# System Architecting Approach for Designing Deep Neural Networks

Cihan H Dagli

Missouri University of Science and Technology

# System Architecting Approach for Designing Deep Neural Networks

Ram Deepak Gottapu
Cihan H Dagli

MISSOURI S&T
Smart Engineering Systems Lab
Engineering Management and Systems Engineering Department

# Need for this approach

- Manual effort
- Hyper-tuning of parameters
- Computation

# Objective Function

- General objective function

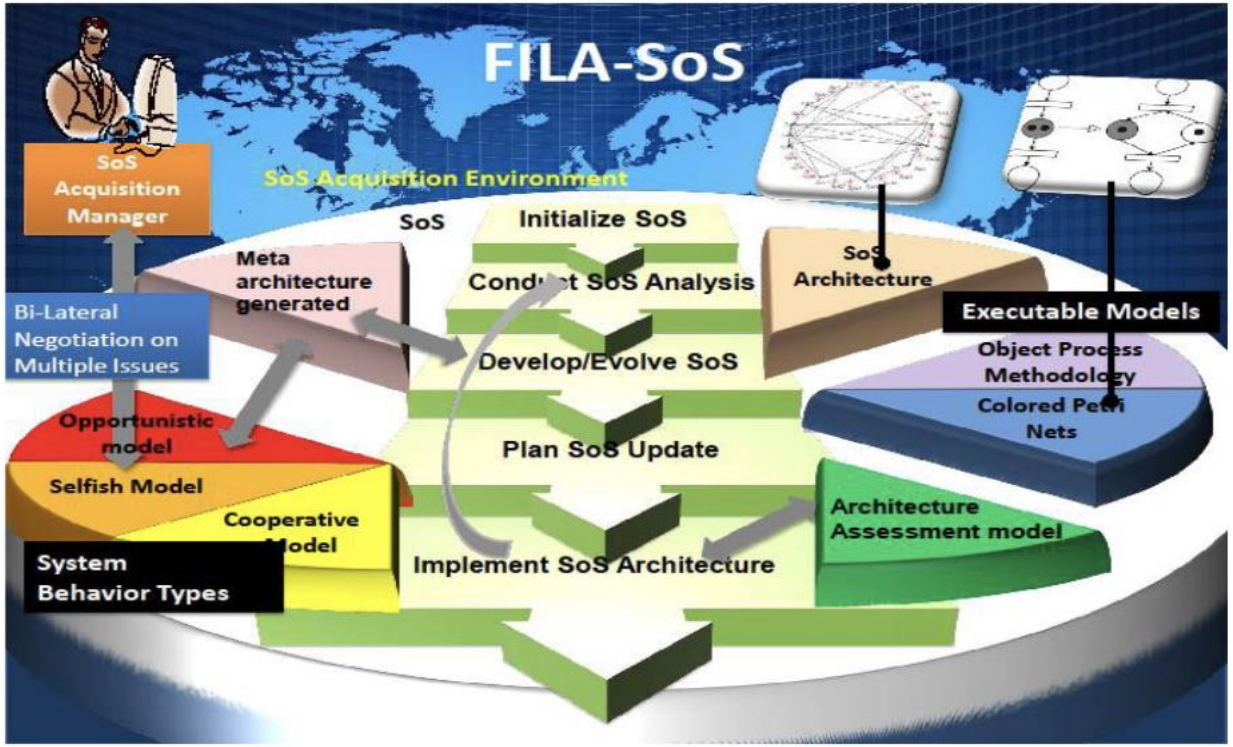
$$\underset{w}{argmin}\ L_{train}(w, \alpha)$$

- Updated objective function

$$\min_{\alpha}\ L_{val}(w^*(\alpha), \alpha)$$

$$s.t.\ \ w^*(\alpha) = argmin_w\ L_{train}(w, \alpha)$$

# Origin


FILA-SoS
SoS Acquisition Manager
SoS Acquisition Environment
SoS
Initialize SoS
Meta architecture generated
Conduct SoS Analysis
SoS Architecture
Executable Models
Bi-Lateral Negotiation on Multiple Issues
Develop/Evolve SoS
Object Process Methodology
Colored Petri Nets
Opportunistic model
Plan SoS Update
Selfish Model
Cooperative Model
Architecture Assessment model
System Behavior Types
Implement SoS Architecture

# Sos Approach

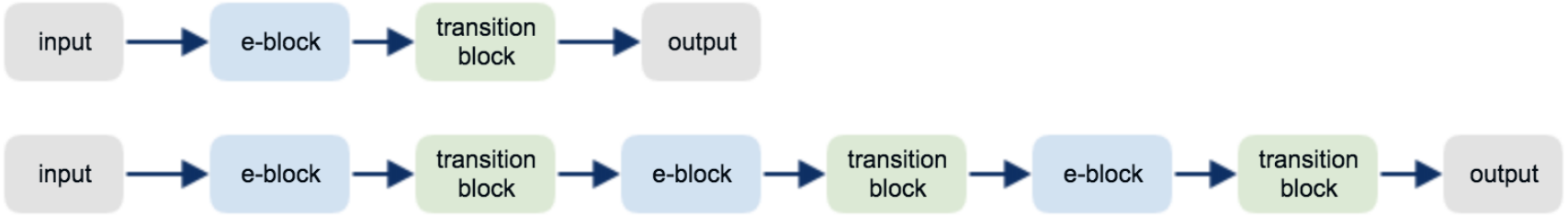
- Choose multiple objectives.
- Choose search space
- Can be used for both CNN and LSTM

# Systems and Capabilities

| System/Layer | Linear transformation | Non-linear transformation | Regularization | Parameter scaling |
|---|---|---|---|---|
| convolutional layer | x | - | - | - |
| pooling layer | x | - | - | - |
| activation layer | - | x | - | - |
| dropout | - | - | x | - |
| batch-normalization | - | - | x | x |

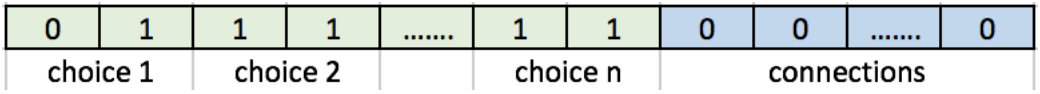
# Pre-determined design

- Reduce search space
- Stack blocks instead of layers



# chromosome

- Select filter size
- Select stride
- Select compression
- Select inputs between $h_0$ and $h_{t-1}$
- Length of chromosome: $n\_total\ d + (d(d+1)/2)$



# Search space

| layer | choices |
|---|---|
| conv | 1,3,5,7 |
| Number of filters | 12,24,36,48 |
| compression | 0.5, 0.65, 0.75, 1 |
| stride | 1,2 |

# Evolution

- Selection
- Crossover
- Mutation
- To prevent training repeated architectures
  - Remove the possibility of generating identical chromosomes both in crossover and mutation by keeping a log of all the chromosomes generated from the beginning.
  - Add randomness to generate few random chromosomes in each generation to prevent the algorithm being struck in local minima.

# Training conditions

- Dataset: CIFAR 10 (60000 labelled images with 10 classes)
- Optimization: stochastic gradient descent (SGD)
- learning rate: 0.1
- batch size: 64
- Total duration: 1 month
- No. of architectures searched: approx 1000

# Limitations

- Size of each e-block is pre-determined
- Different size e-block requires re-training
- Computationally expensive

# results

| Method | Depth | Params (M) | CIFAR-10 error percentage |
|---|---|---|---|
| Network in Network [6] | - | - | 8.81 |
| All CNN [7] | - | - | 7.25 |
| Deeply Supervised Net [5] | - | - | 7.97 |
| Fractal Net [4] | 21 | 38.6 | 5.22 |
| ResNet [10] | 110 | 1.7 | 6.61 |
| Wide ResNet [8] | 16 | 11 | 4.81 |
| ResNet (pre-activation) [2] | 164 | 1.7 | 5.46 |
| DenseNet [3] | 100 | 0.8 | 4.51 |
| e-block@1 | 100 | 1.2 | 5.23 |
| e-block@2 | 100 | 2.8 | 4.61 |