

ConvLens:

Visualizing Internal Components of Convolutional Neural Networks

Hooman Shariati
Mahdi Ghodsi

INTRODUCTION

What's a ConvNet?

- a Supervised Machine Learning Algorithm
- Used in:
 - Image and Video Recognition
 - Recommender Systems
 - Natural Language Processing

Recognition + Localization

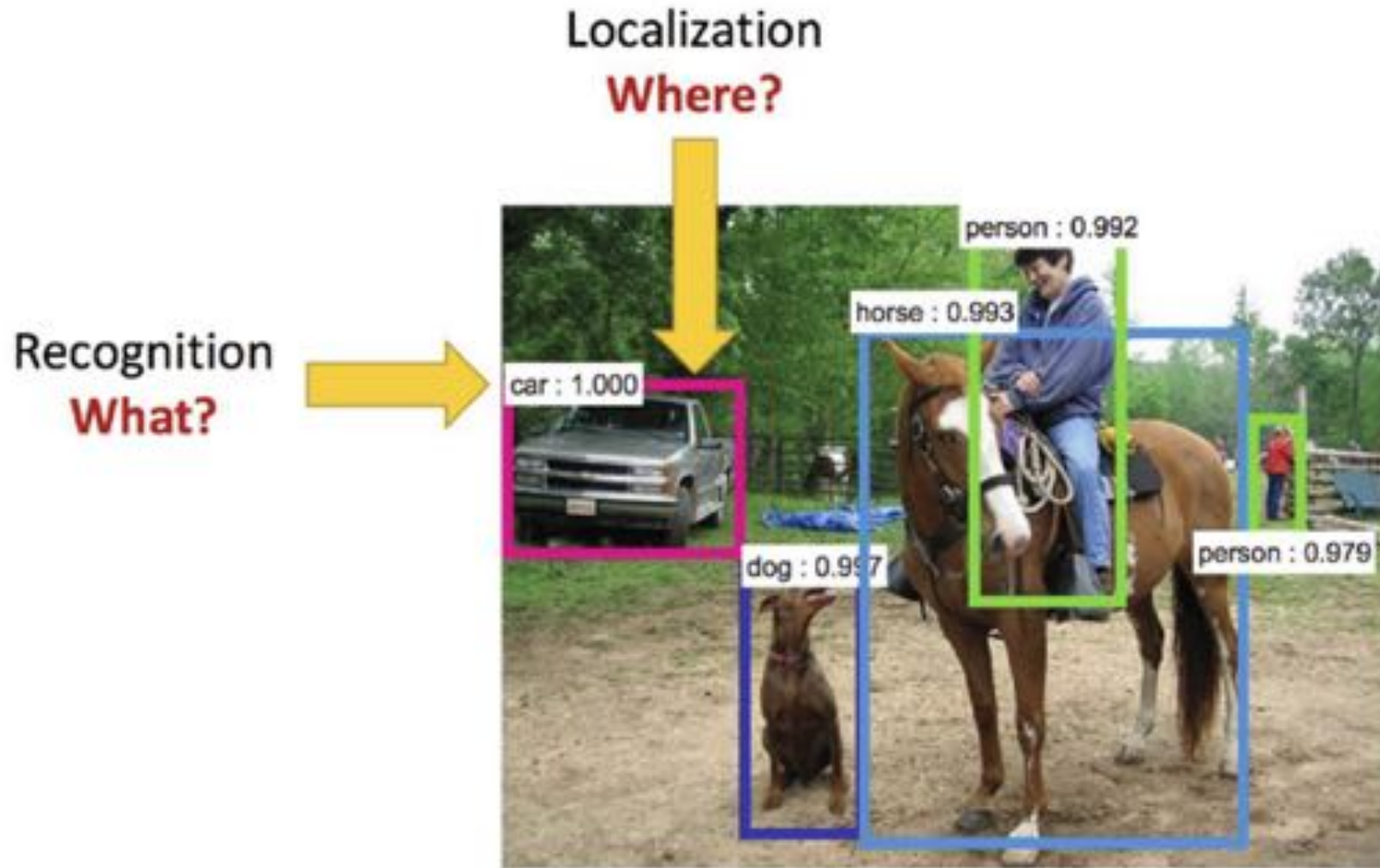


Image Captioning



"man in black shirt is playing guitar."



"construction worker in orange safety vest is working on road."



"two young girls are playing with lego toy."



"boy is doing backflip on wakeboard."



"girl in pink dress is jumping in air."



"black and white dog jumps over bar."



"young girl in pink shirt is swinging on swing."



"man in blue wetsuit is surfing on wave."

Source: Kuarrpaeth y1 anqd F-e i-8Fei, "Deep Visual-Semantic Alignments for Generating Image Descriptions", CVPR 2015

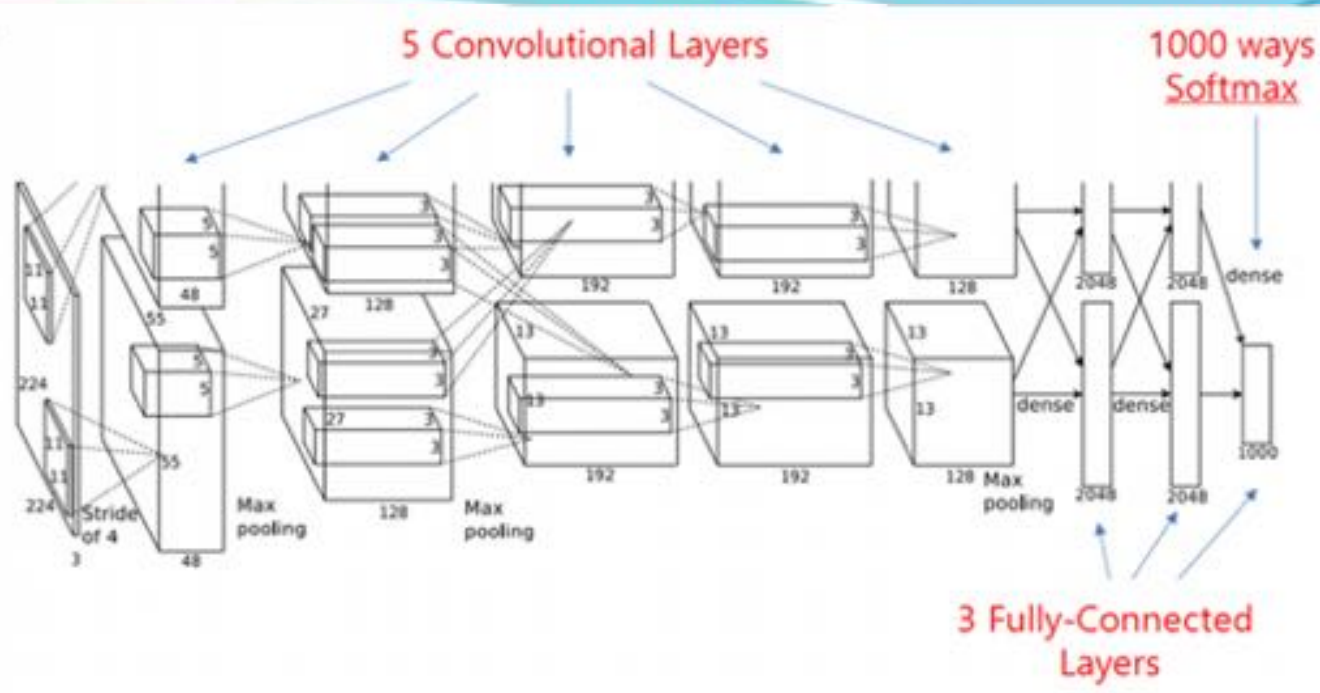
Problem

“Neural networks have long been known as “black boxes” because it is difficult to understand exactly how any particular, trained neural network functions due to **the large number of interacting, non-linear parts.**”

Yajin Zhou



Alexnet



[227x227x3] INPUT

[55x55x96] CONV1: 96 11x11 filters at stride 4, pad 0

[27x27x96] MAX POOL1: 3x3 filters at stride 2

[27x27x96] NORM1: Normalization layer

[27x27x256] CONV2: 256 5x5 filters at stride 1, pad 2

[13x13x256] MAX POOL2: 3x3 filters at stride 2

[13x13x256] NORM2: Normalization layer

[13x13x384] CONV3: 384 3x3 filters at stride 1, pad 1

[13x13x384] CONV4: 384 3x3 filters at stride 1, pad 1

[13x13x256] CONV5: 256 3x3 filters at stride 1, pad 1

[6x6x256] MAX POOL3: 3x3 filters at stride 2

VggNet

ConvNet Configuration					
A	A-LRN	B	C	D	E
11 weight layers	11 weight layers	13 weight layers	16 weight layers	16 weight layers	19 weight layers
input (224 × 224 RGB image)					
conv3-64	conv3-64 LRN	conv3-64 conv3-64	conv3-64 conv3-64	conv3-64 conv3-64	conv3-64 conv3-64
maxpool					
conv3-128	conv3-128	conv3-128 conv3-128	conv3-128 conv3-128	conv3-128 conv3-128	conv3-128 conv3-128
maxpool					
conv3-256 conv3-256	conv3-256 conv3-256	conv3-256 conv3-256	conv3-256 conv3-256 conv1-256	conv3-256 conv3-256 conv3-256	conv3-256 conv3-256 conv3-256
maxpool					
conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512 conv1-512	conv3-512 conv3-512 conv3-512	conv3-512 conv3-512 conv3-512 conv3-512
maxpool					
conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512 conv1-512	conv3-512 conv3-512 conv3-512	conv3-512 conv3-512 conv3-512 conv3-512
maxpool					
FC-4096					
FC-4096					
FC-1000					
soft-max					

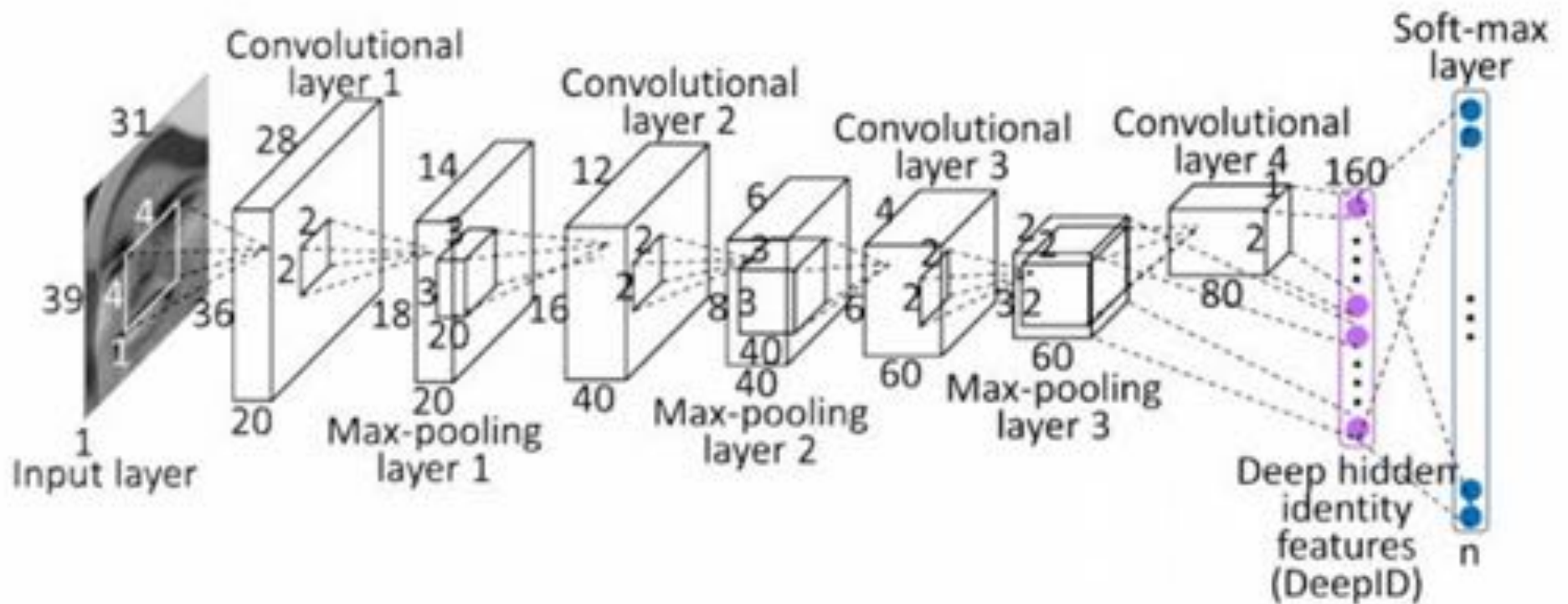
The 6 different architectures of VGG Net. Configuration D produced the best results

To Optimize a ConvNet:

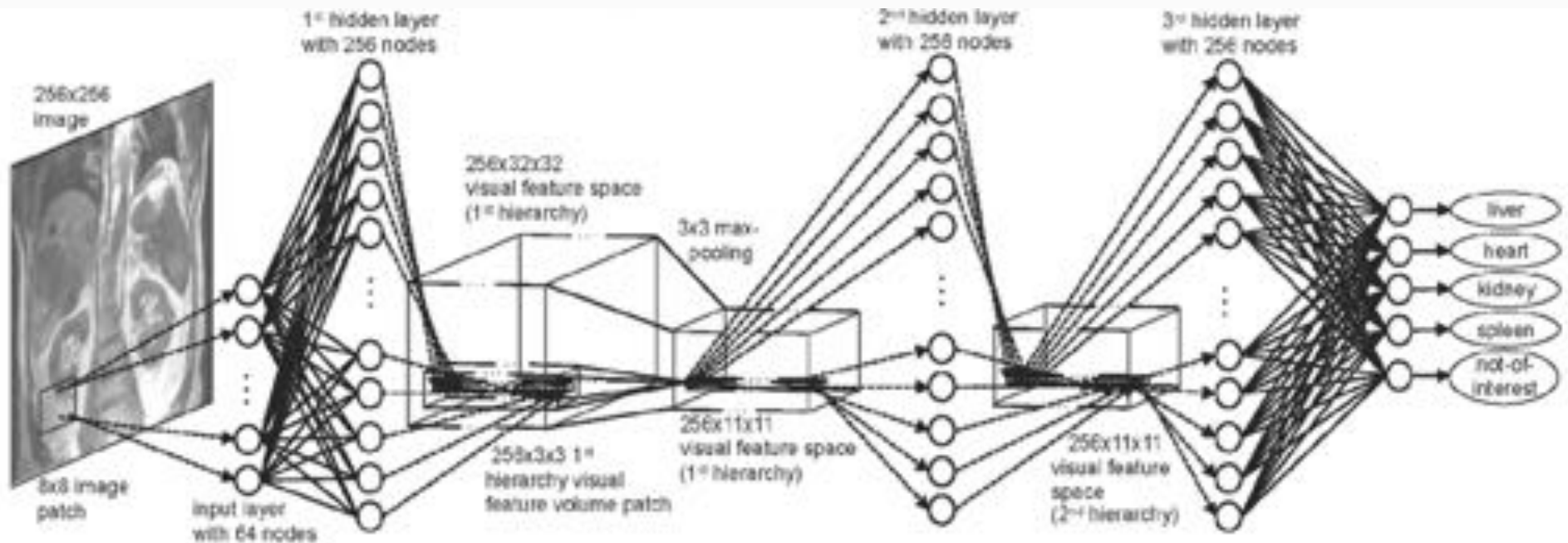
1. Need to understand its structure
2. Need to know what it has learned given its training

Existing Visualizations

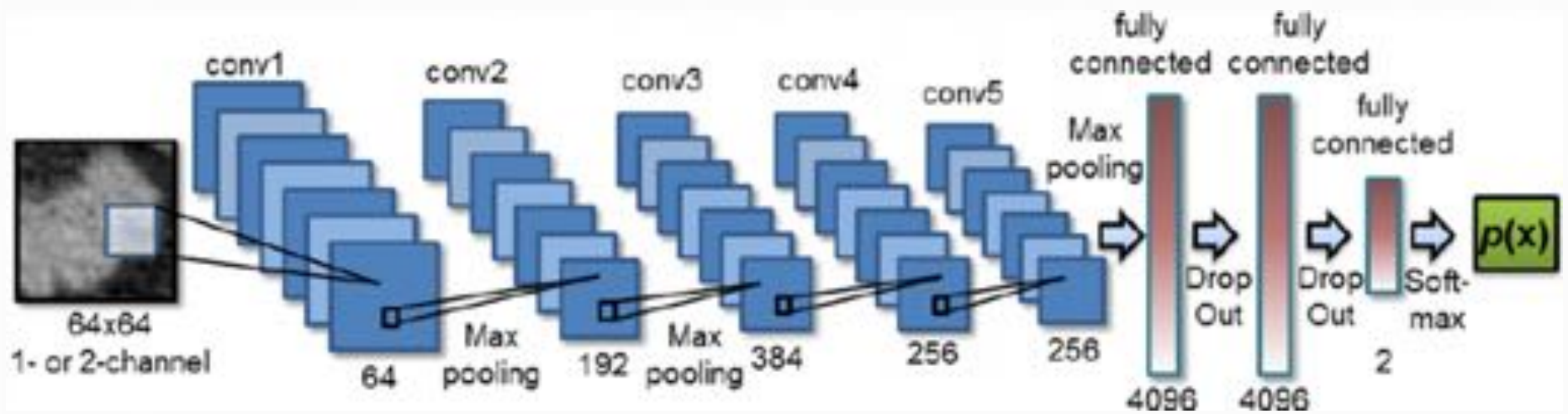
Structure: Mental Model



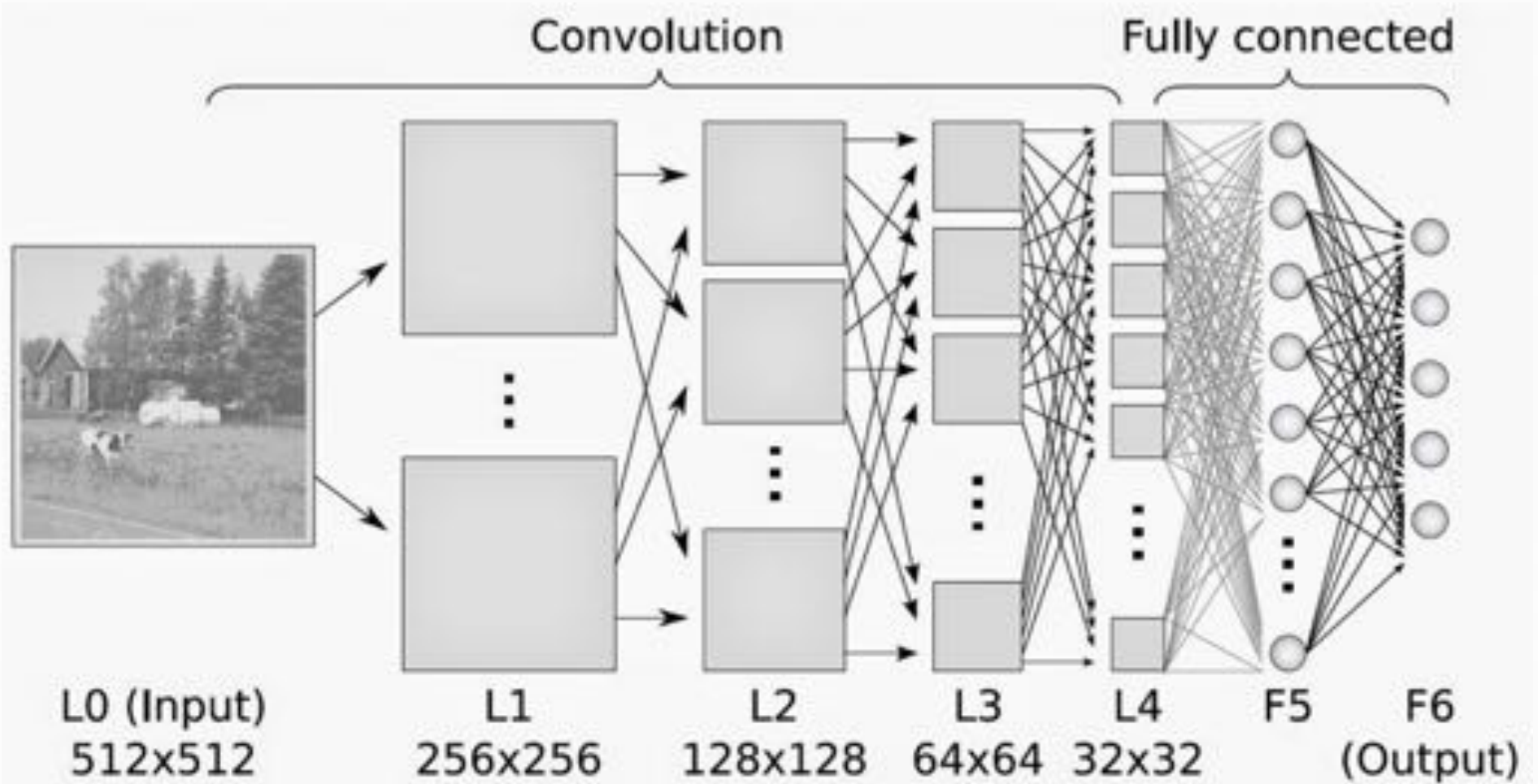
Structure: Too Much



Structure: Too Little



Structure: Seems Right



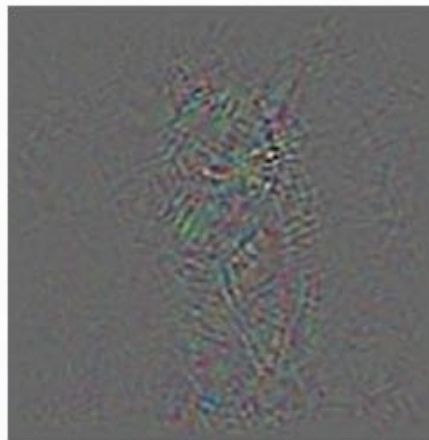


Learning: Forward Activation

Learning: Guided Back Propagation



Input



BackPropagation

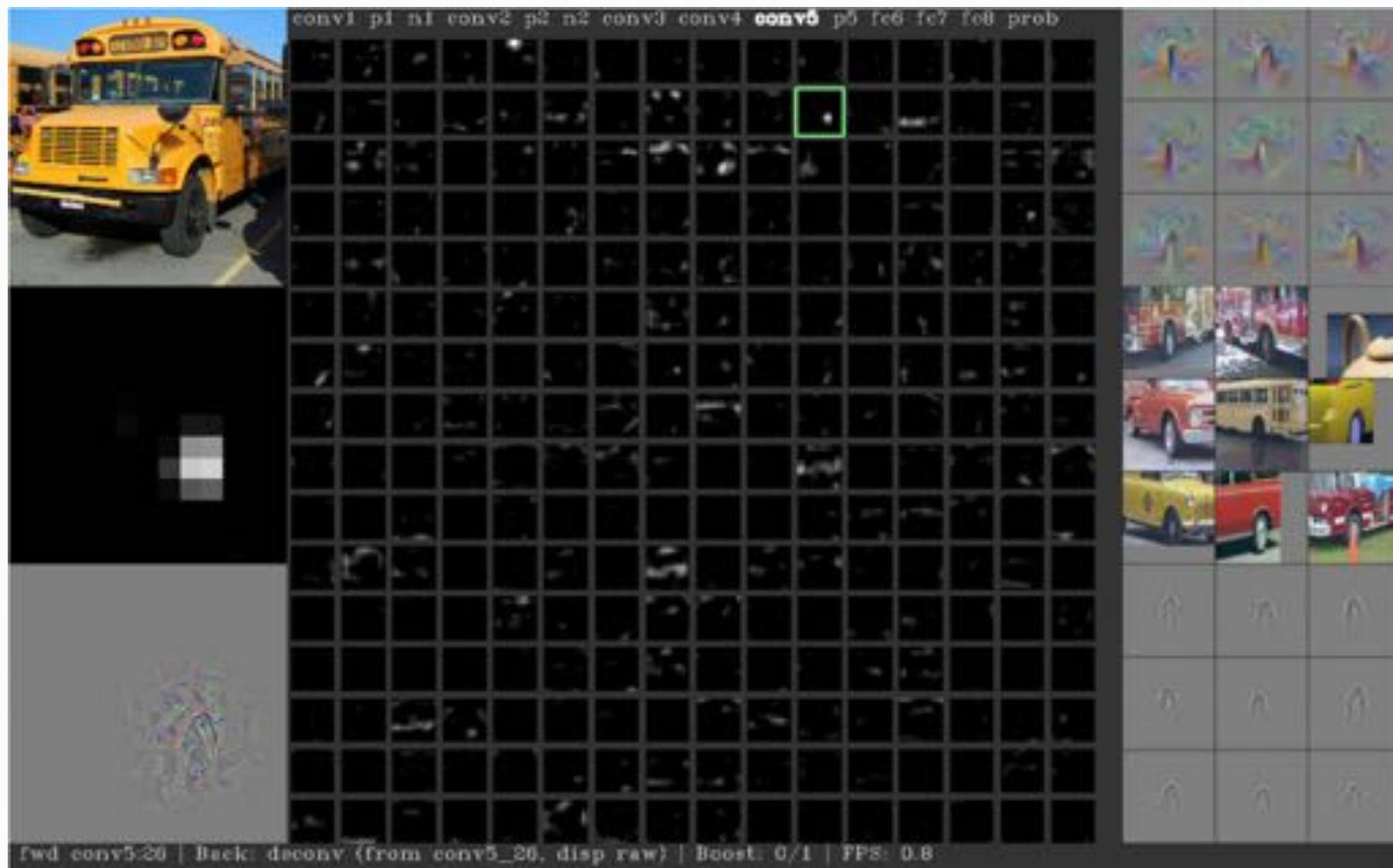


Guided BackProp

Existing Tools: TensorBoard

The screenshot displays the TensorBoard web interface. At the top, there is an orange header with the 'TensorBoard' logo on the left and navigation links for 'EVENTS', 'IMAGES', 'GRAPH', and 'HISTOGRAMS' on the right. On the left side, there is a control panel with options for 'Fit to screen', 'Run' (with a dropdown menu), 'Upload' (with a 'Choose file' button), and 'Color' (set to 'Structure'). Below these are 'Graph' settings, including a legend for node types: 'Non-quant', 'Optimize', 'Unconnected series', 'Connected series', 'Constant', 'Summary', 'Dataflow edge', 'Control dependency edge', and 'Reference edge'. The main area is divided into two sections: 'Main Graph' and 'Auxiliary nodes'. The 'Main Graph' shows a complex network of nodes and edges, with nodes labeled with operations like 'stack', 'stack_1', 'stack_2', 'stack_3', 'stack_4', 'stack_5', 'stack_6', 'stack_7', 'stack_8', 'stack_9', 'stack_10', 'stack_11', 'stack_12', 'stack_13', 'stack_14', 'stack_15', 'stack_16', 'stack_17', 'stack_18', 'stack_19', 'stack_20', 'stack_21', 'stack_22', 'stack_23', 'stack_24', 'stack_25', 'stack_26', 'stack_27', 'stack_28', 'stack_29', 'stack_30', 'stack_31', 'stack_32', 'stack_33', 'stack_34', 'stack_35', 'stack_36', 'stack_37', 'stack_38', 'stack_39', 'stack_40', 'stack_41', 'stack_42', 'stack_43', 'stack_44', 'stack_45', 'stack_46', 'stack_47', 'stack_48', 'stack_49', 'stack_50', 'stack_51', 'stack_52', 'stack_53', 'stack_54', 'stack_55', 'stack_56', 'stack_57', 'stack_58', 'stack_59', 'stack_60', 'stack_61', 'stack_62', 'stack_63', 'stack_64', 'stack_65', 'stack_66', 'stack_67', 'stack_68', 'stack_69', 'stack_70', 'stack_71', 'stack_72', 'stack_73', 'stack_74', 'stack_75', 'stack_76', 'stack_77', 'stack_78', 'stack_79', 'stack_80', 'stack_81', 'stack_82', 'stack_83', 'stack_84', 'stack_85', 'stack_86', 'stack_87', 'stack_88', 'stack_89', 'stack_90', 'stack_91', 'stack_92', 'stack_93', 'stack_94', 'stack_95', 'stack_96', 'stack_97', 'stack_98', 'stack_99', 'stack_100'. The 'Auxiliary nodes' section shows a smaller, simplified graph with nodes labeled 'stack_101', 'stack_102', 'stack_103', 'stack_104', 'stack_105', 'stack_106', 'stack_107', 'stack_108', 'stack_109', 'stack_110', 'stack_111', 'stack_112', 'stack_113', 'stack_114', 'stack_115', 'stack_116', 'stack_117', 'stack_118', 'stack_119', 'stack_120', 'stack_121', 'stack_122', 'stack_123', 'stack_124', 'stack_125', 'stack_126', 'stack_127', 'stack_128', 'stack_129', 'stack_130', 'stack_131', 'stack_132', 'stack_133', 'stack_134', 'stack_135', 'stack_136', 'stack_137', 'stack_138', 'stack_139', 'stack_140', 'stack_141', 'stack_142', 'stack_143', 'stack_144', 'stack_145', 'stack_146', 'stack_147', 'stack_148', 'stack_149', 'stack_150'. The interface is clean and professional, with a clear focus on the computational graph.

Existing Tools: VisToolBox



Existing Tools: Harley's



Existing Tools: CNNVis

CNNVis

Towards Better Analysis of Deep Convolutional Neural Networks
By Visual Analytics Group of Tsinghua University

Model

CIFAR10_BaseCNN

Category

RootCategory

Edge opacity

1.0

Edge color

weight

Edge Size

0.0

View Layer

learning features

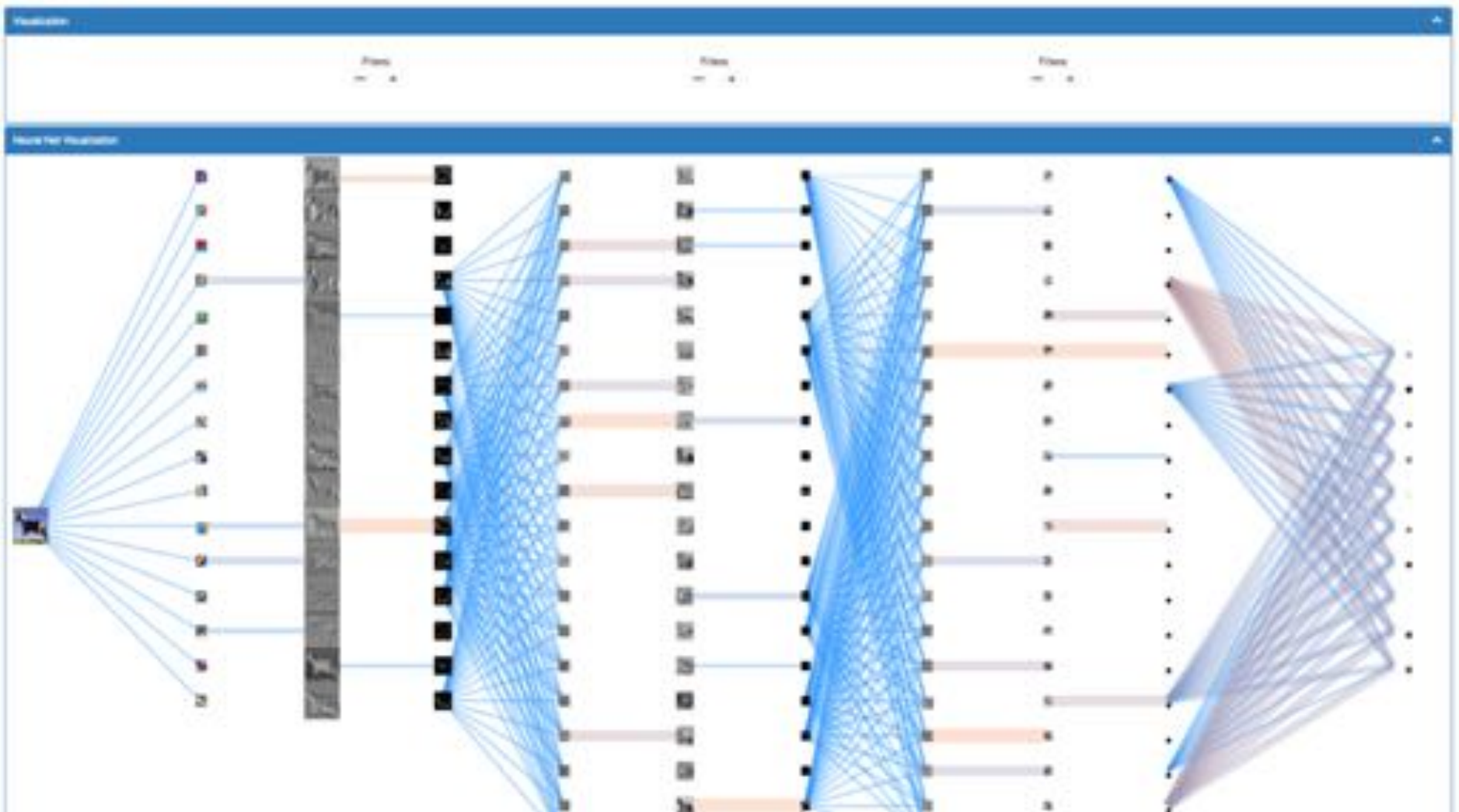
Print



Existing Tools: ReVACNN

ReVACNN

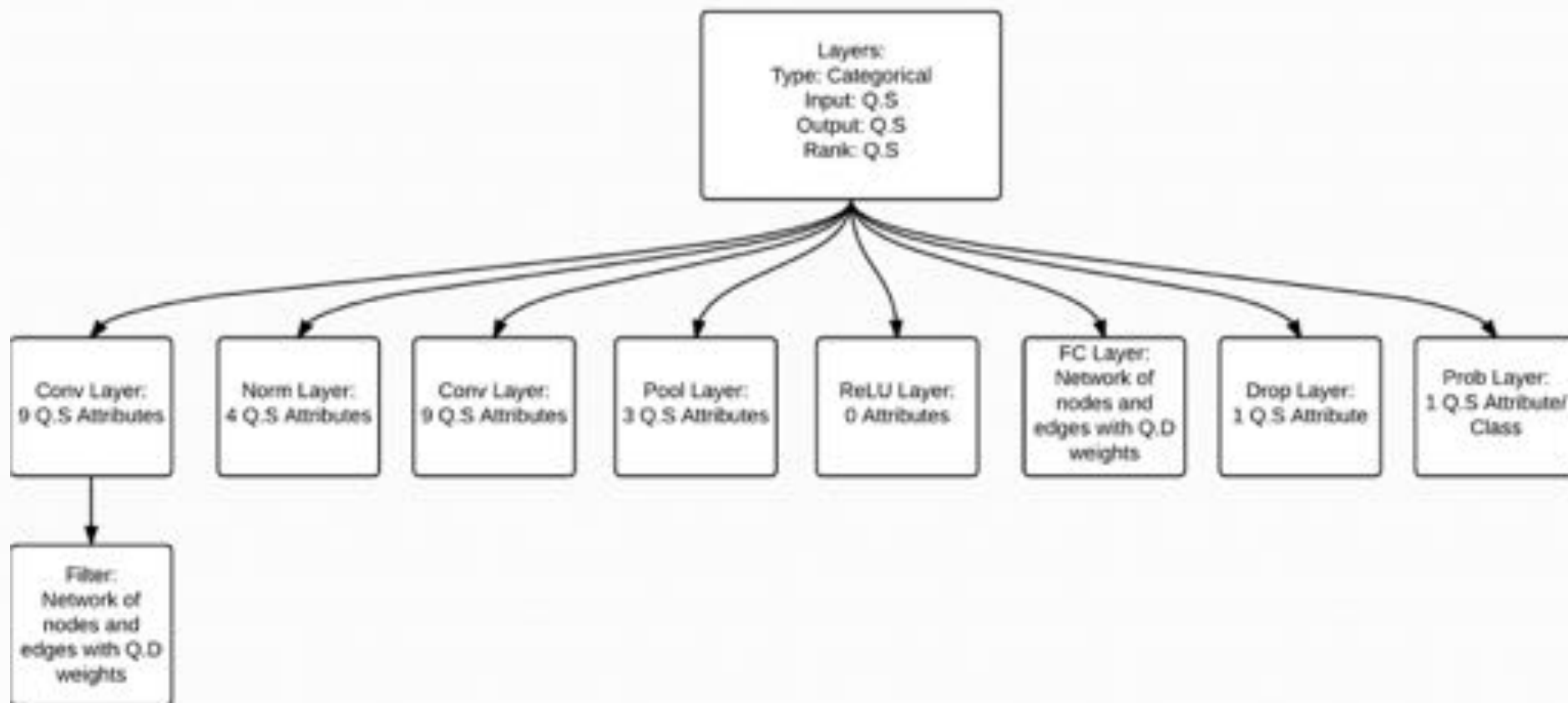
Network Training State: 1-000 70%



Our Design Process

What: Data

Hierarchichal Network



What: Derived

- **For each Filter:**
 - **Activation Images: Diverging**
 - **Guided Back Propagation: Sequential**

Why: Tasks

- **Explore → Summarize**
 - Over all Architecture of ConvNet
 - Parameters in each layer
- **Locate → Identify**
 - Filters that have learned useful features
 - Filters that are useless

Demo

How: Encode

- Layers: Nodes in a chain
- Rank: Position in the chain, text label on the node
- Output size: Text Label on chain links
- All other attributes: Extra information on click
- Filter number (inside a conv layer): Text label and position in the stack of images.

How: Facet

- Juxtaposed and Coordinate Side-by-Side Views
- Shared data (Magnified, and related content)

How: Manipulate

- Select
 - a layer to show more details for
 - a specific filter to show more details for
 - A filter visualization method to be used for the overview of filters
- Annotate useful/ useless filters

How: Aggregate

- Aggregation: Grouped all the nodes with the same depth into a single layer
- Aggregation: Grouped the information contained in all nodes and edges of each filter into a single image
- Filtering: Eliminated low probability outcomes from Prob Layers.

How: Reduce

- Our Guided BackPropagation reduces the data by eliminating negative weights

How: Scale

- Layers: Hundreds
 - No limit on the complexity of layers.
- Filters: Hundreds
 - No limit on the complexity of the NN inside filters.

Future Works & Limitations

- More dynamic front-end for screens with different aspect ratios
- Display the filter visualization images in a sorted order according to the importance of the filter.
 - Importance is implied by the sizes of weights in each filter.
- Find a meaningful visualization for the Fully Connected layers.
 - Deep Dream is one possible solution, but it only shows what the network has learned during training.
- Integrate the data generation tools (scripts) into ConvLens.
 - This requires heavy computation on the server side.