Evolutionary Computation and Evolutionary Deep Learning for Image Analysis, Signal Processing and Pattern Recognition

Mengjie Zhang[1] and Stefano Cagnoni[2]

1 Evolutionary Computation Research Group, Victoria University of Wellington, Wellington, New Zealand

2 IBIS Lab, University of Parma, Parma, Italy

Mengjie.zhang@ecs.vuw.ac.nz, cagnoni@ce.unipr.it

http://gecco-2019.sigevo.org/

GECCO '19 Companion, July 13–17, 2019, Prague, Czech Republic

ACM ISBN 978-1-4503-6748-6/19/07. ...$15.00

DOI https://doi.org/10.1145/3319619.3323388

1

Instructors

- **Mengjie Zhang** is a Professor of Computer Science at the School of Engineering and Computer Science, Victoria University of Wellington (VUW), New Zealand. His research is mainly focused on evolutionary computation, particularly genetic programming, particle swarm optimization and evolutionary deep learning in image analysis, multi-objective optimization, classification with unbalanced data, feature selection and reduction, and job shop scheduling. He has published over 500 academic papers in refereed international journals and conferences. He has been serving as an associated editor or editorial board member for five international journals (including IEEE Transactions on Evolutionary Computation and the Evolutionary Computation Journal) and as a reviewer of over fifteen international journals. He has been serving as a steering committee member and a program committee member for over eighty international conferences.

- Stefano Cagnoni is an Associate Professor at the University of Parma. Recent research grants include co-management of a project funded by Italian Railway Network Society (RFI) aimed at developing an automatic inspection system for train pantographs, and a "Marie Curie Initial Training Network" grant, for a four-year research training project in Medical Imaging using Bio-Inspired and Soft Computing. Editor-in-chief of the "Journal of Artificial Evolution and Applications" from 2007 to 2010. Since 1999, he has been chair of EvoIASP, an event dedicated to evolutionary computation for image analysis and signal processing, now a track of the EvoApplications conference. Since 2005, he has co-chaired MedGEC, workshop on medical applications of evolutionary computation at GECCO. Co-editor of special issues of journals dedicated to Evolutionary Computation for Image Analysis and Signal Processing. Member of the Editorial Board of the journals "Evolutionary Computation" and "Genetic Programming and Evolvable Machines".

2

Course Agenda

- Introduction
- Evolutionary deep learning for image classification
- Complex system analysis for pattern clustering and feature extraction
- Summary

3

Introduction

4

Computer Vision

- The "art" of making computers see (and understand what they see)
- Computer vision vs image processing
- Sub-topics:
  - Image acquisition
  - Image enhancement
  - Image segmentation
  - 3D-information recovery/feature extraction
  - Image understanding

Computer and Human Vision

HUMAN	COMPUTER
Perception	Image acquisition
Selective information extraction	Feature enhancement (signal/image processing)
Grouping by 'similarity'	Segmentation
Extraction of spatial relationships	3D-information Recovery
Object recognition and semantic interpretation	Image Understanding

Computer and Human Vision

HUMAN	COMPUTER
Perception	Image acquisition
Selective information extraction	Feature enhancement (signal/image processing)
Grouping by 'similarity'	Segmentation
Extraction of spatial relationships	3D-information Recovery
Object recognition and semantic interpretation	Image Understanding

LOW-LEVEL VISION

Computer and Human Vision

HIGH-LEVEL VISION

HUMAN	COMPUTER
Perception	Image acquisition
Selective information extraction	Feature enhancement (signal/image processing)
Grouping by 'similarity'	Segmentation
Extraction of spatial relationships	3D-information Recovery
Object recognition and semantic interpretation	Image Understanding

Application Taxonomy

- EC techniques
  - GA, GP, ES, EP, PSO, DE, LCS, EMO, EDA, etc.
- Solution types
  - Optimisation of parameters of specific solutions (using GA, ES, PSO…)
    - Related with a well-defined task or for a whole system
  - Generation of solutions from scratch (GP, …)
    - Performance optimization based on specific objective functions
    - It is difficult to choose a model with reasonable assumptions
- Role of EC techniques
  - Interactive qualitative comparisons between solutions
  - Generation of emergent collective solutions
    - Achievement of higher-level and complex tasks from collective use of trivial, local, hard-wired behaviours: generation of full EC-based solutions, NOT parameter optimization tasks

Applications

- EC techniques: GP, PSO, LCS, EMO
- Image Analysis
  - Object tracking
  - Edge detection
  - Segmentation
  - Motion detection
  - Object/digit recognition
- English stress detection(signal processing)
- Pattern Recognition: feature selection and biomarker detection

Evolutionary Deep Learning for Image Classification

Evolutionary Deep Learning

- Deep Learning – personal view
  - Definition
  - NN-based deep learning
  - Non-NN type deep learning
- Evolutionary Deep Learning – personal view
  - evolving NNs/neuro-evolution → evolutionary deep learning
  - GAs/PSO/GP for evolving NNs
  - GP for deep learning
- Examples of EvoDL for Image Classification
  - GAs for evolving CNNs for image classification
  - PSO for evolving CNNs for image classification
  - GAs for evolving auto-encoders for image classification
  - GP for evolving deep structures for image classification
- Summary

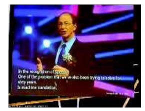
Deep Learning -- Overview

❖ It aims at learning hierarchical/meaningful representations through a *deep* and *non-linear* transformation

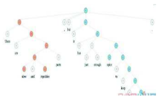
FIELD

- SPEECH : Lower the error rate by 30%, which is a most big breakthrough
- VISION : Error rate from 26% to 15% in ImageNet
- NATURE LANGUAGE PROCESSING : Deep auto-encoders

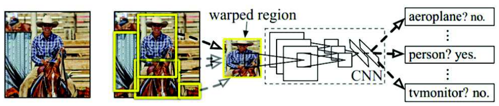

Google voice assistant

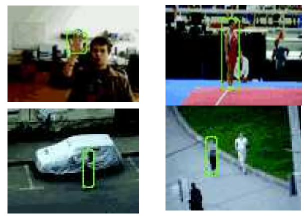

Microsoft Voice

PRODUCT


Stanford emotion analysis

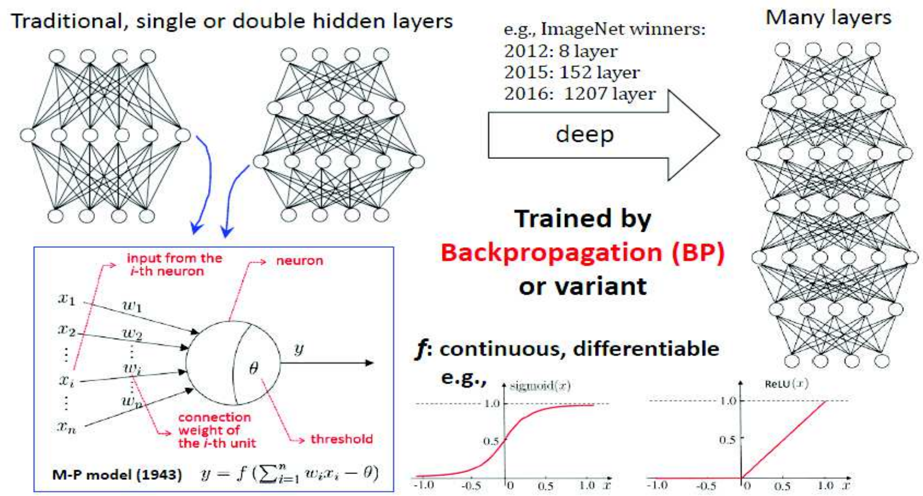

Google Translate



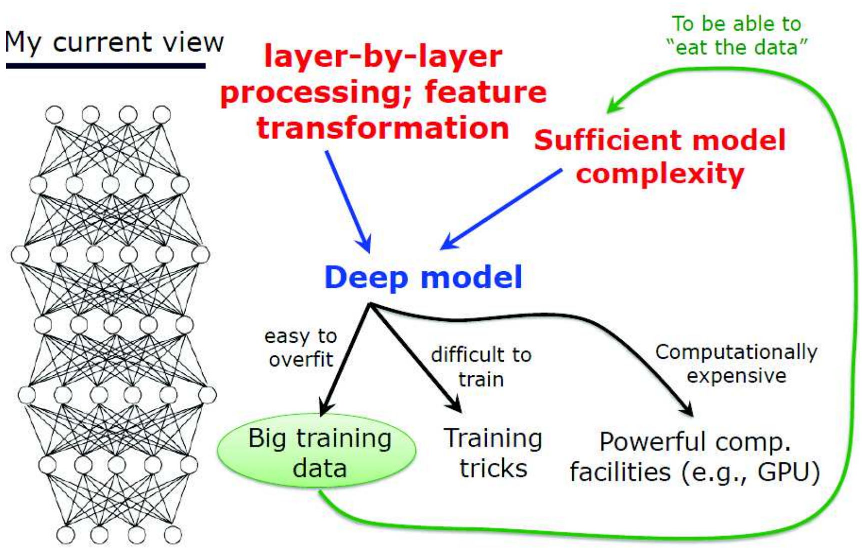


Deep Learning -- Definition

❖ What is "Deep Learning"? Deep Learning = Deep Neural Networks?



Deep Learning – Definition [Zhou]



Deep Learning – My View

- ❖ Layer-by-layer processing
- ❖ Feature transformation
  - Feature extraction
  - Feature construction
  - Feature learning
- ❖ Sufficient model complexity
  - Complexity ≠ the number (#) of nodes, layers
  - Including function complexity
  - Not necessarily symmetrical
- ❖ #examples?
- ❖ Interpretation?

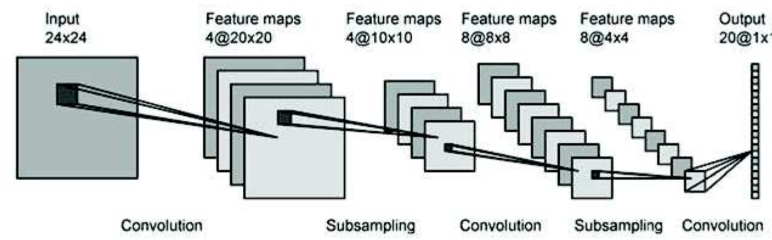
Deep Neural Networks (Learning)

Neural network-based DL methods are very popular

- Convolutional layers, pooling layers, fully-connected layers → Convolutional Neural Networks (supervised)
- Restricted Boltzmann Machines → Deep Belief Networks (unsupervised)
- Auto-encoders → Stacked Deep Auto-encoders (unsupervised)

Convolutional Neural Networks

- Supervised Deep Learning method, dominant DL algorithm
- Rumelhart and PDP Group's T-C Problem of weight Sharing [Chap 8,1986]
- Yan LeCun's SWNNs [1989, 90, …]
- A CNN is composed of multiple convolutional layers, pooling layers and fully-connected layers [1998?]
- State-of-the-art CNNs: VGG (2015), ResNet (2015), DenseNet [2016]

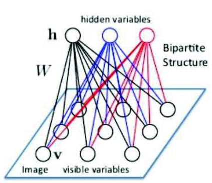


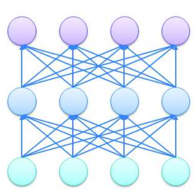
Architecture of LeNet-5

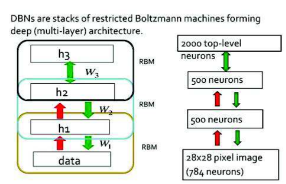
Deep Belief Networks

- Pioneering work on Deep Learning
  - A Fast Learning Algorithm for Deep Belief Nets
    Published in Neural Computing, 2006, Hinton and etc.,
  - Reducing to Dimensionality of Data with Neural Networks
    Published in Science, 2006, Hinton and etc.,
- Unsupervised Deep Learning method (DRBMs)

Geoffrey Hinton



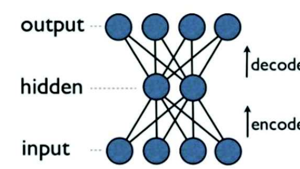


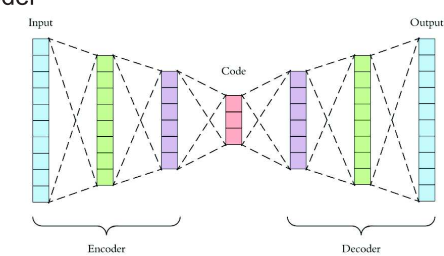


$$E(v,h;\theta) = -\sum_{ij} W_{ij} v_i h_j - \sum_i b_i v_i - \sum_j a_j h_j \qquad \theta = \{W, a, b\}$$

Deep (Stacked) Auto-encoders

- Unsupervised Deep Learning method
- Several variants
  - Deep sparse auto-encoders
  - Deep de-nosing auto-encoders
  - Deep contractive auto-encoders
  - Deep unsymmetrical auto-encoders
  - Deep explicit auto-encoders
- Learn a reconstruction between the output of the decoder and the input of the encoder





Non-NN Deep Learning

- Deep Convex Net [2011]
- PCA Net [2014/15]
- Deep FisherNet [2016]
- Deep Forest learning [2017]
- Genetic Programming based Deep Structures/Learning
  - 2012: GP is doing (evolutionary) deep learning
  - 2018: GP is deep learning

Disadvantages of NN-based DL methods

- Too many hyper-parameters
  - Tricky tuning, particularly for cross-tasks
  - Hard to repeat others' results. For example, when several people use CNNs, they are actually using different learning models due to too many different options such as convolutional-pooling layer structures
- Currently, gradient-based algorithms are used to train the weights
  - Theoretically resulting in local optimal, does not matter too much by using other tricks, such as good initialisation
- Model complexity fixed once structure decided; usually, more than sufficient
- Architectures of state-of-the-art NN DL methods are manually designed
- Big training data required
- Theoretical analysis difficult
- Blackbox and interpretation hard
- …

(Manually Designed) State-of-the-art DNNs and Limitations

- State-of-the-art DNNs
  - ResNet 101, ResNet 1202, DensNet, VGG, Maxout
  - Network in Network, highway Network, All-CNN
  - Manually designed for specific image data classification problems
- Architectures of state-of-the-art DL methods become more and more deep and complex, manual design is difficult to respond
- Manual design highly relies expertise in both DL methods and problems investigated
- Researchers from other communities commonly have no expertise in DL methods
- Evolutionary computation (EC) methods work well in addressing non-convex/no-differentiable problems, and do not require domain knowledge

Evolutionary Deep Learning

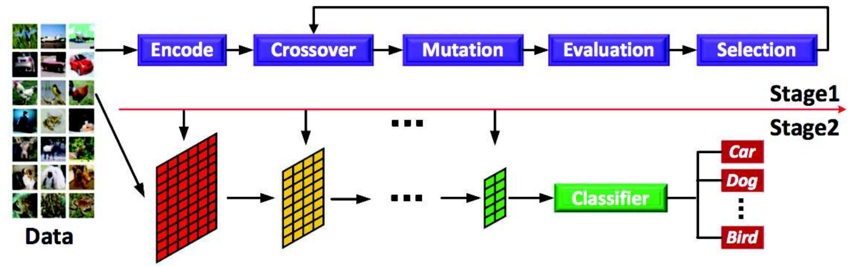
- Two stages:
  - evolving NNs/neuro-evolution →
  - evolutionary deep learning
- GAs/PSO/DE/GP for evolving DNNs
- GP for deep learning

Evolutionary Deep Learning – EC for Evolving NNs

- EC methods have been successfully used to optimize the architecture and even the weights of neural networks over 20 years ago (Yao 1999)
  - Neuro-genetic evolution (Ronald 1994), Cellular Encoding (Gruau 1994)
  - GNARL (Angeline 1994), EPNet (Yao 1997), NEAT (Stanley and Miikkulainen 2002)
  - HyperNEAT (Stanley 2008), ES-HyperNEAT (Risi, Stanley 2012)
  - EANT/EANT2 (Kassahun and Sommer 2005), (Siebel and Sommer 2007)
  - ICONE (Rempis 2012), DXNN (Sher 2012), SUNA (Vargas 2016), MABE (Bohm 2016)
  - CMA-HAGA (Rostami 2016/17), …
- Neural networks were typically shallow and have a small number of parameters
- NEAT and its variants are capable of address the problem regarding median-scale neural networks
- Recently, a number of EC-based new methods have been proposed to automatically evolve/learn DNNs

Evolving Unsupervised DNN

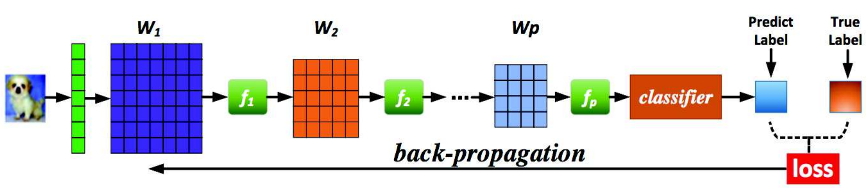
- One method using GA to automatically evolve unsupervised DNN
- The goal is achieved by two stages:
  - Architecture and initialized weights are evolved for building blocks
  - Stacked building blocks stacked are trained by Stochastic Gradient Descent



Yanan Sun, Gary G. Yen, Zhang Yi, "Evolving Unsupervised Deep Neural Networks for Learning Meaningful Representations". IEEE Transactions on Evolutionary Computation. DOI:10.1109/TEVC.2018.2808689.

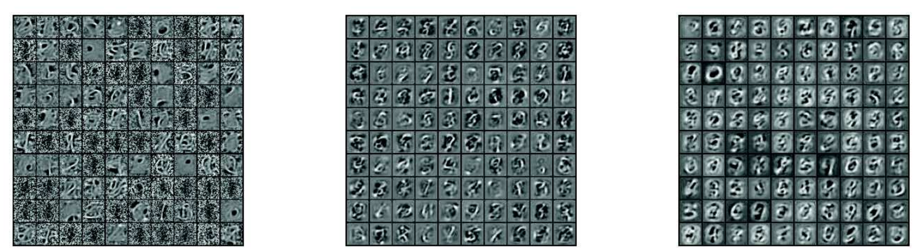
Evolving Unsupervised DNN

- Evolved building blocks are stacked with the architecture and weight initialization values
- Using SGD to achieve the best performance of the deep model



Evolving Unsupervised DNN

Benchmark	EUDNN AE	EUDNN RBM	DAE	CAE	SAE	DBN
MNIST	0.9878(0.00751)	**0.9885(0.00255)**	0.9820(0.00506)(+)	0.9843(0.00699)(+)	0.9832(0.00891)(+)	0.9771(0.00959)(+)
MNIST-basic	0.9674(0.00616)	0.9633(0.00473)	0.9580(0.00352)(+)	0.9635(0.00831)(+)	**0.9776(0.00585)**(-)	0.9658(0.00550)(+)
MNIST-rot	**0.7952(0.00917)**	0.7549(0.00286)	0.7274(0.00757)(+)	0.7706(0.00754)(+)	0.7852(0.00380)(+)	0.7639(0.00568)(+)
MNIST-back-rand	0.8843(0.00076)	0.8386(0.00054)	0.7725(0.00531)(+)	0.5741(0.00779)(+)	**0.8851(0.00934)**(=)	0.8221(0.00130)(+)
MNIST-back-image	0.4325(0.00569)	**0.4830(0.00469)**	0.4022(0.00012)(+)	0.4010(0.00337(+)	0.4638(0.00162)(+)	0.4587(0.00794)(+)
MNIST-rot-back-image	**0.8925(0.00906)**	0.8879(0.00815)	0.8691(0.00127)(+)	0.6574(0.00913)(+)	0.8733(0.00632)(+)	0.8830(0.00098)(=)
Rectangles	0.9627(0.00311)	**0.9681(0.00829)**	0.9232(0.00166)(+)	0.6275(0.00602)(+)	0.9408(0.00263)(+)	0.9622(0.00154)(=)
Rectangles-image	0.7521(0.00689)	0.7716(0.00048)	0.7598(0.00451)(+)	**0.7810(0.00784)**(=)	0.7725(0.00002)(-)	0.7628(0.00913)(+)
Convex	**0.8113(0.00052)**	0.8085(0.00826)	0.7930(0.00538(+)	0.8016(0.00996)(+)	0.8053(0.00878)(+)	0.7895(0.00443)(+)
Cifar10-bw	**0.4798(0.00107)**	0.4331(0.00962)	0.4309(0.00005)(+)	0.4860(0.00775)(+)	0.4423(0.00817)(+)	0.4598(0.00869)(+)
	+/-/=		10/0/0	9/0/1	7/2/1	8/0/2



First layer   Second layer   Third layer

EvoCNN

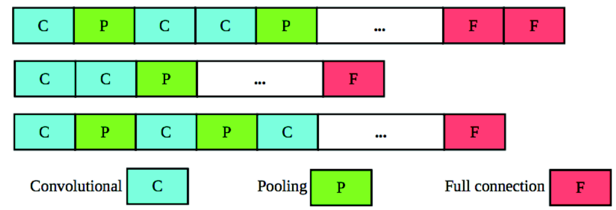
- One method using GA to evolve the architectures and the weight initialization of CNNs
- Designed a variable-length individual method encoding CNNs with unequal depths
- Proposed a crossover operator for individuals with different lengths
- Train the individual with a small number of epochs to find the potentially better one
- Find the best one when the evolutionary process terminates, and then fully trained it for the best performance

Sun, Yanan, Bing Xue, and Mengjie Zhang. "Evolving deep convolutional neural networks for image classification." arXiv preprint arXiv:1710.10741 (2017).

29

EvoCNN

- The architecture is encoded with real numbers representing the configurations of building blocks in CNN

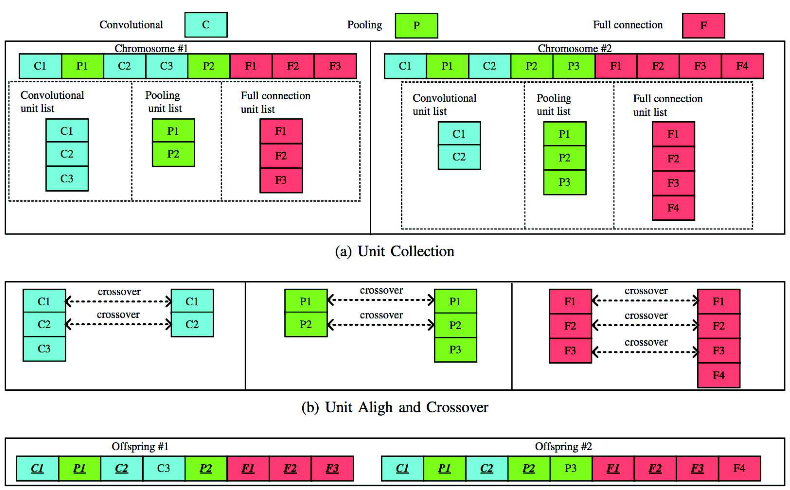


- Weights are initialized with Normal distribution of which the mean and standard derivation are evolved

30

EvoCNN

- Crossover operation is composed of three phases: UC, UAC and UR



(a) Unit Collection

(b) Unit Aligh and Crossover

(c) Unit Restore

31

EvoCNN

- Comparisons on the FASHION dataset

classifier	error(%)	# parameters	# epochs
2C1P2F+Drouout	8.40(+)	3.27M	300
2C1P	7.50(+)	100K	30
3C2F	9.30(+)	—	—
3C1P2F+Dropout	7.40(+)	7.14M	150
GRU+SVM+Dropout	10.30(+)	—	100
GoogleNet [41]	6.30(+)	101M	—
AlexNet [3]	10.10(+)	60M	—
SqueezeNet-200 [51]	10.00(+)	500K	200
MLP 256-128-64	10.00(+)	41K	25
VGG16 [52]	6.50(+)	26M	200
EvoCNN (best)	5.47	6.68M	100
EvoCNN (mean)	7.28	6.52M	100

32

EvoCNN

- Comparisons on the MNIST variants

classifier	MB	MRD	MRB	MBI	MRDBI	Rectangle	RI	Convex
CAE-2 [53]	2.48(+)	9.66(+)	10.90(+)	15.50(+)	45.23(+)	1.21(+)	21.54(+)	—
TIRBM [54]	—	4.20(-)	—	—	35.50(+)	—	—	—
PGBM+DN-1 [55]	—	—	6.08(+)	12.25(+)	36.76(+)	—	—	—
ScatNet-2 [56]	1.27(+)	7.48(+)	12.30(+)	18.40(+)	50.48(+)	0.01(=)	8.02(+)	6.50(+)
RandNet-2 [57]	1.25(+)	8.47(+)	13.47(+)	11.65(+)	43.69(+)	0.09(+)	17.00(+)	5.45(+)
PCANet-2 (softmax) [57]	1.40(+)	8.52(+)	6.85(+)	11.55(+)	35.86(+)	0.49(+)	13.39(+)	4.19(-)
LDANet-2 [57]	1.05(-)	7.52(+)	6.81(+)	12.42(+)	38.54(+)	0.14(+)	16.20(+)	7.22(+)
SVM+RBF [50]	3.03(+)	11.11(+)	14.58(+)	22.61(+)	55.18(+)	2.15(+)	24.04(+)	19.13(+)
SVM+Poly [50]	3.69(+)	15.42(+)	16.62(+)	24.01(+)	56.41(+)	2.15(+)	24.05(+)	19.82(+)
NNet [50]	4.69(+)	18.11(+)	20.04(+)	27.41(+)	62.16(+)	7.16(+)	33.20(+)	32.25(+)
SAA-3 [50]	3.46(+)	10.30(+)	11.28(+)	23.00(+)	51.93(+)	2.41(+)	24.05(+)	18.41(+)
DBN-3 [50]	3.11(+)	10.30(+)	6.73(+)	16.31(+)	47.39(+)	2.61(+)	22.50(+)	18.63(+)
EvoCNN (best)	1.18	5.22	2.80	4.53	35.03	0.01	5.03	4.82
EvoCNN (mean)	1.28	5.46	3.59	4.62	37.38	0.01	5.97	5.39

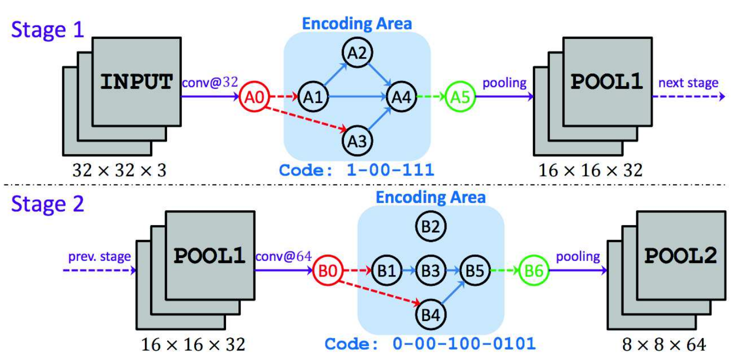
Genetic CNN

- One method using GA evolving CNNs
- The encoding process is composed of multiple stages
- The maximum number of stages must be predefined, which reflects the depth of the evolved CNN
- Each individual is directed trained from scratch
- Individuals have the equal lengths

Lingxi Xie and Alan Yuille, “Genetic CNN,” in Proceedings of 2017 IEEE International Conference on Computer Vision, Venice, Italy, 2017, pp.1388–1397.
(John-Hopskins Uni, USA)

Genetic CNN

- A set of convolutional operations is predefined
- A directed acyclic graph is used to denote the connections
- Binary-string is used to encode such connections
- One-point crossover operation is used



Large-scale Evolution

- One method using GA to evolve architectures of CNNs
- Individuals are with unequal lengths
- Only mutation operation, no crossover operation
- Once a new individual is evaluated, mutation is done and the worse one is discarded
- Fitness is the classification accuracy in terms of image classification tasks
- Weights are inherited from the parent individual

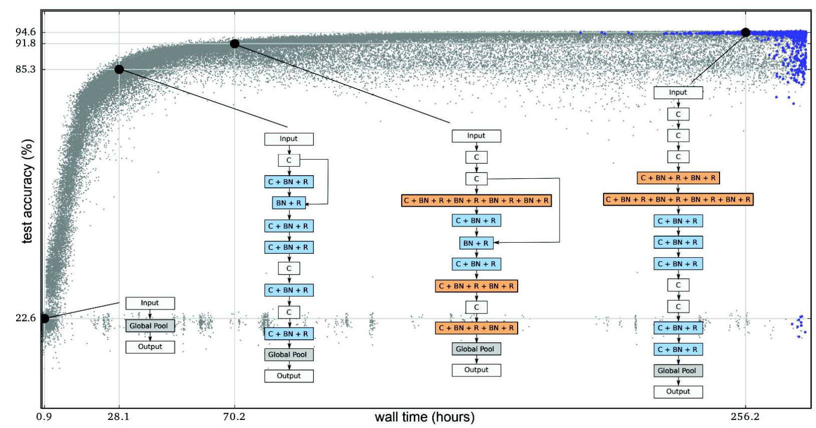
[4] Esteban Real, Sherry Moore, Andrew Selle, Saurabh Saxena, Yutaka Leon Suematsu, Jie Tan, Quoc Le and Alex Kurakin, “Large-scale evolution of image classifiers,” in Proceedings of Machine Learning Research, Sydney, Australia, 2017, pp. 2902–2911.
Google DeepMind

Large-scale Evolution

- ❖ A set of predefined convolutional operations are provided
- ❖ Randomly select multiple convolutional operations and then stacked them
- ❖ During mutation, the setting of one convolutional operation could be changed, removing or adding new connections, and so on
- ❖ Large-scale Evolution defined 12 operations for mutation

Large-scale Evolution

- ❖ Evolutionary process



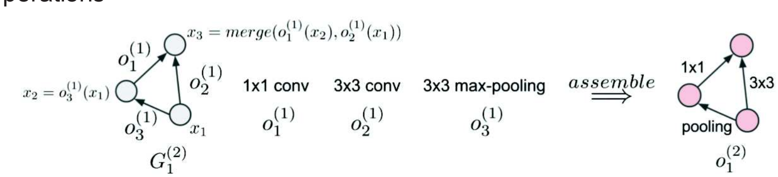
Hierarchical Evolution

- ❖ One method using GA to evolve architectures of CNNs
- ❖ The whole architecture is evolved by several steps
- ❖ In each step, only a small architecture is evolved
- ❖ Multiple small architectures are stacked to form a big/deep architecture

Hanxiao Liu, Karen Simonyan, Oriol Vinyals, Chrisantha Fernando and Koray Kavukcuoglu, "Hierarchical representations for efficient architecture search," in Proceedings of 2018 Machine Learning Research (ICML), Stockholm, Sweden, 2018.

Hierarchical Evolution

- ❖ In the first step, a set of primitive operations is provided
  - 1x1 convolution of C channels
  - 3x3 depth-wise convolution
  - 3x3 separable convolution of C channels
  - 3x3 max-pooling
  - 3x3 average-pooling
  - Identify
- ❖ Randomly select several primitive operations, and then use a Directed Acyclic Graph to denote the connection between selected operations

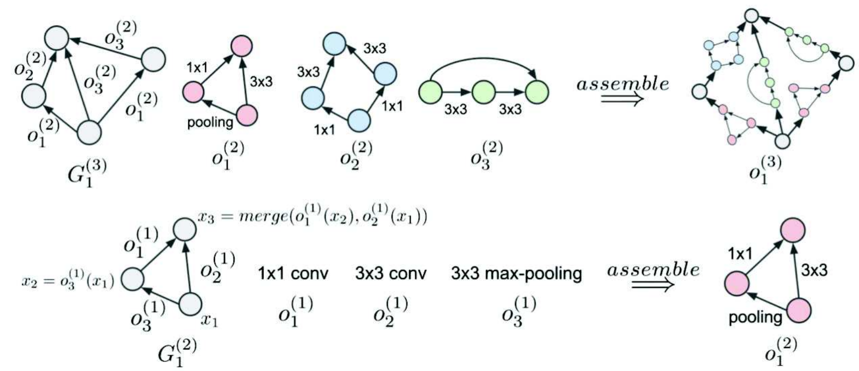


Hierarchical Evolution

- GA is used to change the connections between primitive operations
- Available mutation operations:
  - Add/remove a node in the existing architecture
  - Add a new connection
  - Alter an existing edge
  - Remove an existing edge
- Each individual is evaluated on image classification tasks, the best is selected in terms of the classification accuracy

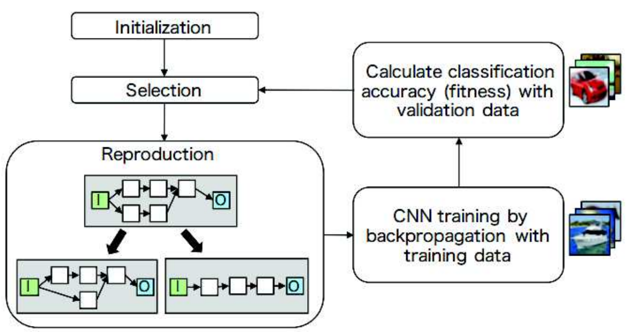
Hierarchical Evolution

- In the second step, the best one found in the previous step is as a new primitive operation, and do the same evolutionary process



CGP-CNN

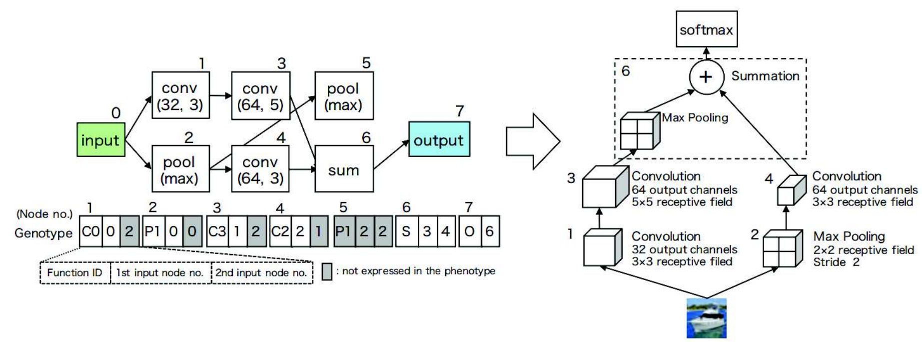
- One genetic programming approach evolving architectures of CNNs
- By providing a set of primitive operations, the Cartesian genetic programming is used to evolve different connections between the primitive operations



Masanori Suganuma, Shinichi Shirakawa and Tomoharu Nagao, "A genetic programming approach to designing convolutional neural network architectures," in Proceedings of the Genetic and Evolutionary Computation Conference. ACM, 2017: 497-504.

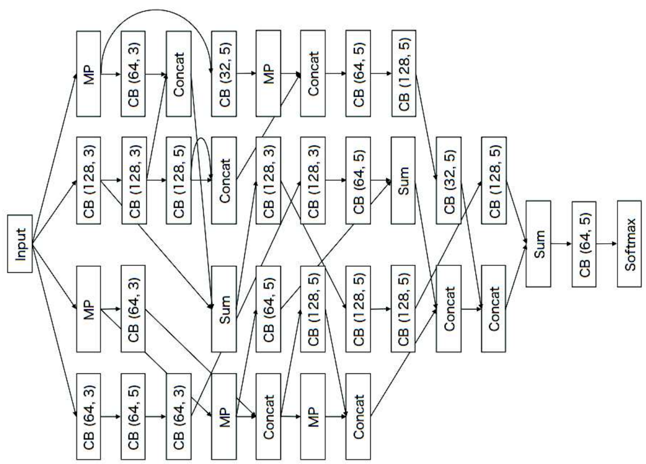
CGP-CNN

- In encoding process, each operation is encoded by three unit, the first is the index, the second and the third refers to the indices of its input
- Each one is converted to the CNN for fitness evaluation on image classification task, the fitness is the corresponding classification accuracy



CGP-CNN

- One example of the evolved CNN



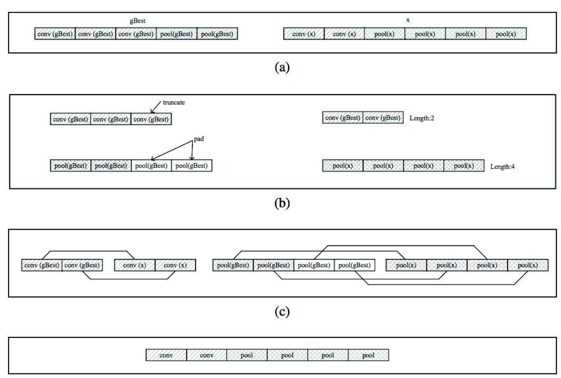
PSOAO

- A PSO method to evolve architectures of the Flexible Convolutional Auto-encoder
  - A traditional convolutional auto-encoder has one encoder and one decoder
    - one encoder is composed of on convolutional layer and on pooling layer
    - one decoder contains only one de-convolutional layer
    - State-of-the-art CNNs do not have such architectures
  - In the flexible convolutional auto-encoder
    - its encoder has multiple convolutional layers and pooling layers
    - It can form the state-of-the-art CNNs
    - but its architecture is not easy to manually tune
- Particles with different lengths to represent different flexible convolutional auto-encoders
- In PSOAO, a x-reference velocity updating strategy is proposed

Yanan Sun, Bing Xue and Mengjie Zhang, "A particle swarm optimization-based flexible convolutional auto-encoder for image classification," arXiv preprint arXiv:1712.05042, 2017.

PSOAO

- In x-reference, gBest and pBest adopt the length of the current particle
- If the lengths of gBest and pBest exceed that of the current particle, truncation is performed, otherwise zeros are padded



(d)

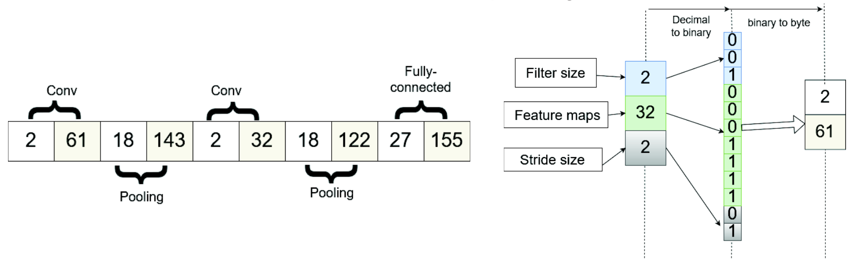
PSOAO

- The performance of flexible convolutional auto-encoder outperforms state-of-the-art auto-encoders and convolutional auto-encoders

Algorithm	CIFAR-10	MNIST	STL-10	Caltech-101
SSAE	74.0 (0.9)	96.29 (0.12)	55.5 (1.2)	66.2 (1.2)
SDAE	70.1 (1.0)	99.06 [25]	53.5 (1.5)	59.5 (0.3)
SCAE	78.2 [16]	99.29 [16]	40.0 (3.1)	58.0 (2.0)
SCRBM	78.9 [63]	99.18 [19]	43.5 (2.3)	65.4 (0.5) [19]
SCDAE-1	75.0 (1.2)	99.17 (0.10)	56.6 (0.8)	71.5 (1.6)
SCDAE-2	80.4 (1.1)	99.38 (0.05)	60.5 (0.9)	78.6 (1.2)
SFCAE-1	78.9 (0.3)	99.30 (0.03)	**61.2 (1.2)**	**79.8 (0.0)**
SFCAE-2	**83.5 (0.5)**	**99.51 (0.09)**	56.8 (0.2)	79.6 (0.0)

IPPSO

- A PSO to effectively evolve the architectures of CNNs
- The encoding strategy is based on IP protocol
- Binary string is used to encode the architecture
- Masks are used to disable/enable the corresponding unit



Bin Wang, Yanan Sun, Bing Xue and Mengjie Zhang, "Evolving deep convolutional neural networks by variable-length particle swarm optimization for image classification," Accepted by IEEE Congress on Evolutionary Computation.

49

IPPSO

- Best on MDRBI
- Second Best on MB
- Fifth on Convex

classier	MB	MDRBI	CS
CAE-2	2.48(+)	45.23(+)	–
TIRBM	–	35.50(+)	–
PGBM+DN-1	–	36.76(+)	–
ScatNet-2	1.27(+)	50.48(+)	6.50(-)
RandNet-2	1.25(+)	43.69(+)	5.45(-)
PCANet-2 (softmax)	1.40(+)	35.86(+)	4.19(-)
LDANet-2	1.05(-)	38.54(+)	7.22(-)
SVM+RBF	3.03(+)	55.18(+)	19.13(+)
SVM+Poly	3.69(+)	56.41(+)	19.82(+)
NNet	4.69(+)	62.16(+)	32.25(+)
SAA-3	3.46(+)	51.93(+)	18.41(+)
DBN-3	3.11(+)	47.39(+)	18.63(+)
IPPSO(mean)	1.21	34.50	12.06
IPPSO(best)	1.13	33	8.48
IPPSO(standard deviation)	0.103	2.96	2.25

50

CNN-GA (our recent work)

- One method to automatically find architectures of CNNs

		CIFAR10	CIFAR100	# Parameter	GPU-day	Manual assistance?
state-of-the-art CNNs	ResNet (depth=101)	93.57	74.84	1.7M	–	completely need
	ResNet (depth=1202)	92.07	72.18	10.2M	–	completely need
	DenseNet	94.17	76.58	27.2M	–	completely need
	VGG	93.34	71.95	20.04M	–	completely need
	Maxout	90.70	61.40	–	–	completely need
	Network in Network	91.19	64.32	–	–	completely need
	Highway Network	92.40	67.66	–	–	completely need
	All-CNN	92.75	66.29	1.3M	–	completely need
semi-automatic algorithms	Genetic CNN	92.90	70.97	–	17	partially need
	Hierarchical Evolution	96.37	–	–	300	partially need
	EAS	95.77	–	23.4M	10	partially need
	Block-QNN-S	95.62	79.35	6.1M	90	partially need
automatic algorithms	Large-scale Evolution	94.60	–	5.4M	2,750	completely not need
	Large-scale Evolution	–	77.00	40.4M	2,750	completely not need
	CGP-CNN	94.02	–	1.68M	27	completely not need
	NAS	93.99	–	2.5 M	22,400	completely not need
	Meta-QNN	93.08	72.86	–	100	completely not need
	CNN-GA (ours)	95.22	–	2.9M	35	completely not need
	CNN-GA (ours)	–	77.97	4.1M	40	completely not need

51

GP-based Evolutionary Deep Learning

- 3-Tier/2-Tier GP for image classification [2012, 2013]
- GP-HoG [2015-16]
- MLGP [2017]
- ConvGP [2017]
- GP-Criptor – (Deep) Transfer Learning [2014-16]

52

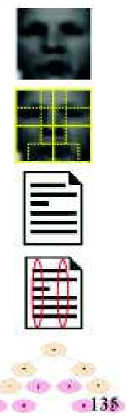
3-Tier/2-Tier GP (2012)

GP for Image Recognition/Classification

The traditional way

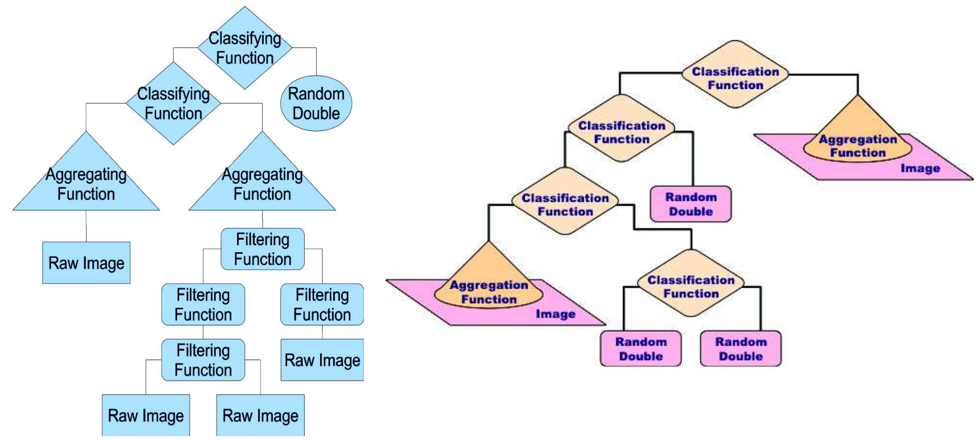
Domain-specific pre-extracted features approach

Input	The input is raw image pixel values
Design	The feature areas need to be designed by domain-experts
Feature Extraction	Transform the pixel values of the selected areas to a different domain
Feature Selection	Select a subset out of the extracted features (optional)
Classification	Feed the extracted features (with or without selection) to a GP-based classifier



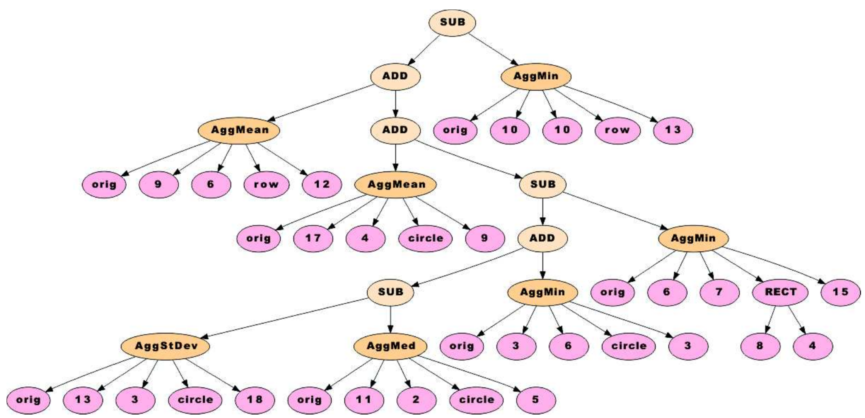
53

3-Tier/2-Tier GP



54

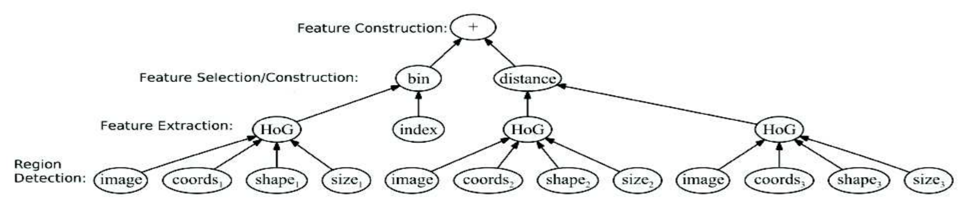
2-Tier GP (2012)



55

GP-HoG [2015-16]

- GP-HoG uses strongly typed GP to perform three tasks in the same tree structure.
- All layers are trained simultaneously and coherently.
- Output of the tree is thresholded.

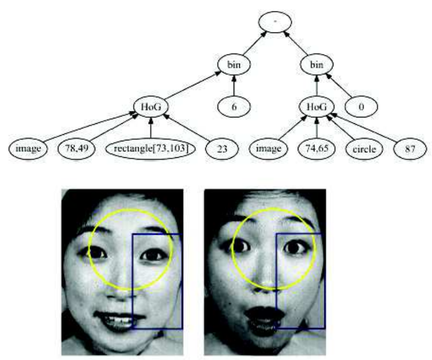


Andrew Lensen, Harith Al-Sahaf, Mengjie Zhang, Bing Xue. "Genetic Programming for Region Detection, Feature Extraction, Feature Construction and Classification in Image Data". Proceedings of the 19th European Conference on Genetic Programming (EuroGP 2016). Lecture Notes in Computer Science. Vol. 9594. Porto, Portugal. March 30 - April 1. 2016. pp. 51-67

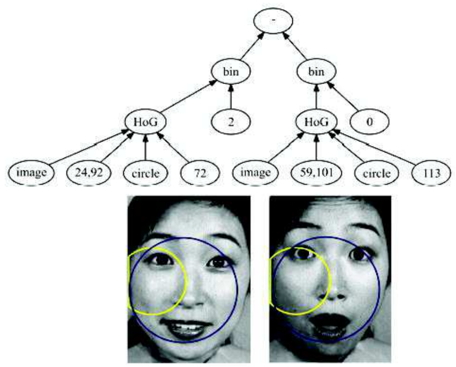
56

GP-HoG

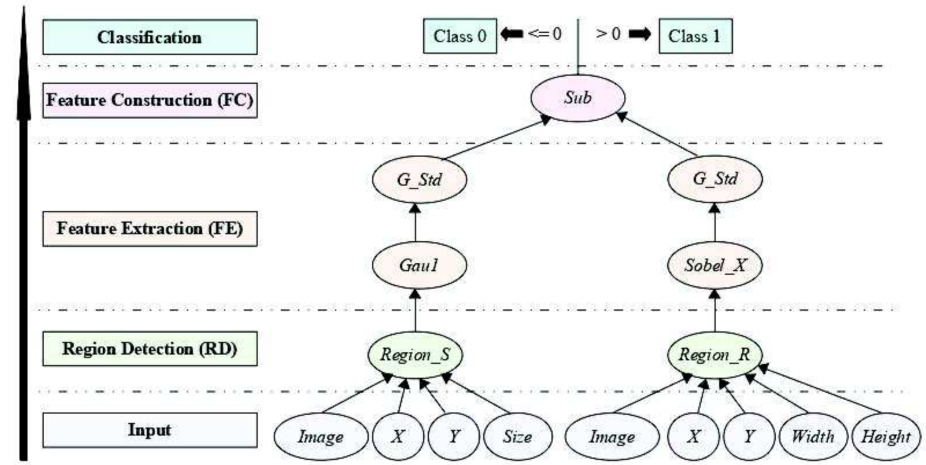
- The below tree has 98% training and 95% test performance on the Jaffe dataset despite being very simple.



- The below tree has 95% training and 100% test performance on the Jaffe dataset.



Multi-Later GP (MLGP) [2017]



Compared with Existing GP methods

	Max	Mean±St.D.	Max	Mean±St.D.	Max	Mean±St.D.
	COIL-20		UIUC		JAFFE	
MLGP	100.0	99.91±0.5	92.38	89.47±2.06	100.0	91.67±6.50
2TGP	100.0	100.0±0.0=	90.48	86.55±2.89+	95.00	82.83±8.53+
3TGP	100.0	100.0±0.0=	93.02	88.42±2.42=	100.0	82.67±9.20+
FeEx+GP	100.0	100.0±0.0=	88.25	81.76±2.56+	90.00	70.67±13.59+
Hist+GP	100.0	99.91±0.50=	65.71	60.81±2.08+	75.00	52.17±9.28+
uLBP+GP	100.0	99.81±0.69=	85.71	81.51±2.22+	65.00	53.83±6.01+
	SCENE		TEXTURE		BIRDS	
MLGP	92.75	90.97±1.40	97.74	90.23±3.48	71.43	61.67±6.45
2TGP	86.23	81.33±2.12+	81.90	75.60±3.87+	67.86	51.79±7.70+
3TGP	88.41	82.56±2.19+	88.24	82.68±4.18+	71.43	56.19±5.75+
FeEx+GP	86.96	83.16±2.37+	88.69	83.65±2.36+	64.29	54.64±5.77+
Hist+GP	86.96	83.29±1.65+	94.57	87.36±3.86+	78.57	51.67±9.53+
uLBP+GP	96.38	92.85±1.92−	97.29	92.37±2.77−	71.43	60.36±7.57=

30 Cases

21 "+"

7 "="

2 "-"

Compared with non-GP methods

Boosting and ensemble methods

	1NN	NB	DT	MLP	AdaBoost	RF	SVM-RDF
	COIL-20		(MLGP 100/99.91±0.5)				
FeEX	100.0=	100.0=	97.22+	100.0=	97.22+	100.0=	97.22+
Histogram	100.0=	100.0=	100.0=	100.0=	100.0=	100.0=	50.00+
GLCM	100.0=	100.0=	100.0=	80.56+	100.0=	100.0=	50.00+
HOG	100.0=	100.0=	100.0=	100.0=	100.0=	100.0=	100.0=
LBP	100.0=	100.0=	100.0=	100.0=	100.0=	100.0=	52.78+
uLBP	100.0=	100.0=	97.22+	100.0=	97.22+	100.0=	50.00+
	UIUC		(MLGP 92.38/89.47±2.06)				
FeEX	83.49+	87.62+	83.49+	77.78+	86.67+	86.67+	77.14+
Histogram	55.87+	66.03+	62.22+	60.95+	65.08+	67.94+	52.38+
GLCM	84.13+	79.68+	85.40+	61.90+	86.98+	86.67+	52.38+
HOG	92.06−	64.76+	86.98+	68.89+	97.14−	92.38−	66.98+
LBP	85.71+	85.40+	79.37+	86.35+	88.25+	84.76+	52.38+
uLBP	86.67+	85.08+	82.22+	83.17+	86.98+	87.62+	52.38+
	JAFFE		(MLGP 100.0/91.67±6.5)				
FeEX	90.00=	55.00+	80.00+	50.00+	85.00+	75.00+	50.00+
Histogram	90.00=	55.00+	60.00+	90.00=	60.00+	45.00+	90.00=
GLCM	65.00+	60.00+	80.00+	50.00+	75.00+	70.00+	50.00+
HOG	100.0−	100.0−	90.00=	50.00+	90.00=	90.00=	90.00=
LBP	75.00+	65.00+	70.00+	80.00+	55.00+	60.00+	50.00+
uLBP	75.00+	65.00+	35.00+	45.00+	65.00+	85.00+	75.00+

10 "+"

32 "="

0 "-"

39 "+"

0 "="

3 "-"

32 "+"

8 "="

2 "-"

Compared with non-GP methods

Boosting and ensemble methods

	1NN	NB	DT	MLP	AdaBoost	RF	SVM-RDF
SCENE (MLGP 92.75/90.97±1.4)							
FeEX	88.41+	85.51+	80.43+	80.43+	86.23+	86.23+	79.71+
Histogram	79.71+	81.16+	81.88+	82.61+	85.51+	87.68+	52.90+
GLCM	92.03−	88.41+	89.86+	91.30=	91.30=	93.48−	52.90+
HOG	94.93−	87.68+	89.13+	89.86+	92.75−	92.03−	90.58=
LBP	89.86+	94.20−	94.20−	87.68+	96.38−	94.93−	52.90+
uLBP	89.86+	94.20−	95.65−	90.58=	96.38−	94.93−	52.90+
TEXTURE (MLGP 97.74/90.23±3.48)							
FeEX	90.50=	84.16+	87.33+	51.13+	90.50=	90.50=	78.28+
Histogram	93.21−	85.97+	91.86−	90.95=	95.93−	94.12−	48.87+
GLCM	83.71+	72.40+	94.57−	47.06+	96.38−	92.31−	48.87+
HOG	81.90+	52.04+	74.21+	52.04+	76.02+	78.73+	52.04+
LBP	98.19−	83.26+	87.78+	93.67−	91.40=	88.69+	48.87+
uLBP	96.83−	86.88+	85.07+	85.52+	90.05=	90.05=	48.87+
BIRDS (MLGP 71.43/61.67±6.45)							
FeEX	57.14+	53.57+	46.43+	53.57+	64.29−	46.43+	53.57+
Histogram	53.57+	50.00+	53.57+	50.00+	53.57+	53.57+	53.57+
GLCM	53.57+	53.57+	60.71=	60.71=	53.57+	53.57+	53.57+
HOG	57.14+	60.71=	57.14+	57.14+	64.29−	53.57+	57.14+
LBP	71.43−	71.43−	57.14+	64.29−	75.00−	78.57−	53.57+
uLBP	78.57−	75.00−	57.14+	75.00−	71.43−	78.57−	53.57+

SCENE: 25"+" 4 "=" 13 "-"

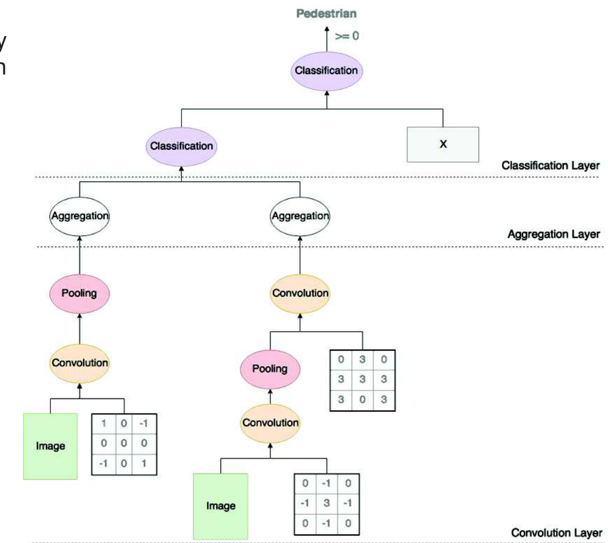
TEXTURE: 25 "+" 7 "=" 10 "-"

BIRDS: 27 "+" 3 "=" 12 "-"

61

ConvGP [2017]

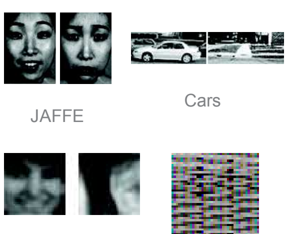
Incorporate key ideas from both GP and CNN!



62

Experiment Design

Datasets



JAFFE

Cars

Faces

Pedestrians

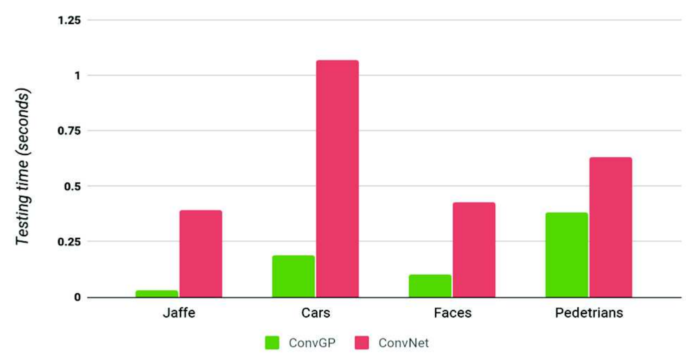
Benchmark Methods

- ConvNet
- Existing GP approach (two-tier GP)
- Decision Trees
- Naive Bayes
- Nearest Neighbour
- Adaboost
- Support Vector Machine

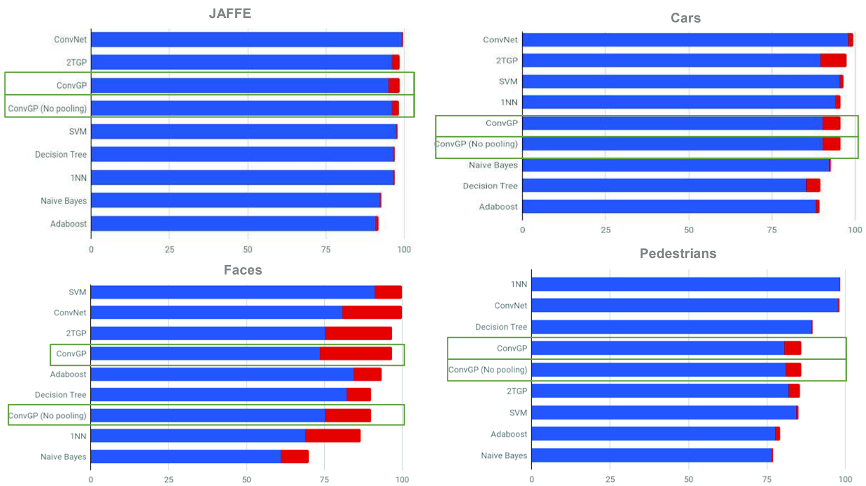
Parameter Settings

Population	1024
Generations	50
Max Depth	10
Tournament	7
Crossover	0.8
Mutation	0.2
Elitism	0.01

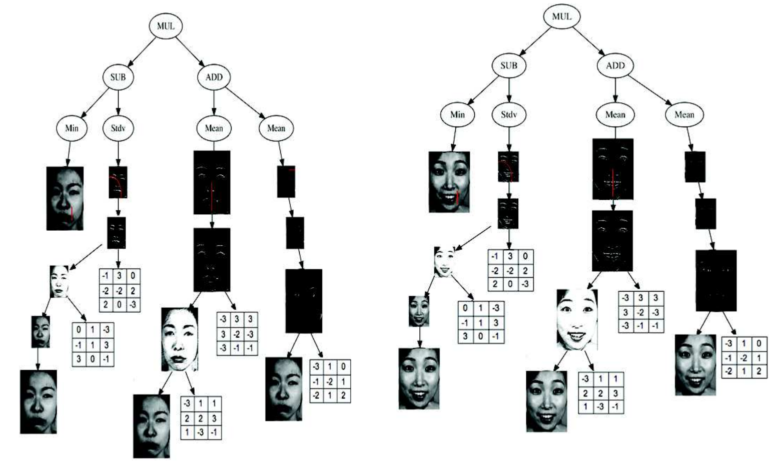
Efficiency Results



Results

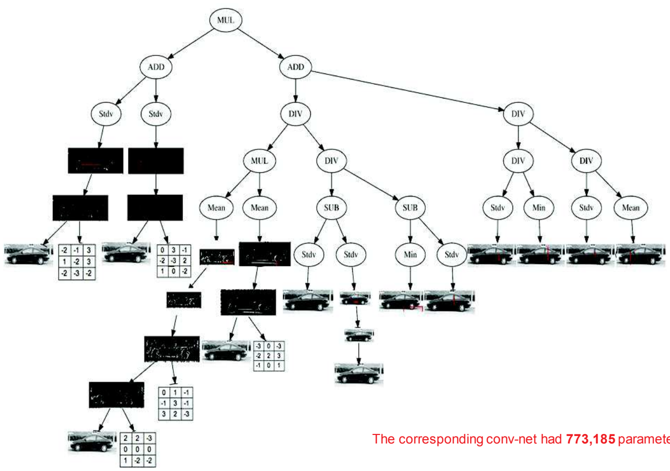


Visualisation (JAFFE)



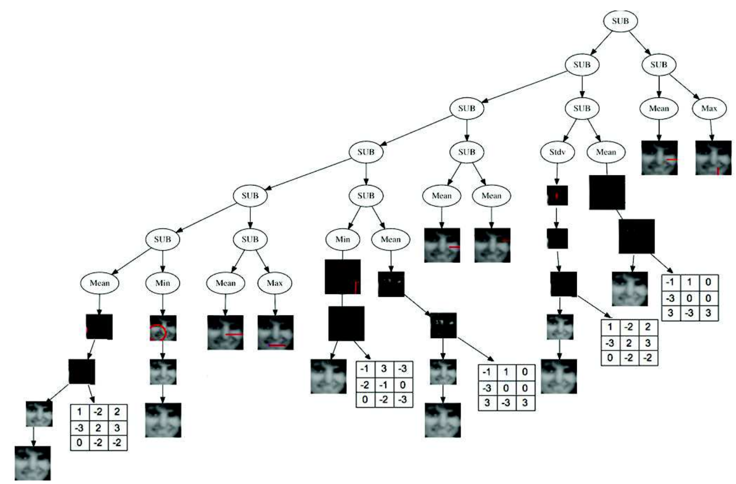
The corresponding conv-net had **5,479,489** parameters

Visualisation (Cars)



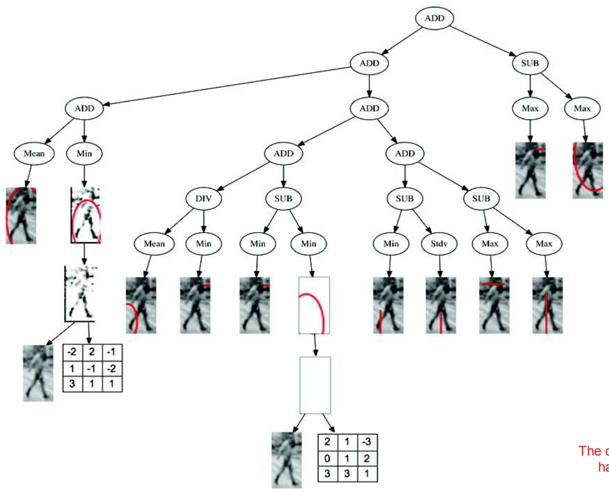
The corresponding conv-net had **773,185** parameters

Visualisation (Faces)



The corresponding conv-net had **56,385** parameters

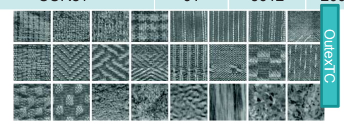
Visualisation (Pedestrian)

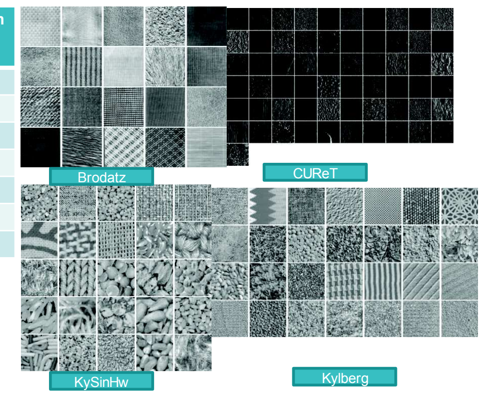


The corresponding conv-net had **105,537** parameters

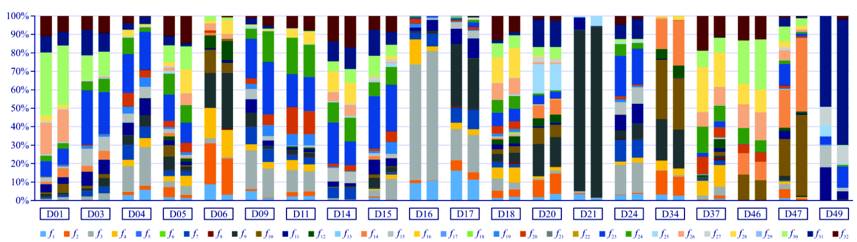
Evolutionary (Deep) Transfer Learning [2015-17]

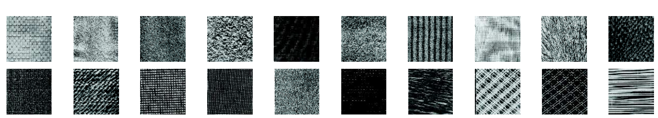
Dataset	Classes	Total instances	Dimensions
Brodatz (No rotation)	20	1680	64 x 64
Brodatz (With rotation)	20	20160	64 x 64
OutexTC	24	2817	128 x 128
KySinHw	25	22500	122 x 122
Kylberg (no rotation)	28	4480	115 x 115
Kylberg (With rotation)	28	53760	115 x 115
CUReT	61	5612	200 x 200





Feature Vector





Harith Al-Sahaf, Ausama Al-Sahaf, Bing Xue, Mark Johnston, Mengjie Zhang. "Automatically Evolving Rotation-invariant Texture Image Descriptors by Genetic Programming". IEEE Transaction on Evolutionary Computation. 2017. pp. 83-101.
Harith Al-Sahaf, Mengjie Zhang, Ausama Al-Sahaf, Mark Johnston. "Keypoints Detection and Feature Extraction: A Dynamic Genetic Programming Approach for Evolving Rotation-invariant Texture Image Descriptors". *IEEE Transaction on Evolutionary Computation*. 2017. DOI: 10.1109/TEVC.2017.2685639.
Muhammad Iqbal, Bing Xue, Harith Al-Sahaf, Mengjie Zhang. "Cross-Domain Reuse of Extracted Knowledge in Genetic Programming for Image Classification". *IEEE Transaction on Evolutionary Computation*. 2017. DOI: 10.1109/TEVC.2017.2657556.

Feature Transfer

Dataset	Classes	Total instances	Dimensions
Webcam	31	795	152-752 x 152-752
Amazon	31	2817	300 x 300
DSLR	31	498	1000 x 1000

Summary

- NN-based evolutionary deep learning has started to demonstrate great potential to outperform the manually designed state-of-the-art deep networks in image classification and analysis
- GP based evolutionary deep learning has also started, and is expected to demonstrate the advantages in effectiveness, efficiency and interpretability in image analysis
- Evolutionary deep learning is still in an early stage, but is expected to show the great accuracy, efficiency, small training set, and good interpretability of the deep models.

73

Acknowledgement

- Thanks my colleagues and research students particularly Dr Bing Xue, Dr Yanan Sun, Andrew Lensen, Ying Bi, Ben Evans, A/Prof Will Browne, Dr Will Smart and Dr Ignas Kukenys, Dr Toktam Ebadi, Dr Mahdi Setayesh, Dr Andy Song, Harith Al-Sahaf, Dr Yuyu Liang, Liam Cervante, Mitch lane, and other members in our ECRG Group.
- Thanks GECCO2018 organisers
- Funding Agents:
  1. Marsden Fund of New Zealand award number(s): VUW 1509, 16-VUW-111, E2280/3663 (Huawei)
  2. University Research Fund at Victoria University of Wellington award number(s): 210375/3557, 209861/3580, 209862/3580, 213150/3662.

74

More Recent Group Photo

35 people -- several people are missing!

75

References

- Yanan Sun, Gary G. Yen, Zhang Yi, "Evolving Unsupervised Deep Neural Networks for Learning Meaningful Representations". IEEE Transactions on Evolutionary Computation. DOI:10.1109/TEVC.2018.2808689.
- Toktam Ebadi, Ignas Kukenys, Will N. Browne, Mengjie Zhang: Human-Interpretable Feature Pattern Classification System Using Learning Classifier Systems. Evolutionary Computation 22(4): 629-650 (2014)
- Bing Xue, Mengjie Zhang, Will N. Browne: Particle Swarm Optimization for Feature Selection in Classification: A Multi-Objective Approach. IEEE T. Cybernetics 43(6): 1656-1671 (2013)
- Soha Ahmed, Mengjie Zhang, Lifeng Peng and Bing Xue."Multiple Feature Construction for Effective Biomarker Identification and Classification using Genetic Programming". Proceedings of 2014 Genetic and Evolutionary Computation Conference (GECCO 2014). ACM Press. Vancouver, BC, Canada. 12-16 July 2014.pp.249--256.
- Bing Xue, Liam Cervante, Lin Shang, Will Browne and Mengjie Zhang. “Binary PSO and rough set theory for feature selection: a multi-objective filter based approach". International Journal of Computational Intelligence and Applications, Vol. 13, No. 2 (2014). pp. 1450009 -- 1- 34. DOI: 10.1142/S1469026814500096
- Hoai Bach Nguyen, Bing Xue, Ivy Liu, Peter Andreae, Mengjie Zhang. "Gaussian Transformation based Representation in Particle Swarm Optimisation for Feature Selection". Proceedings of the 18th European Conference on the Applications of Evolutionary Computation (EuroApplications 2015). Lecture Notes in Computer Science. Vol. 9028. Copenhagen, Denmark. 8-10 April 2015. pp. 541-553

76

References

- Yanan Sun, Bing Xue, Mengjie Zhang, Gary G. Yen, An Experimental Study on Hyper-parameter Optimization for Stacked Auto-Encoders, 2018 IEEE Congress on Evolutionary Computation (CEC). Accepted.
- Bing Xue, Mengjie Zhang, Will Browne, Xin Yao. "A Survey on Evolutionary Computation Approaches to Feature Selection", IEEE Transaction on Evolutionary Computation, vol. 20, no. 4, pp. 606-626, Aug. 2016. doi: 10.1109/TEVC.2015.2504420.
- Stefano Cagnoni, GECCO 2008 and GECCO 2014 Tutorial on ECV/IASP
- Will Smart, Mengjie Zhang. "Tracking Object Positions in Real-time Video using Genetic Programming". In *Proceeding of Image and Vision Computing International Conference,* 2004. pp. 113-118.
- Huayang Xie, Mengjie Zhang, Peter Andreae. "Genetic Programming for Automatic Stress Detection in Spoken English". Proceedings of EvoWorkshops 2006 (EvoIASP 2006), *Lecture Notes in Computer Science*, **Vol. 3907.** Springer. 2006. pp.460-471.
- Peter Andreae, Huayang Xie, Mengjie Zhang. "Genetic Programming for Detecting Rhythmic Stress in Spoken English". *International Journal of Knowledge-Based and Intelligent Engineering Systems (KES Journal). Special Issue on Genetic Programming.* **Vol. 12**, No. 1, 2008. pp. 15-28.
- Mahdi Setayesh, Mengjie Zhang, Mark Johnston:Investigating Particle Swarm Optimisation Topologies for Edge Detection in Noisy Images. Australasian Conference on Artificial Intelligence 2011: 609-618

References

- Yuyu Liang, Mengjie Zhang, Will N. Browne: A Supervised Figure-Ground Segmentation Method Using Genetic Programming. EvoApplications 2015: 491-503
- Mahdi Setayesh, Mengjie Zhang, Mark Johnston: A novel particle swarm optimisation approach to detecting continuous, thin and smooth edges in noisy images. Inf. Sci. 246: 28-51 (2013)
- Aaron Scoble, Will N. Browne, Bill Stephenson, Zane Bruce, Mengjie Zhang: Evolutionary spatial auto-correlation for assessing earthquake liquefaction potential using Parallel Linear Genetic Programming. IEEE Congress on Evolutionary Computation 2013: 2940-2947
- Andy Song, Mengjie Zhang: Genetic programming for detecting target motions. Connect. Sci. 24(2-3): 117-141 (2012)
- Harith Al-Sahaf, Andy Song, Kourosh Neshatian, Mengjie Zhang: Two-Tier genetic programming: towards raw pixel-based image classification. Expert Syst. Appl. 39(16): 12291-12301 (2012)
- Harith Al-Sahaf, Andy Song, Kourosh Neshatian, Mengjie Zhang: Extracting image features for classification by two-tier genetic programming. IEEE Congress on Evolutionary Computation 2012: 1-8
- Harith Al-Sahaf, Mengjie Zhang, Mark Johnston: Genetic Programming for Multiclass Texture Classification Using a Small Number of Instances. SEAL 2014: 335-346

Complex System Analysis for Pattern Clustering and Feature Extraction

Relevant set detection in complex systems

Goal

Identification of *Relevant Subsets* (*RSs*): groups of variables most significant for the system dynamics, which are

- integrated among themselves
- segregated from the rest of the system

Methodology

Detect subsets of system variables which:

- behave in a coherent and coordinated way
- loosely interact with the remainder of the system

Search of structures **ranked** according to some measures based on information theory, which describe the organization of the system

Relevant Index (RI)

Based on the observation of the dynamical states of a system

Defined as the ratio between:

- **Integration**: measure of the total statistical dependence within a subset of variables
- **Mutual information**: measure of the statistical dependence between two subsets of variables: here, between subset S_k and the rest of the system

$$\mathrm{RI}(S_k) = \frac{I(S_k)}{MI(S_k; U-S_k)}$$

U: system S: subset (dimension: k)

I: integration MI: mutual information

T_c Index

Statistical index that measures the deviation of the normalized RI of a group of variables with respect to the statistics of a reference system (homogeneous system)

$$T_C(S_k) = \frac{RI(S_k) - < RI_h >}{\sigma(RIh)}$$

$<RI_h>$, $\sigma(RI_h)$: average and standard deviation of the RI of a sample of subsets of size k extracted from a homogeneous system *U_h*

zI Index

Faster to compute than the T_c and useful to overcome the inefficiency caused by the homogeneous system computation

The product *2mI* (*m* being the number of observations) has a *Chi Square distribution* whose degrees of freedom depend on the size of the analyzed subset and on the cardinality L of the alphabet of its variables

$$zI(S_k) = \frac{2mI(S_k) - <2mI(S_k)>_h}{\sigma(2mI(S_k))_h} = \frac{2mI(S_k) - g(S_k)}{\sqrt{2g(S_k)}}$$

$$g(S_k) = \prod_{j=1}^{k}(L_j) - 1 - \sum_{j=1}^{k}(L_j - 1)$$

$<2mI(S_k)>_h$, $\sigma(2mI(S_k))_h$: average and standard deviation of the Chi Square distribution of all subsets of dimension k of the homogeneous system

Index Computation

- Full description of a dynamical system: index computation for all possible subsets of the system variables (**exhaustive search**)
- The number of possible subsets of the system variables increases **exponentially** with the number of variables
  - An exhaustive search cannot be performed in a reasonable time

Solutions:

- Computational optimization (CUDA C kernels)
- Design of efficient strategies that limit the extension of the search by quickly detecting the most promising subsets

Metaheuristics

Niching memetic algorithm:

- Search for relevant dynamical variable sets (Candidate RSs, CRSs) when the dimension of the variable space increases
- Population diversity maintained during the search process to find more than one (possibly all) RSs

Solutions

- Niching genetic algorithm + local search [6]
- K-Means PSO [5]

Memetic Algorithm

1. **Genetic algorithm** to draw the search towards the basins of attraction of the main local maxima in the search space

   The evolutionary phase is based on the Deterministic Crowding niching technique

2. **Local search** to improve the results, exploring those regions more finely and extensively

1. Genetic Algorithm

Individual:

Subset of variables

- Binary string of size N (N = total number of variables)
- 1 → the corresponding variable is included in the CRS

After single-point crossover, each child replaces the most similar parent of lower fitness

Fitness function:

- zI index of the CRS associated to the individual
- Maximization problem
- Parallel implementation (CUDA C kernel)

2. Local Search

- Creation of a buffer to store the N_{best} CRSs found (highest zI indices) and their fitness values
- The local search strategies are driven by statistics, computed at runtime, on the results obtained by the GA
- Extensive exploration of the regions of the search space that are most likely to have high fitness values

K-Means PSO [4]

Particle representation:

A particle represents a subset of variables

Binary string of size N (N = total number of variables)
1 → the corresponding variable is included in the CRS

Floating point vectors → Binary encoding
1 → particle's elements having a positive value
0 → negative ones

Fitness function:

zI index of the CRS associated to the particle

Maximization problem

Parallel implementation (CUDA C kernel)

K-Means PSO

Algorithm 1 K-means PSO pseudo-code.

```
procedure KPSO
    randomly initialize the particles' positions and velocities
    compute each particle's fitness
    for t = 1 to T do                                   ▷ T = number of iterations
        if t mod c = 0 then                             ▷ every c steps
            run the K-means algorithm to identify niches
        end if
        update each particle's velocity                 ▷ as in standard PSO
        update each particle's position
        compute each particle's fitness
        update each particle's and each niche's best position
    end for
end procedure
```

SPSO

SPSO search process enhanced by a Niching technique

Creation of a buffer to store the best CRSs found (highest zI indices) and their fitness values.

K-Means PSO

Algorithm 1 K-means PSO pseudo-code.

```
procedure KPSO
    randomly initialize the particles' positions and velocities
    compute each particle's fitness
    for t = 1 to T do                                   ▷ T = number of iterations
        if t mod c = 0 then                             ▷ every c steps
            run the K-means algorithm to identify niches
        end if
        update each particle's velocity                 ▷ as in standard PSO
        update each particle's position
        compute each particle's fitness
        update each particle's and each niche's best position
    end for
end procedure
```

K-Means

At regular intervals K-means is applied to the swarm

Reorganization into sub-swarms (by elements' proximity in the search space)

SPSO independently applied to each sub-swarm

Iterative sieving algorithm

The iterative sieving algorithm [1] derives a hierarchy of RSs by iteratively grouping one or more RSs into a single entity

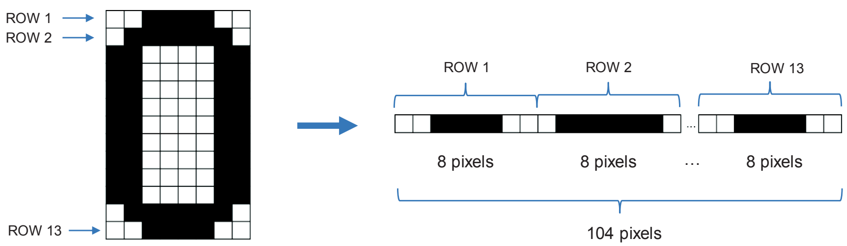
One step of the **sieving algorithm** keeps only those sets that are not included or do not include any other set with higher index value

- **Iterative runs** of the sieving algorithm on the same data, each time using a new representation
- The top-ranked RS of the previous iteration (highest zI) is considered as atomic and substituted by a single variable (group variable)
- The algorithm **stops** when the zI value of the most relevant set detected falls below a pre-set **threshold**

zI/T_c-based clustering and classification

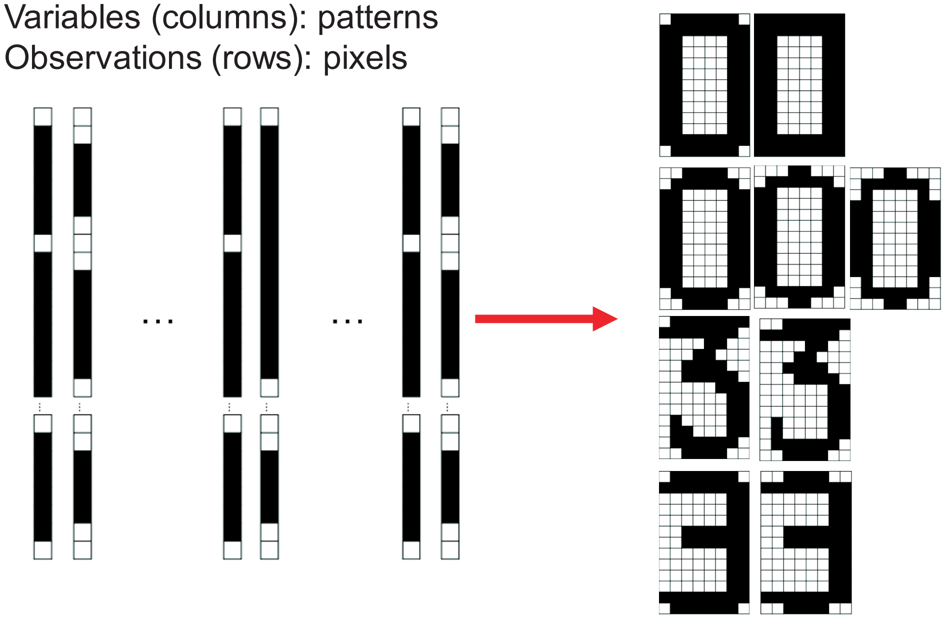
Real-world dataset collected by Società Autostrade SpA at highway toll booths

- 11034 binary patterns representing the ten digits from 0 to 9
- Size: 13x8 pixels → strings of 104 binary features



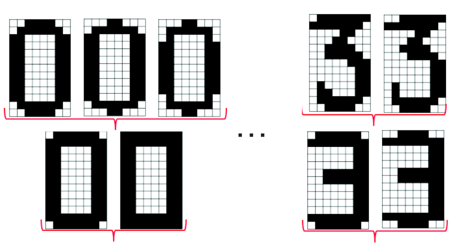
zI-based pattern clustering [2]

Variables (columns): patterns
Observations (rows): pixels



Results

296 groups detected



Classification
296 groups → 296 centroids

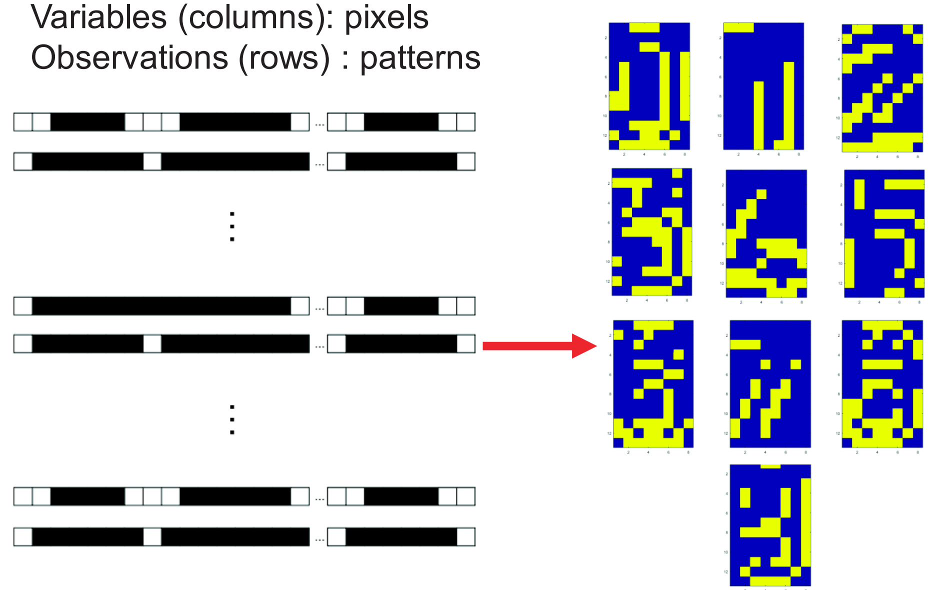
296-centroid distance-based zI classifier

Fully zI-based classifier

Classifier	Accuracy
Random Forest	99.02%
SVM	98.54%
296-centroid LVQ	98.37%
OSLVQ (71-81 centroids)	98.37%
296-centroid distance-based zI	98.27%
Fully zI-based classifier	97.77%
110-centroid distance-based zI	97.65
80-centroid LVQ	97.65%
Naive Bayes Multinomial	96.05%
J48 Consolidated	95.72%

T_c-based classification [3]

Variables (columns): pixels
Observations (rows) : patterns



Results

- Relevant pixel subsets were found in a *supervised* fashion by computing the T_c index separately from patterns representing each digit.
- Considering all 79 subsets having T_c above a given threshold, accuracy of a Random Forest classifier was almost as good (98.93% vs. 99.08%) as using the 104 original features (despite considering almost 25% fewer variables)
- Using only 20 features, accuracy (97.74%) was better than using similarly-sized feature sets computed by more conventional techniques (PCA 97.54%, Weka's InfoGainAttributeEval filter 93.47%)

Ongoing work

- Using the zI along with the iterative sieving algorithm to find pixel subsets relevant for classification
- Allowing the sieving algorithm to generate RSs sharing the same variables (currently they are mutually exclusive)
- Allowing the sieving algorithm to exclude non-relevant sets
- Finding a (classifier-dependent?) optimal representation for the RSs

GECCO

Credits

- Contact person for this work (her current PhD research):
  Laura Sani (laura.sani@unipr.it)
- The RI/T_c/zI indices were defined by
  Roberto Serra, Marco Villani, Andrea Roli
- Many thanks also to:
  Michele Amoretti, Gianluca D'Addese, Chiara Lasagni, Monica Mordonini, Riccardo Pecori, Gianluigi Silvestri, Emilio Vicari

GECCO

References

1. Villani, M., Sani, L., Pecori, R., Amoretti, M., Roli, A., Mordonini, M., Serra, R., Cagnoni, S. (2018) *An Iterative Information-theoretic Approach to the Detection of Structures in Complex Systems*. Complexity
2. Sani, L., D'Addese, G., Pecori, R., Mordonini, M. Villani, M., Cagnoni, S. (2018) *An Integration-Based Approach to Pattern Clustering and Classification* . In: AI*IA 2018 Advances in Artificial Intelligence (pp.362-374). Springer
3. Sani, L., Pecori, R., Vicari, E., Amoretti, M., Mordonini, M., Cagnoni, S. (2018, April). *Can the Relevance Index be Used to Evolve Relevant Feature Sets?*. In International Conference on the Applications of Evolutionary Computation (pp. 472-479). Springer

References

4. Silvestri, G., Sani, L., Amoretti, M., Pecori, R., Vicari, E., Mordonini, M., and Cagnoni, S. (2017). *Searching relevant variable subsets in complex systems using K-means PSO*. In: Italian Workshop on Artificial Life and Evolutionary Computation (pp. 308-321). Springer.
5. Vicari, E., Amoretti, M., Sani, L., Mordonini, M., Pecori, R., Roli, A., Villani, M., Cagnoni, S. and Serra, R. (2016). *GPU-Based Parallel Search of Relevant Variable Sets in Complex Systems*. In Italian Workshop on Artificial Life and Evolutionary Computation (pp. 14-25). Springer
6. Sani, L., Amoretti, M., Vicari, E., Mordonini, M., Pecori, R., Roli, A., Villani, M., Cagnoni, S. and Serra, R. (2016). *Efficient search of relevant structures in complex systems*. In: AI*IA 2016 Advances in Artificial Intelligence (pp. 35-48). Springer

Summary

Concluding Remarks

- Evolutionary computer vision and image analysis is still a big and hot topic
  - Evolutionary deep learning will play a significant role
  - GP-based deep learning will have more developments
  - Interpretability and expandability will be a major focus
- EC techniques will be more popular in pattern recognition
  - GP, GAs, PSO, DE,
  - EC will be in more main stream conferences and journals
  - Including the alpha series: AlphaGo, AlphaZero, AlphaStar
- GPU will be a popular tool