PyData

London 2017

Analyzing 3D objects with power of Deep Learning and Cython

Alexandr Notchenko
avnotchenko@gmail.com

RUSSIAN ACADEMY OF SCIENCES

INSTITUTE FOR INFORMATION TRANSMISSION PROBLEMS (Kharkevich Institute)

Skoltech

Skolkovo Institute of Science and Technology

Open Data Science

3D Shape representations

- Meshes
- Point clouds
- Implicit surfaces / potentials
- Voxels
- Set of 2D projections

3D Shape representations

- Meshes
- Point clouds
- Implicit surfaces / potentials

Irregular size, not clear how to use in NN

- Voxels
- Set of 2D projections

Regular size, good to go in CNN

3D Shape representations

- Meshes
- Point clouds
- Implicit surfaces / potentials

Irregular size, not clear how to use in NN

- Voxels → Regular size, good to go in CNN
- Set of 2D projections → Not really 3D, self occlusion

Sparsity of voxel representation

* Mean sparsity for all classes of ModelNet40 train dataset at voxel resolution 40 equal to 5.5%.

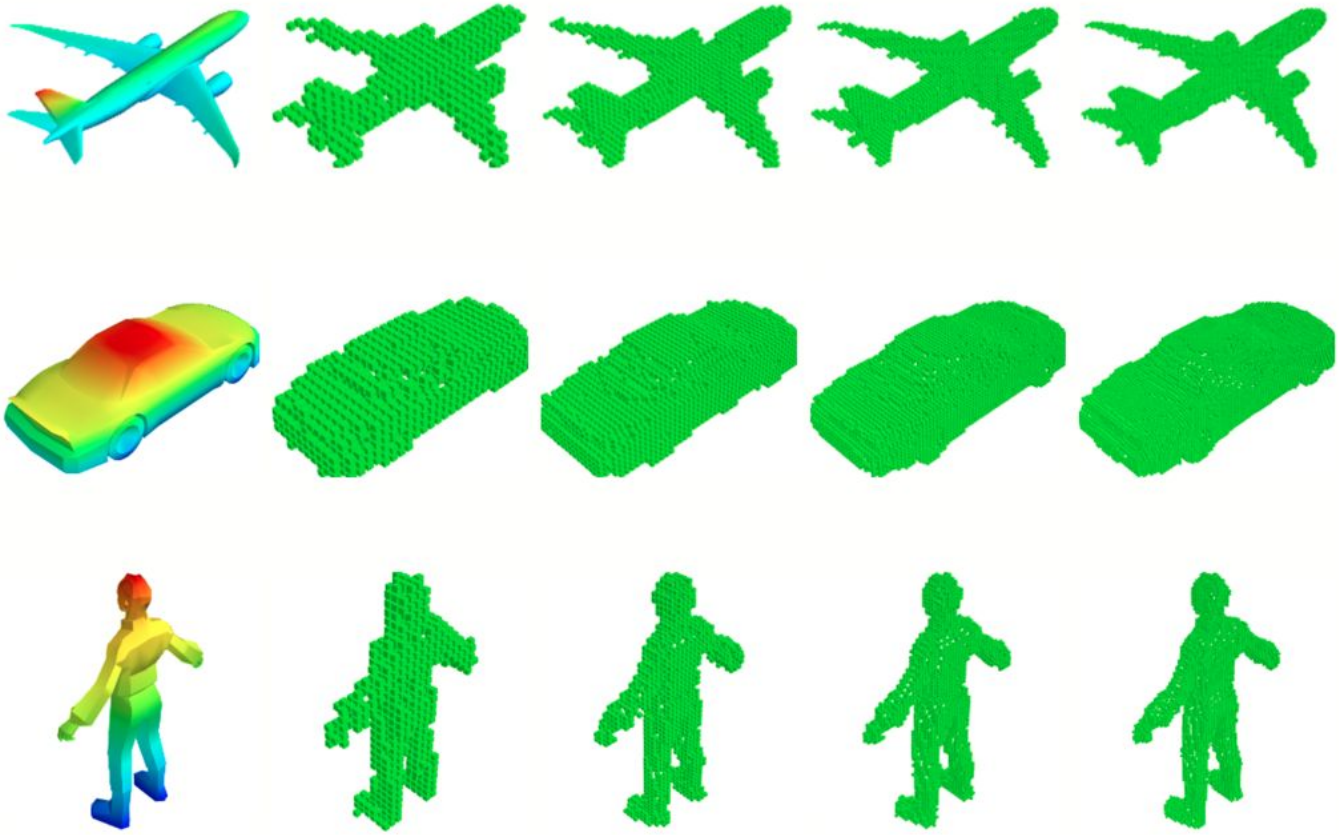
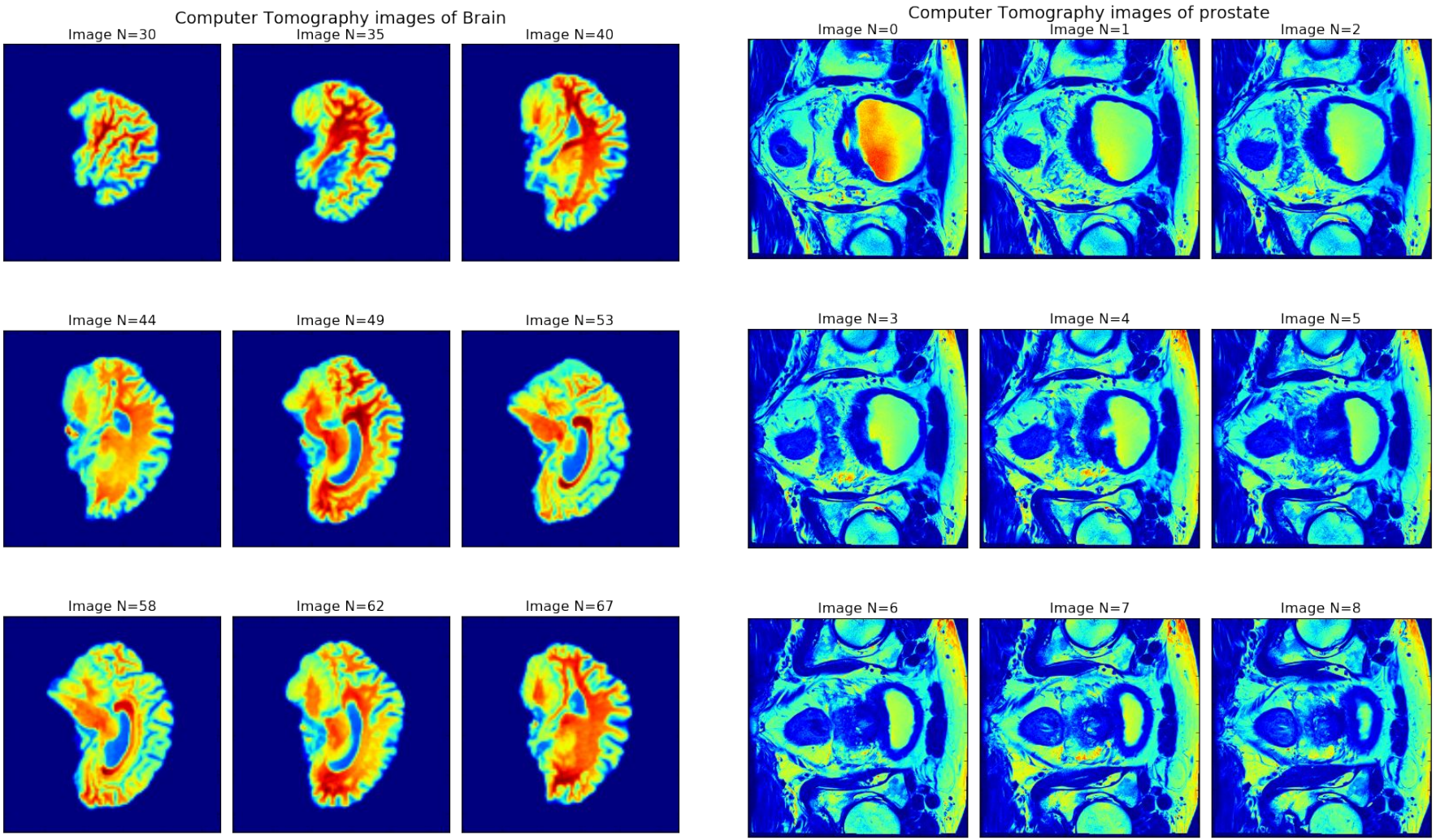


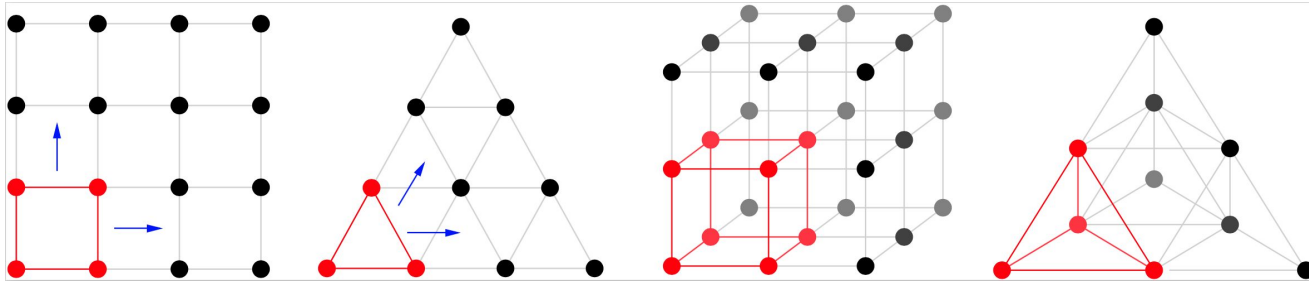
Figure: Examples of some objects voxelizations at different resolutions **30**, **50**, **70**, **100** (from left to right), left-most objects are depicted using original meshes

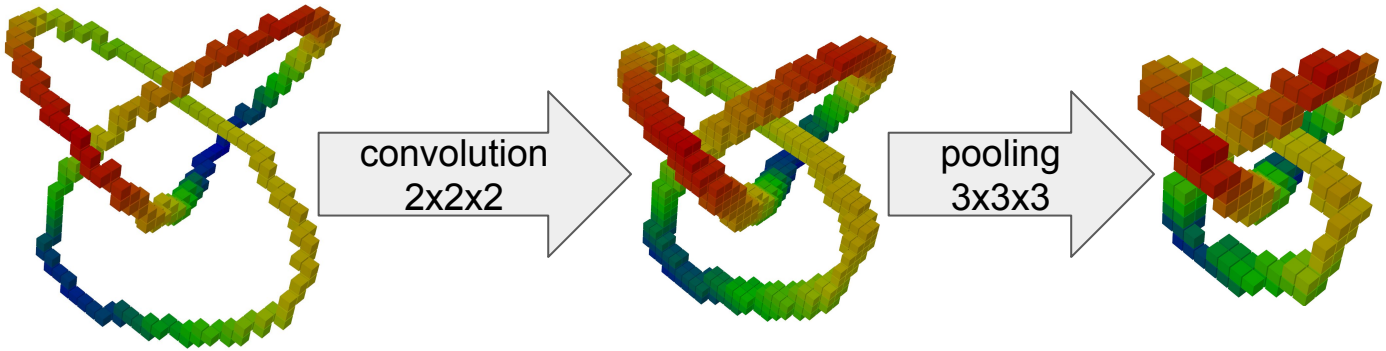


Slices of MRI and CT Images with resolutions 110x110x110 (left), 384x384x19 (right)

SparseConvNet

Dr. Benjamin Graham
assoc. prof. at Warwick University
Facebook AI Research, Paris Lab





http://www2.warwick.ac.uk/fac/sci/statistics/staff/academic-research/graham/bmvc.pdf

gangiman / PySparseConvNet

Unwatch 5 | Star 8 | Fork 2

Code | Issues 0 | Pull requests 0 | Projects 0 | Wiki | Pulse | Graphs | Settings

Python Framework for sparse neural networks

Edit

Add topics

200 commits | 10 branches | 0 releases | 5 contributors

Branch: master | New pull request | Create new file | Upload files | Find file | Clone or download

gangiman * Moved tests to their own directory ... Latest commit a07b4ad 9 days ago

PySCNutils	* Moved tests to their own directory	9 days ago
SparseConvNet	* added some function for Voxel Pictures like	13 days ago
tests	* Moved tests to their own directory	9 days ago
.gitignore	Added a little bit of docstrings	a year ago
Makefile	* Moved tests to their own directory	9 days ago
README.md	Updated README, added reqs and Installation instruction	4 months ago
_SparseConvNet.pxd	* PyVoxelPicture - takes sparse matrix as input	20 days ago
requirements.txt	* added multi-class classification routine	23 days ago
setup.py	* PyVoxelPicture - takes sparse matrix as input	20 days ago
sparseNetwork.pyx	* added some function for Voxel Pictures like	13 days ago

PySparseConvNet

Pros	Cons
C++ / CUDA kernels	Not a general purpose Deep Learning Framework
Effective Memory usage	Complicated code base
Can use any loss functions	Non-standard loss functions are in python-land, overhead memory transfer from GPU (for now)
Can access internal layer activations	
Interactivity	
Train on Sparse data, infer with Dense network	

```
class SparseConvNetCUDA {
public:
  std::vector<SpatiallySparseLayer *> layers;
  std::vector<float> inputNormalizingConstants;
  int dimension;
  int nClasses;
  int nTop;
  int inputSpatialSize;
  int nInputFeatures;
  int nOutputFeatures;
  int deviceID;
  int nBatchProducerThreads;
  cudaMemStream memStream;
  cudaMemStream batchMemStreams[N_MAX_BATCH_PRODUCER_THREADS];
  std::mutex batchLock[N_MAX_BATCH_PRODUCER_THREADS];
  std::vector<SpatiallySparseBatch> batchPool;
  std::vector<SpatiallySparseBatchSubInterface *> initialSubInterfaces;
  std::vector<SpatiallySparseBatchSubInterface *> sharedSubInterfaces;
  cublasHandle_t cublasHandle;

  SparseConvNetCUDA(int dimension, int nInputFeatures, int nClasses,
                    int pciBusID = -1, int nTop = 1,
                    int nBatchProducerThreads = 1);
  ~SparseConvNetCUDA();
  void processBatch(SpatiallySparseBatch &batch, float learningRate,
                    float momentum, std::ofstream &f, std::ofstream &g);
  activation processBatchForward(SpatiallySparseBatch &batch);
  void processBatchBackward(SpatiallySparseBatch &batch,
                                  float learningRate, float momentum,
                                  const std::vector<float> &dfeatures);
  void processIndexLearnerBatch(SpatiallySparseBatch &batch, float learningRate,
                                float momentum, std::ofstream &f);

  void addLearntLayer(int nFeatures, ActivationFunction activationFn = RELU,
                      float dropout = 0.0f, float alpha = 1.0f);
  void addNetworkInNetworkLayer(int nFeatures,
                                ActivationFunction activationFn = RELU,
                                float dropout = 0.0f);
  void addConvolutionalLayer(int nFeatures, int filterSize, int filterStride,
                             ActivationFunction activationFn = RELU,
                             float dropout = 0.0f, int minActiveInputs = 1,
                             float poolingToFollow = 1.0f);
  void addLeNetLayerMP(int nFeatures, int filterSize, int filterStride,
```

```
cdef extern from "SparseConvNet/SparseConvNetCUDA.h":
    cdef cppclass SparseConvNetCUDA:
        vector[SpatiallySparseLayer*] layers
        int computeInputSpatialSize(int outputSpatialSize)
        vector[vector[float]] predict(SpatiallySparseDataset &dataset)
        vector[activation] layer_activations(SpatiallySparseDataset &dataset)
        activation processBatchForward(SpatiallySparseBatch &batch)
        void processBatchBackward(SpatiallySparseBatch &batch,
                                  float learningRate,
                                  float momentum,
                                  vector[float] dfeatures)
        void addConvolutionalLayer(int nFeatures,
                                   int filterSize,
                                   int filterStride,
                                   ActivationFunction activationFn,
                                   float dropout,
                                   int minActiveInputs,
                                   float poolingToFollow)
```

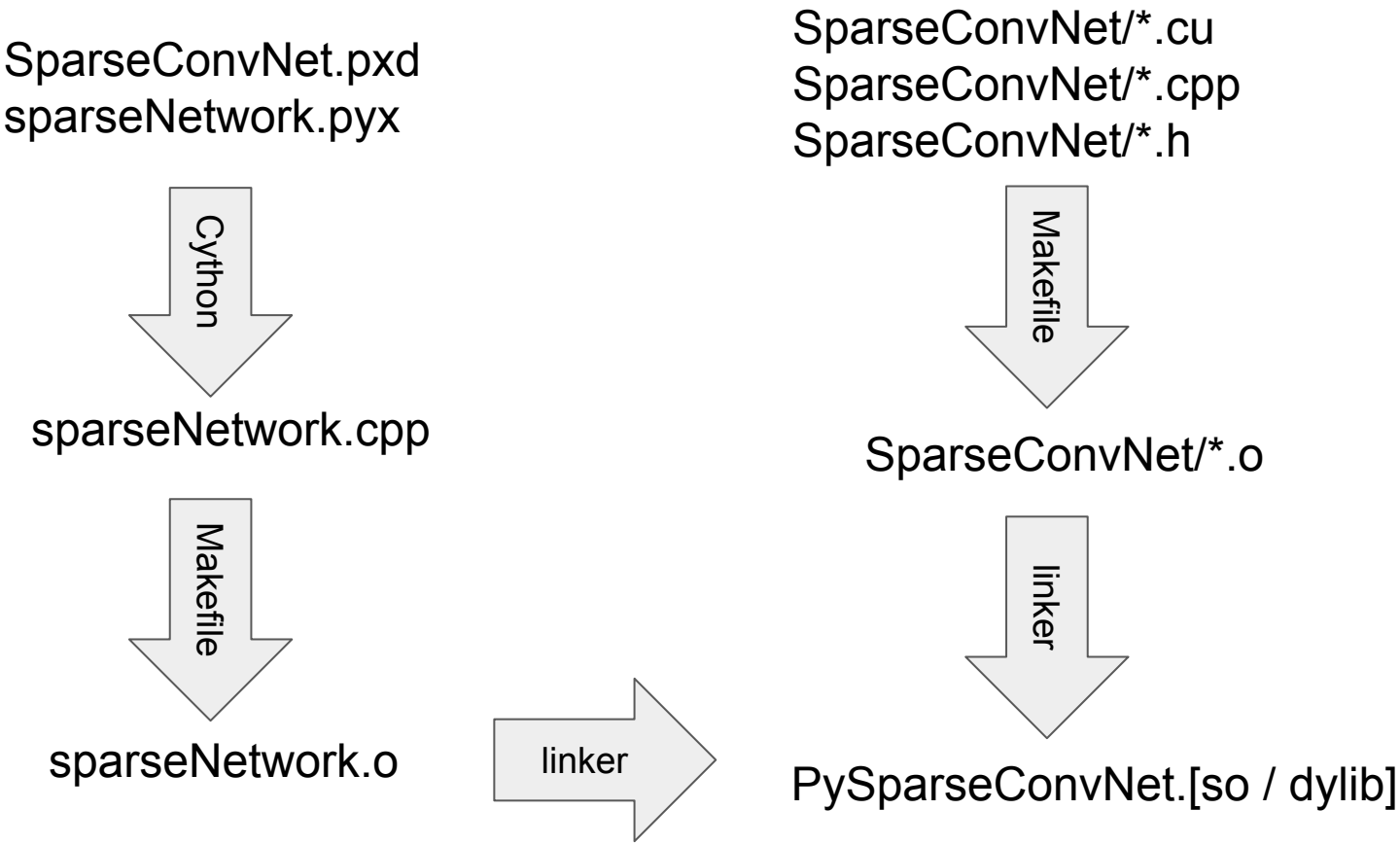
```
cdef class SparseNetwork:
    """create a network object, configure layers, threads and
     dimensionality of input
    """
    cdef SparseConvNet* net
    cdef list layers
    cdef int dimension
    cdef int nInputFeatures
    cdef int nClasses
    cdef int input_spatial_size

    def __cinit__(self, int dimension, int nInputFeatures, int nClasses,
                  int cudaDevice=-1, int nTop=1, int nThreads=1):
        """Initializing Network.

        dimension - number of input dimension
        nInputFeatures number of features in one cell of the grid
        """
        self.layers = []
        self.net = new SparseConvNet(dimension, nInputFeatures,
                                     nClasses, cudaDevice, nTop, nThreads)
        self.dimension = dimension
        self.nInputFeatures = nInputFeatures
        self.nClasses = nClasses
        self.input_spatial_size = -1

    def __dealloc__(self):
        del self.net
```

Cython


SparseConvNet.pxd
sparseNetwork.pyx
Cython
sparseNetwork.cpp
Makefile
sparseNetwork.o
linker
SparseConvNet/*.cu
SparseConvNet/*.cpp
SparseConvNet/*.h
Makefile
SparseConvNet/*.o
linker
PySparseConvNet.[so / dylib]

```
cdef class PyVoxelPicture:
    """wraps VoxelPicture
    """
    cdef VoxelPicture* pic
    cdef SparseGrid grid
    cdef vector[float] features
    cdef int nSpatialSites

    def __cinit__(self, np.ndarray[long, mode="c", ndim=2] indices,
                  np.ndarray[double, mode="c", ndim=2] input_features,
                        int renderSize, int label=-1, int n_features=1):

        self.nSpatialSites = 0
        self.pic = new VoxelPicture(indices, input_features, renderSize,
                                    label, n_features)


    def __dealloc__(self):
        del self.pic
        del self.grid
        del self.features

```

```
class TestVoxelPicture(unittest.TestCase):

    def test_constructor_from_row_matrix(self):
        # indices - an array of shape (num_points, 3),
        #   its columns are indices x,y,z
        indices = np.array([
            [0, 0, 0],
            [1, 0, 5],
            [3, 4, 2],
            [5, 5, 5]
        ], dtype=np.int)
        # size of 3-d tensor, all sides are equal
        spatial_size = 6
        n_features = 1
        # features of size (num_points, num_features)
        # in this case num_features=1
        features = np.ones((indices.shape[0], 1), dtype=np.float)
        # creating a picture object
        pic = pscn.PyVoxelPicture(indices, features, spatial_size)
        # extracting indices and features must be the same
        # as ones it was created from
        returned_indices, returned_features = pic.codifyInputData(spatial_size)
        sparse_indicies, sparse_features = convert_pairs_and_features_to_map(
            returned_indices, returned_features, spatial_size, n_features)
```

Data type	Description
bool_	Boolean (True or False) stored as a byte
int_	Default integer type (same as C `long`; normally either `int64` or `int32`)
intc	Identical to C `int` (normally `int32` or `int64`)
intp	Integer used for indexing (same as C `ssize_t`; normally either `int32` or `int64`)
int8	Byte (-128 to 127)
int16	Integer (-32768 to 32767)
int32	Integer (-2147483648 to 2147483647)
int64	Integer (-9223372036854775808 to 9223372036854775807)
uint8	Unsigned integer (0 to 255)
uint16	Unsigned integer (0 to 65535)
uint32	Unsigned integer (0 to 4294967295)
uint64	Unsigned integer (0 to 18446744073709551615)
float_	Shorthand for `float64`.
float16	Half precision float: sign bit, 5 bits exponent, 10 bits mantissa
float32	Single precision float: sign bit, 8 bits exponent, 23 bits mantissa
float64	Double precision float: sign bit, 11 bits exponent, 52 bits mantissa

C types	From Python types	To Python types
[unsigned] char	int, long	int
[unsigned] short		
int, long		
unsigned int	int, long	long
unsigned long		
[unsigned] long long		
float, double, long double	int, long, float	float
char *	str/bytes	str/bytes [1]
struct		dict

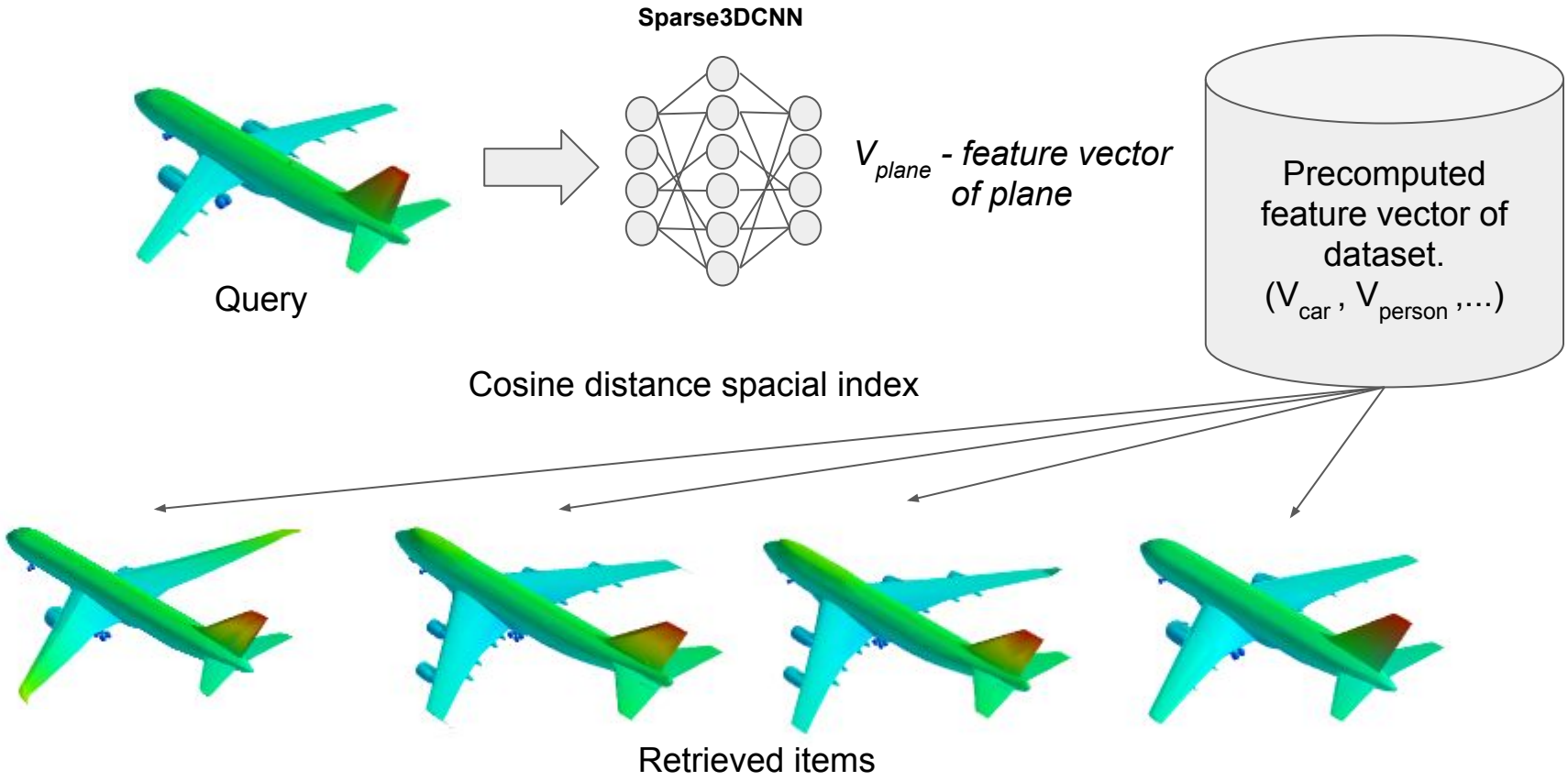
Shape Retrieval

Problem statement

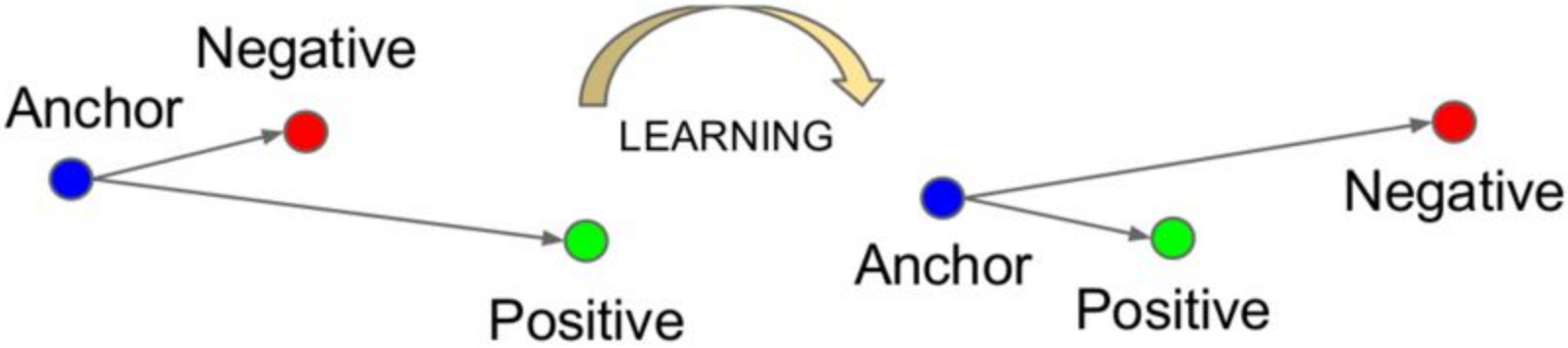
Given a query object find several the most “similar” to the query objects from the given database.

The objects are considered to be similar if they belong to the same category of objects and have similar shapes.

Shape Retrieval



Triplet loss



The representation can be efficiently learned by minimizing triplet loss.

Triplet is a set (***a***, ***p***, ***n***), where

- ***a*** - anchor object
- ***p*** - positive object that is similar to anchor object
- ***n*** - negative object that is not similar to anchor object

$$\lambda(\delta_+, \delta_-) = \max(\mu + \delta_+ - \delta_-)$$

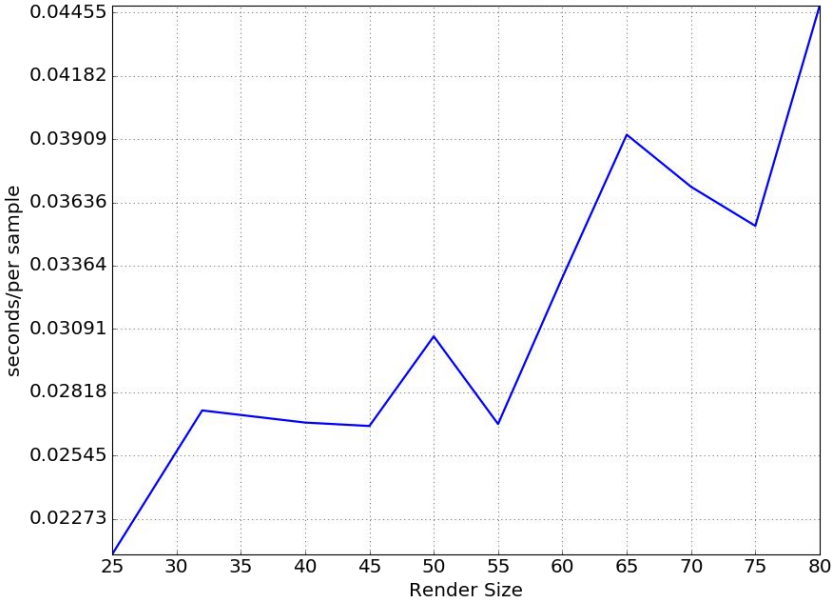
where μ is a margin parameter, δ_+ and δ_- are distances between ***p*** and ***a*** and ***n*** and ***a***.

Network description

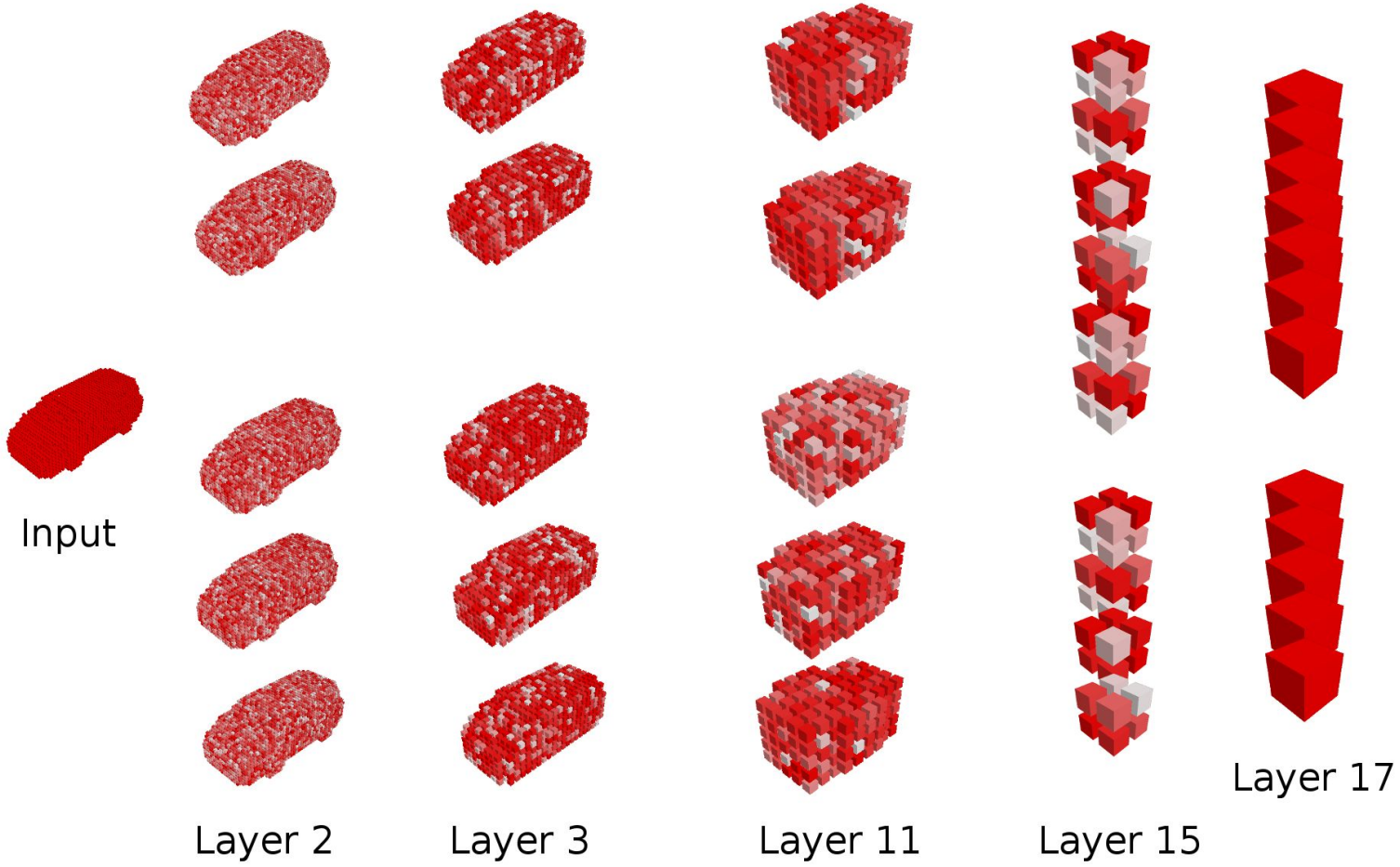
layer type	size	stride	channels	spatial size	sparsity (%) [1]	MRI sparsity [2]
Data input	-	-	1	126	0.18	9.76
Sparse Convolutional Layer	2	1	8	125	-	-
Leaky ReLU ($\alpha = 0.33$)	-	-	32	125	0.35	10.75
Sparse MaxPooling Layer	3	2	32	62	0.69	12.59
Sparse Convolutional Layer	2	1	256	61	-	-
Leaky ReLU ($\alpha = 0.33$)	-	-	64	61	1.07	14.94
Sparse MaxPooling Layer	3	2	64	30	1.93	19.62
Sparse Convolutional Layer	2	1	512	29	-	-
Leaky ReLU ($\alpha = 0.33$)	-	-	96	29	3.26	26.49
Sparse MaxPooling Layer	3	2	96	14	7.32	41.47
Sparse Convolutional Layer	2	1	768	13	-	-
Leaky ReLU ($\alpha = 0.33$)	-	-	128	13	15.14	64.55
Sparse MaxPooling Layer	3	2	128	6	46.30	95.21
Sparse Convolutional Layer	2	1	1024	5	-	-
Leaky ReLU ($\alpha = 0.33$)	-	-	160	5	97.54	100.00
Sparse MaxPooling Layer	3	2	160	2	100.00	100.00
Sparse Convolutional Layer	2	1	1280	1	-	-
Leaky ReLU ($\alpha = 0.33$)	-	-	192	1	100.00	100.00

[1] column "sparsity" is computed for render size = **40**

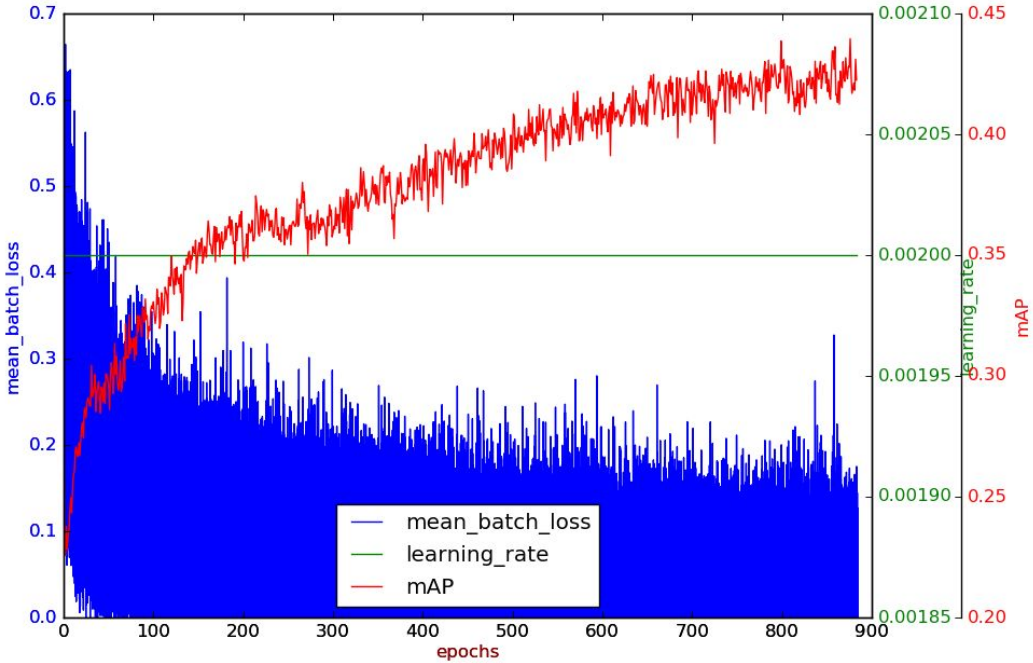
[2] column "MRI sparsity" is computed for render size = **110**

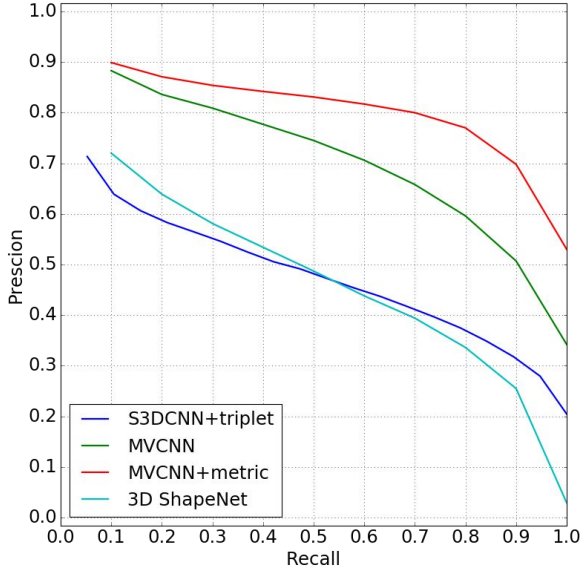


Forward Pass Activations



Experimental results

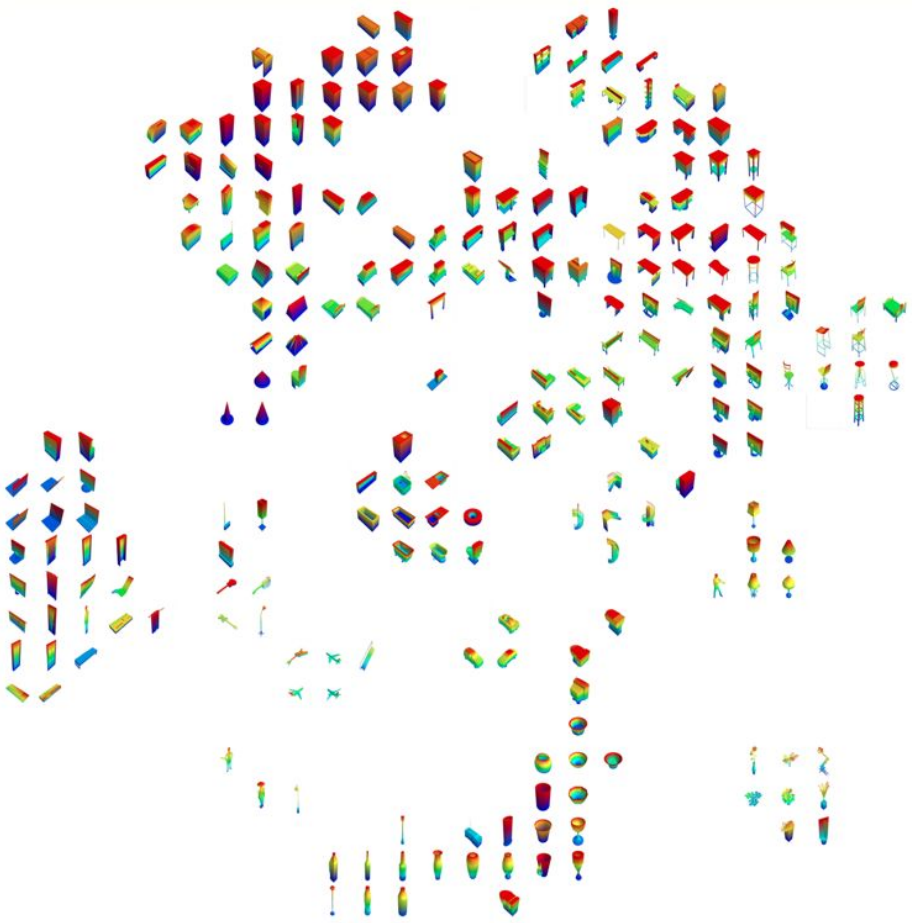




Optimisation algorithm:
Nesterov Accelerated Gradient:
momentum = 0.99
Constant Learning Rate = 0.002

method	Classification	Retrieval mAP
3DShapeNet	77.32%	49.23%
MVCNN	90.10%	**80.20%**
3DSCNN	**90.3%**	45.16%
S3DCNN + triplet	---	46.71%

Obligatory t-SNE



Conclusions

- For Shape Datasets in voxel form - resolution beyond 30^3 doesn’t improves performance very much
- More voxels - change scale of features, probably needs more layers
- 3D CNNs are more efficient on volumetric data

Organizing hyperparameter search

Retrieval_Experiments

File Edit View Insert Format Data Tools Add-ons Help Last edit was on 9 December 2016

avnotchen

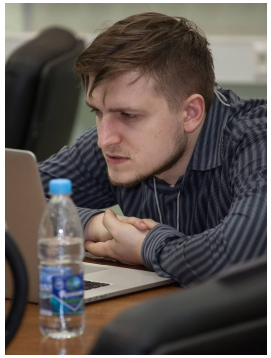
Comments

dataset

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T
1	dataset	net_arch	batch_size	unique_c	test_even	pair_taking_method	lr_decay_rate	render_	in_batch_sam	margin	norm	exp_hash	server	status	current_ep	mAP	last_update		links	total number of
2	ShapeNet55	deepC2	252	6	100	lazy_almost_shuffled_pe	0.005	32	FALSE	0.1	cosine	aaaa111vvvv	burn	stopped	828	0.3963241766	Mon Sep 26 17:57:04 2016		png plot	2.09E+07
3	ModelNet40	deepC2	252	6	100	lazy_almost_shuffled_pe	0.005	32	FALSE	0.1	cosine	adfcndd33nd	labten-01	done					png plot	0.00E+00
4	ModelNet40	deepC2	60	4	100	random_subsampling	0.005	40	FALSE	0.1	cosine	bbbb322adfd	iitp-01	done					png plot	0.00E+00
5	ModelNet40	deepC2	270	6	100	lazy_almost_shuffled_pe	0.005	60	FALSE	0.1	cosine	cccc12515af	titan	done					png plot	0.00E+00
6	ModelNet40	deepC2	180	5	200	random_subsampling	0.005	50	FALSE	0.1	cosine	dddd34252gd	labten-02	stopped	286	0.3312430251	Wed Sep 28 02:16:13 2016		png plot	1.03E+07
7	ModelNet40	deepC2	90	5	200	lazy_almost_shuffled_pe	0.005	70	FALSE	1	L2	fsfwer331vv	titan	stopped	244	0.2998781443	Wed Sep 28 02:10:57 2016		png plot	4.39E+06
8	ModelNet40	deepC2	90	3	200	lazy_almost_shuffled_pe	0.005	32	FALSE	1	cosine	mZLKABODnxl	burn	stopped	510	0.2110905835	Tue Sep 27 16:43:21 2016		png plot	9.18E+06
9	ModelNet40	deepC2	90	6	200	random_subsampling	0.005	45	FALSE	1	L2	mdqY14gfwXu	deepburn	done	639	0.3035397731			png plot	1.15E+07
10	ModelNet40	deepC2	90	6	200	random_subsampling	0.005	60	FALSE	1	L2	aEKclGUNWX4	labten-01	stopped	335	0.3137650743			png plot	6.03E+06
11	ModelNet40	deepC2	90	5	200	random_subsampling	0.005	55	FALSE	0.5	cosine	5ciPkGQBUHE	deepburn	stopped	368	0.2896838771			png plot	6.62E+06
12	ModelNet40	deepC2	120	10	800	random_subsampling	0.002	32	FALSE	1	cosine	HyBxR0QAZkh	burn	stopped	104	0.1979471797	Wed Oct 5 16:36:27 2016		png plot	9.98E+06
13	ModelNet40	deepC2	90	10	800	random_subsampling	0.002	40	FALSE	1	cosine	PaknYB9i3R0	iitp-01	stopped	124	0.2085503059	Sat Oct 1 04:45:20 2016		png plot	8.93E+06
14	ModelNet40	deepC2	72	8	800	random_subsampling	0.002	45	FALSE	1	cosine	0fWOUruVEaz	iitp-03	done	199	0.2303943628	Sun Oct 2 02:51:04 2016		png plot	1.15E+07
15	ModelNet40	deepC2	120	5	400	random_subsampling	0.004	50	FALSE	1	cosine	AEWxBRuSJ4n	burn	stopped	226	0.2542267267	Tue Oct 11 19:54:57 2016		png plot	1.08E+07
16	ModelNet40	deepC2	90	10	800	random_subsampling	0.002	55	FALSE	1	cosine	87ALzZ1QwBK	deepburn	done	159	0.2326888225	Mon Oct 3 13:47:29 2016		png plot	1.14E+07
17	ModelNet40	deepC2	120	10	800	random_subsampling	0.002	60	FALSE	1	cosine	tew16fZSoyl	labten-02	stopped	85	0.2235581123	Sat Oct 1 21:22:33 2016		png plot	8.16E+06
18	ModelNet40	deepC2	120	10	800	random_subsampling	0.002	65	FALSE	1	cosine	EvO6ftwRjSl	labten-01	stopped	58	0.2526285167	Fri Sep 30 13:11:04 2016		png plot	5.57E+06
19	ModelNet40	deepC2	90	10	800	random_subsampling	0.002	70	FALSE	1	cosine	9kCNSE4huYB	deepburn	done	159	0.233684883	Tue Oct 4 02:47:30 2016		png plot	1.14E+07
20	ModelNet40	deepC2	90	10	800	random_subsampling	0.002	75	FALSE	1	cosine	9nhgJRCoO0Q	iitp-04	stopped	78	0.2224471503	Sat Oct 1 01:47:54 2016		png plot	5.62E+06
21	ModelNet40	deepC2	120	10	800	random_subsampling	0.002	80	FALSE	1	cosine	K7IG4U3cJ5V	titan	done	119	0.2205873523	Tue Oct 4 12:01:47 2016		png plot	1.14E+07
22	ModelNet40	deepC2	120	10	800	random_subsampling	0.002	25	FALSE	1	cosine	OclCWJjVLmb	iitp-03	stopped	86	0.1971750831	Tue Oct 4 10:53:00 2016		png plot	8.26E+06
23	ShapeNet55	deepC2	150	10	600	random_subsampling	0.002	32	FALSE	1	cosine	AksKitQfRh8	burn	stopped	20	0.2578580269	Thu Sep 29 15:55:45 2016		png plot	1.80E+06
24	ModelNet40	deepC2_wide	90	5	500	shuffled_permutations	{"lr_step": 5,"lr_base": 0.01,"lr_decay": 0.66,"m_step": 12,"m_base": 0.5,"m_inc": 0.1,"m_max": 0.99}	50	FALSE	2	L2	IlbtmHoQ9UF	titan	stopped	142	0.2389334476	Fri Oct 14 12:28:42 2016		png plot	6.39E+06
25	ModelNet40	deepC2_wide	90	5	500	shuffled_permutations	{"lr_step": 5,"lr_base": 0.01,"lr_decay": 0.66,"m_step": 20,"m_base": 0.7,"m_inc": 0.1,"m_max": 0.99}	50	FALSE	2	L2	pHEKMNeSJ38	titan	stopped	139	0.237851468	Fri Oct 14 12:24:56 2016		png plot	6.26E+06
26	ModelNet40	deepC2_wide	90	5	500	shuffled_permutations	{"lr_step": 5,"lr_base": 0.01,"lr_decay": 0.5,"m_step": 20,"m_base": 0.5,"m_inc": 0.1,"m_max": 0.99}	50	FALSE	2	L2	wFnYuDlxHdZ	burn	stopped	117	0.2495343194	Fri Oct 14 10:52:24 2016		png plot	5.27E+06
27	ModelNet40	deepC2_wide	90	5	500	shuffled_permutations	{"lr_step": 10,"lr_base": 0.003,"lr_decay": 0.5,"m_step": 20,"m_base": 0.5,"m_inc": 0.1,"m_max": 0.99}	50	FALSE	2	L2	cMrR2zgevqW	labten-01	stopped	107	0.2529385175	Fri Oct 14 12:09:14 2016		png plot	4.82E+06

Please contribute:
https://github.com/gangiman/PySparseConvNet
GPLv3

ADISE@Skoltech



Alexandr Notchenko, Ermek Kapushev, Evgeny Burnaev

Special thanks to:

Dmitry Yarotsky

Rasim Akhunzyanov

From OpenDataScience with Love