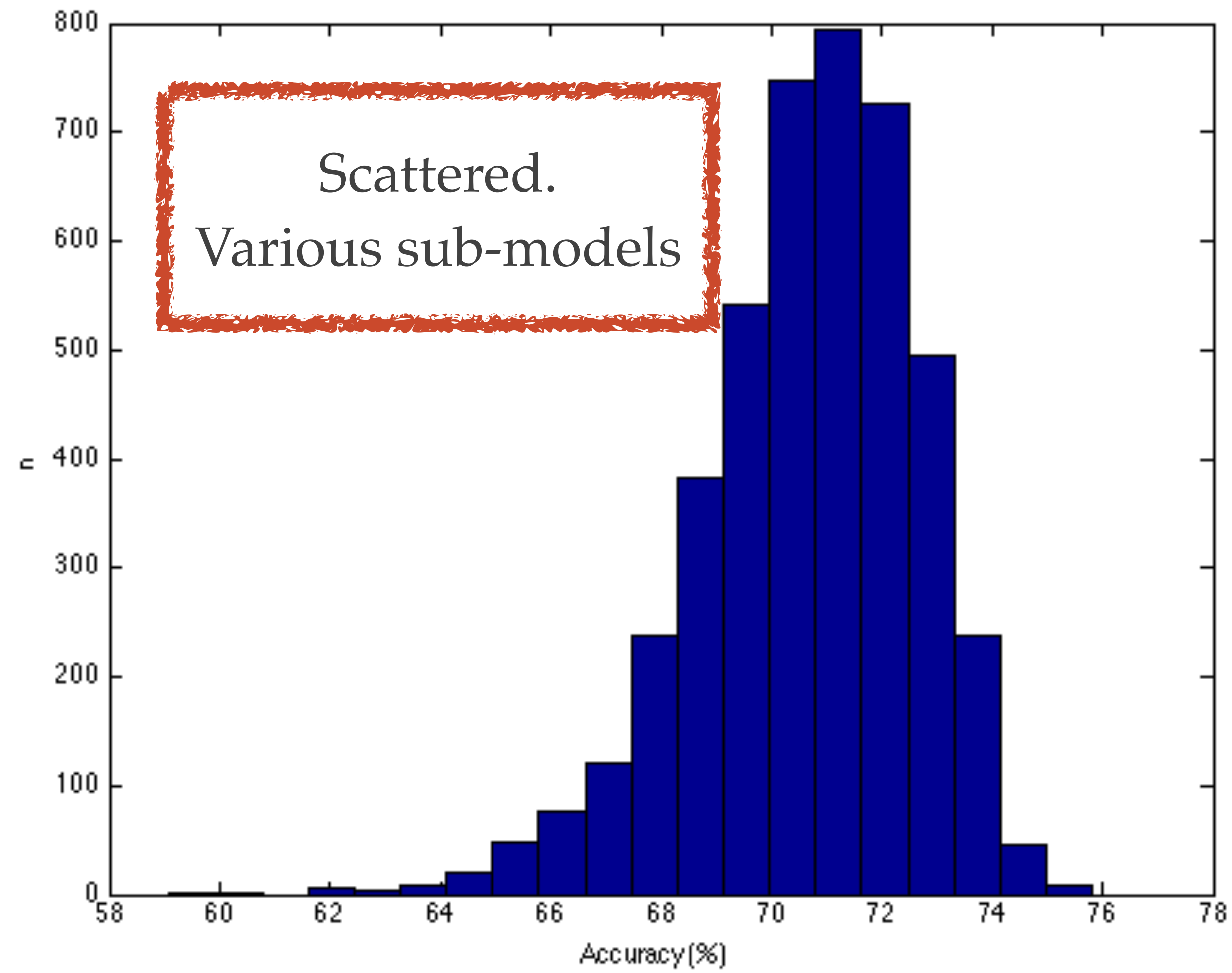
Sub-model combination for non-dropout network

Results - the role of dropout

- Accuracy of each sub-model for both networks:



Accuracies of each sub-model for dropout network



Accuracies of each sub-model for non-dropout network