Neural Architecture Search

Sara Boyd, Lauren Gillespie, Elyssa Sliheet

Motivation

- DNN meta-architecture drives network success
- Hand-designed architectures are successful in certain domains [5, 6]
- How to find good architecture for arbitrary domain?
- Enter **neural architecture search (NAS)**
- NAS: finding a good network architecture automatically [7]
    - Semantics, sometimes includes hyperparameter search, sometimes separate
- Our goal: NAS using a simple EA

DNN- Design Choices

1. Number of layers
2. Type - convolutional, pooling, fully connected
3. The ordering of layers
4. Hyperparameters for each type of layer
    a. Receptive field size
    b. Stride
    c. Number of receptive fields for convolution layer
    d. Activation function
    e. Dropout rate

The number of possible choices makes the design space of CNN architectures extremely large and hence, infeasible for an exhaustive manual search.

Design the CNN using Reinforcement Learning [1]

Uses a novel Q-learning agent

Goal: discover CNN architectures that perform well on a given ML task with no human intervention

Agent has a large **space of model architectures** to search from

Learns through random exploration

∈-greedy

Validation accuracy = reward

Design the CNN using Reinforcement Learning [1]

MetaQNN - Q-learning based meta modeling

Three standard image classification datasets: CIFAR-10 [2], MNIST [6], SVHN [3]

Task: train an agent to sequentially choose neural network layers

Results: They compared prediction performance of MetaQNN and other state of the art methods on the three datasets, that “outperform similar models”, and provide “competitive results”.

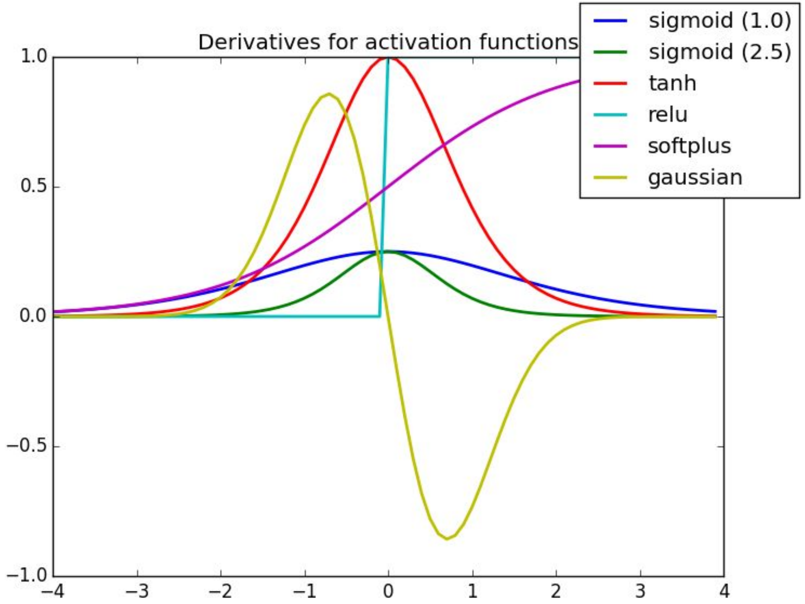
Evolutionary algorithm hyperparameter optimization

- Evolutionary algorithms can do Neural Architecture Search [7,8]
- CoDeepNEAT: NEAT for DNNs - nodes represent whole layers [7]
    - State-of-the-art on Wikipedia Comment Toxicity & Chest X-rays Multitask Image Classification
    - Architecture search better than hyperparameter search
- Simple EA on CIFAR-10 and CIFAR-100 dataset [8]
    - achieved 94.6% and 77% respectively
    - EA with tournament selection
    - Relevant parameters: learning rate, weights, convolutional layer insertion, convolutional layer deletion, stride, channel number, filter size
- Limitations of these systems: computational & time cost

Our Initial Attempt

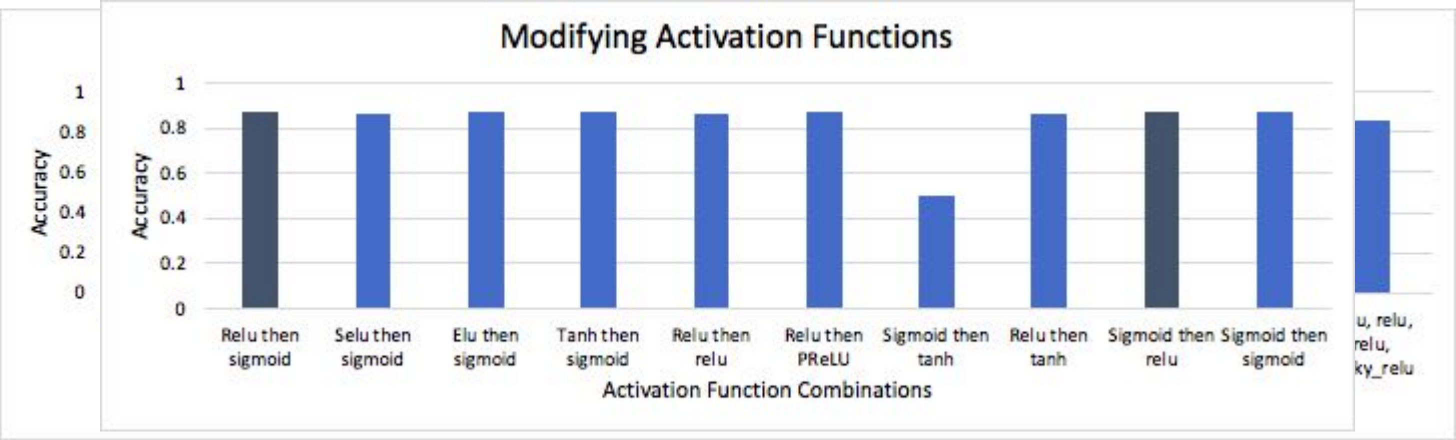
- Manually modified the network for training on the Keras IMDB dataset

- Number of layers
    - Activation function
        - relu, sigmoid, tanh, softmax, elu, selu, softplus, softsign, exponential, linear
- Drawbacks: manual manipulation, architecture already optimized



[4]

Initial Attempt Results




```
model = keras.Sequential()
model.add(keras.layers.Embedding(vocab_size, 16))
model.add(keras.layers.GlobalAveragePooling1D())
model.add(keras.layers.Dense(16, activation=tf.nn.relu))
model.add(keras.layers.Dense(1, activation=tf.nn.sigmoid))
```

Changed Direction

- Develop a program to search for an optimal solution using an n+1 strategy
- Activation functions
  - tanh, softmax, elu, selu, softplus, softsign, relu, sigmoid, hard_sigmoid, exponential, linear
- Network structure - all layers are dense
  - Input
  - Activation (modified parameter)
  - Dropout (modified parameter)
  - Dense layer with num_classes param
  - Activation (modified parameter)
- Child generation
  - Keep the network with the highest accuracy rate (elite population = 1)
  - Fill the rest of the generation with networks that have two random activation functions and a random dropout rate (0-1 range)
- Datasets
  - Reuters Text Categorization Dataset

Results

Epoch: 5, Population size: 5, Number of Generations: 20

REUTERS DATASET	Input Activation	Dropout Rate	Output Activation	Accuracy
Default Architecture	relu	0.5	softmax	0.79
Our Default Architecture	linear	0.0	linear	0.38
Our Architecture	sigmoid	0.08	sigmoid	0.80

Future Work

- Testing with MNIST dataset, but not fully functional
- Include modifying the number of layers (neural architecture search)
- Increase population size and elitism parameter/tournament selection
- Parallelize to use physics department NVIDIA GPUs
- Add more parameters to modify (kernel size, padding, optimizers)
- Modify shape of layers

References

[1] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, November 1998.

[2] Karen Simonyan∗ & Andrew Zisserman. VERY DEEP CONVOLUTIONAL NETWORKS FOR LARGE-SCALE IMAGE RECOGNITION. In *Proceedings of International Conference on Reinforcement Learning 2015.*

[3] Jason Liang, Elliot Meyerson, Babak Hodjat, Dan Fink, Karl Mutch, Risto Miikkulainen. Evolutionary Neural AutoML for Deep Learning. In *Proceedings of Conference on Genetic and Evolutionary Computation 2019.*

[4] Bowen Baker, Otkrist Gupta, Nikhil Naik, Ramesh Raskar. Designing Neural Network Architectures using Reinforcement Learning. In *Proceedings of International Conference on Reinforcement Learning 2017.*

[5]Krizhevsky, A & Hinton, G. 2009. *Learning multiple layers of features from tiny images - Tech. Rep. 1.* Computer Science Department. University of Toronto, Toronto Canada.

[6] Yuval Netzer, Tao Wang, Adam Coates, Alessandro Bissacco, Bo Wu, Andrew Y. Ng. 2011. Reading Digits in Natural Images with Unsupervised Feature Learning. In *NIPS Workshop on Deep Learning and Unsupervised Feature Learning 2011*.

[7] Esteban Real, Sherry Moore, Andrew Selle, Saurabh Saxena, Yutaka Leon Suematsu, Jie Tan, Quoc Le, Alex Kurakin. Large-Scale Evolution of Image Classifiers. In *Proceedings of 34th International Conference on Machine Learning 2017.*

[8] Sagar Sharma. Sept. 6, 2017. *Activation Functions in Neural Networks Towards Data Science.* Medium.

Questions?