Optimizing Neural-network Learning Rate by Using a Genetic Algorithm with Per-epoch Mutations

Yasusi Kanada
Hitachi, Ltd.

Contents

- ► **Introduction**
- ► **Proposed learning method (LOG-BP)**
- ► **Two application results**
  - ■ Pedestrian recognition (Caltech benchmark)
  - ■ Hand-written digit recognition (MNIST benchmark)
- ► **Conclusion and future work**

Introduction

- ► **Two difficult problems concerning BP** (back propagation)
  - ■ **Decision (or scheduling) of learning rate**
    - Constant or prescheduled learning rates — not adaptive
    - Adaptive scheduling methods — have sensitive hyper parameters difficult to be tuned.
  - ■ **To control locality of search properly**
    - The gradient descent algorithm, including SGD, does not search the space globally.
    - To find a better solution efficiently, multiple trials are required.

Introduction (cont'd)

- **Proposal: LOG-BP** (the learning-rate optimizing genetic back-propagation) **learning method**
  - **A method of new combination of BP and GA** (genetic algorithm)
  - **Multiple neural networks run in parallel.**
  - **Per-epoch genetic operations are used.**

Outline of LOG-BP

- ► **Multiple "individuals"** (neural networks) **learn and search for a best network in parallel in LOG-BP.**
- ► **Each individual contains a chromosome *c*.**
  - $c = (\eta;$ w11, w12, …, w1n1, b1; w21, w22, …, w2n2, b2; … ; wN1, wN2, …, wNnN, bN)

    η: learning rate,

    wij ($1 \le j \le ni$): weights of i-th layer of the network,

    bi: bias of i-th layer.
- ► **A mutation-only GA is applied to these chromosomes.**

Learning algorithm of LOG-BP

Initialization Randomize the weights and learning rates

c1		c2		…	c*n*	
W10	$\eta 1$	W20	$\eta 2$	…	W*n*0	ηn

Epoch 1

1.1 Learning by back-propagation (using stochastic gradient-descent with mini-batch)

c1		c2		…	c*n*	
W11	$\eta 1$	**W21**	$\eta 2$	…	**W*n*1**	ηn

Weights are muted.

1.2 Evaluation (calculating validation losses) (least-square errors)

c1
e11 … best

c2
e21 … worst

c*n*
e*n*1

1.3 Selection and mutation (no crossover)

Duplicate and mute c1. Kill c2.

$\eta 1' = f\,\eta 1$ (probability of 0.5)
$\eta 1' = \eta 1 / f$ (probability of 0.5)
($f > 1$)

c1		c1’		…	c*n*	
W11	$\boldsymbol{\eta 1}$	W11	$\boldsymbol{\eta 1'}$	…	W*n*1	ηn

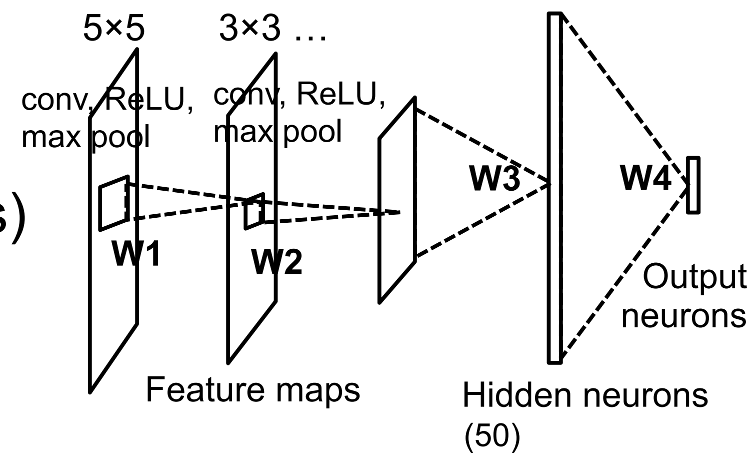
Application to Pedestrian Recognition

► Caltech Pedestrian Dataset

- A famous pedestrian detection benchmark that contains videos with more than 190,000 “small” pedestrians.
- Sets of training data and test data, both of which are 24×48- and 32×64-pixel images were generated.

► Network: CNN2 and CNN3

- Convolutional neural networks (CNNs) with two/three convolution layers



► Environment for computation

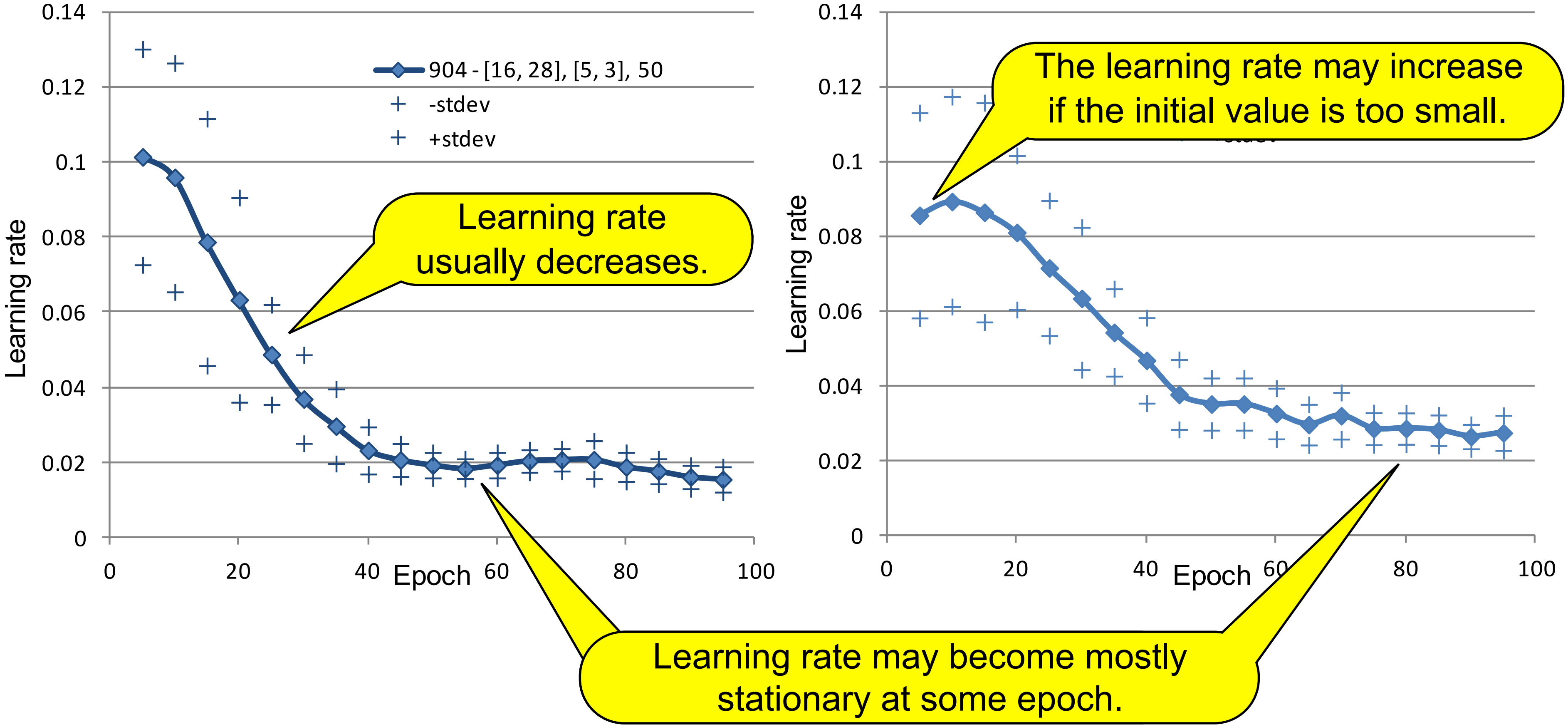
- Deep learning environment: Theano
- GPU: NVIDIA GeForce GTX TITAN X

Pedestrian: Change of Learning Rate

► Example: CNN2

(a) A trial with 12 individuals (filters = [16, 26])

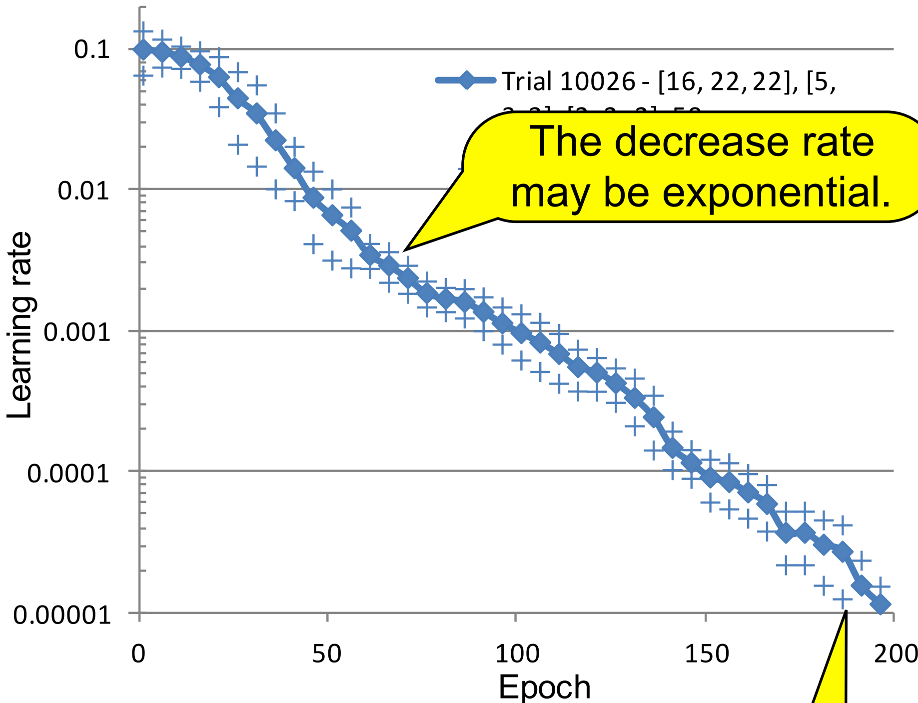
(b) A trial with 12 individuals (filters = [16, 32])



Pedestrian: Change of Learning Rate (cont'd)

► Example: CNN3

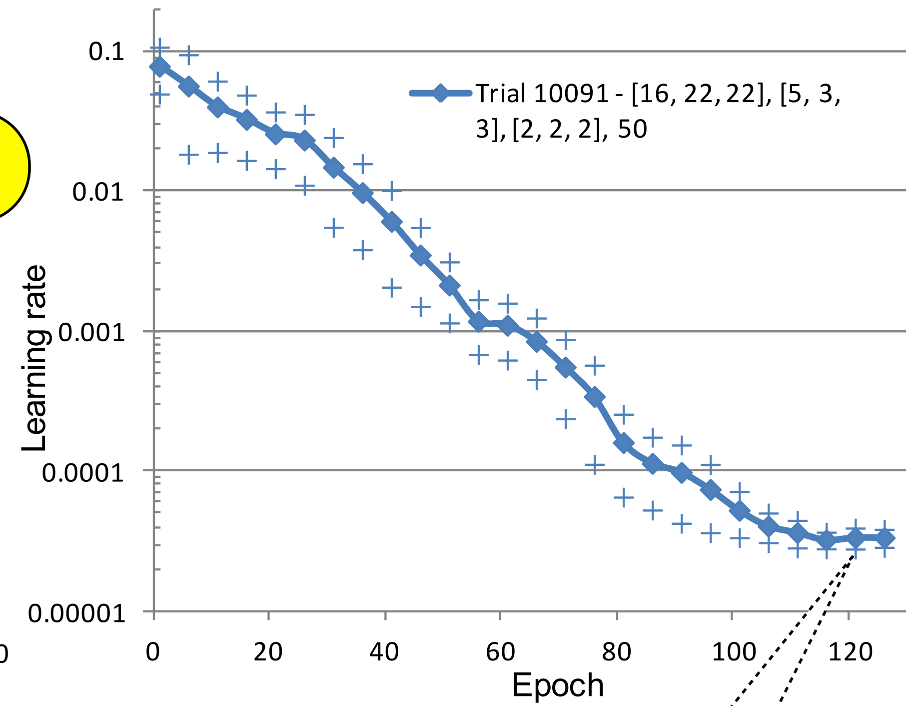
(a) A trial with 12 individuals
(mutation rate = 8.3% (1/12))



The decrease rate may be exponential.

Learning rate continuously decrease by more than three orders of magnitude.

(b) A trial with 24 individuals
(mutation rate = 4.2% (1/24))



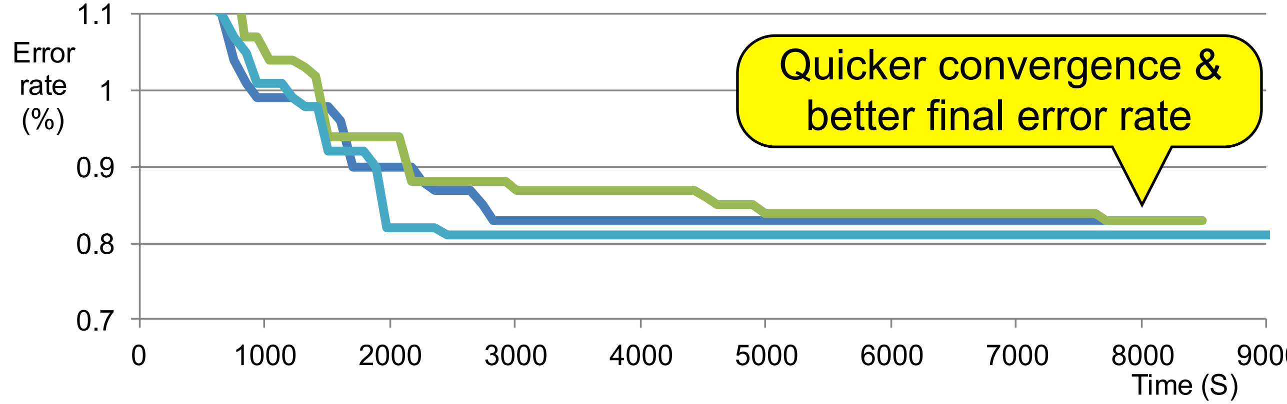
Learning rate may become mostly stationary at some epoch.

Application to MNIST (Character Recognition)

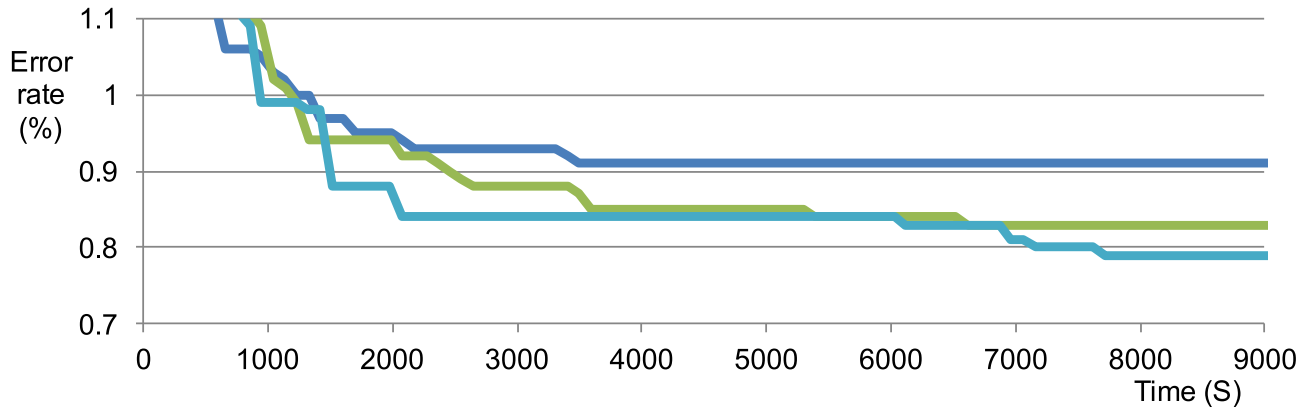
- **MNIST benchmark**
  - A set of hand-written digit images (28×28) containing a training set with 60,000 samples and a test set with 10,000 samples.
- **Adaptive learning rate is not required!**
  - The learning rate during the whole learning process of LOG-BP was around 0.1.
- **Summary: LOG-BP may still have benefits in terms of parallel-search performance.**

MNIST: Performance (CNN3, Examples)

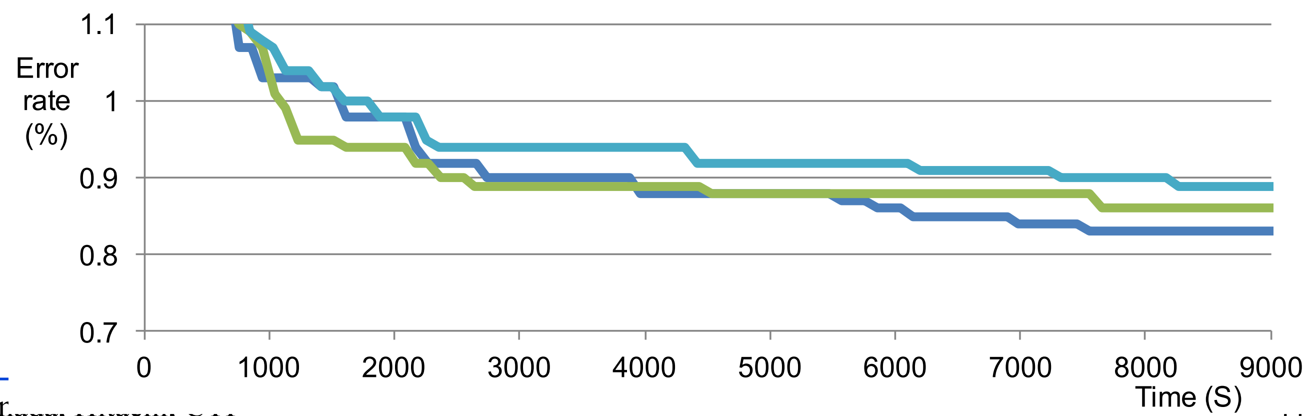
- ▶ **Mutation rate 4%**



- ▶ **Mutation rate 2%**



- ▶ **Mutation rate 0**



MNIST: Performance (CNN3, Statistics)

► **Convergence time and final error rate**

Mutation rate	Average convergence time (std. dev., s)	Final error rate (std. dev., %)
4%	2.6×10^3 (0.5×10^3)	0.82 (0.01)
2%	4.9×10^3 (2.6×10^3)	0.84 (0.07)
0%	5.6×10^3 (0.9×10^3)	0.86 (0.03)

Conclusion

- **LOG-BP that combines BP and a GA by a new manner is proposed.**
- **LOG-BP solves two problems concerning BP.**
  - Scheduling of learning rate
  - Controlling locality of search
- **Two benchmarks show high performance of LOG-BP.**
  - The MNIST benchmarking suggests advantages of LOG-BP over conventional SGD algorithms.
  - LOG-BP will make machine learning less dependent to properties of various applications.