华中科技大学
人工智能与自动化学院
SCHOOL OF ARTIFICIAL INTELLIGENCE AND AUTOMATION.HUST

HUST BCI&ML Lab
脑机接口与机器学习实验室
BRAIN-COMPUTER INTERFACE AND MACHINE LEARNING LABORATORY

New Optimization Techniques for TSK Fuzzy Systems

Dongrui Wu

School of Artificial Intelligence and Automation

Huazhong University of Science and Technology

drwu@hust.edu.cn

https://lab.bciml.cn

Outline

- **Fuzzy Sets**
- TSK Fuzzy Systems (FSs)
- Equivalence between TSK FSs and other Machine Learning Models
- Optimize TSK FSs for Regression Problems
- Optimize TSK FSs for Classification Problems
- Conclusions

Fuzzy Sets

- First proposed by Prof. **Lotfi A. Zadeh** (UC Berkeley) in 1965.
- An approach to model subjective knowledge.

- *I knew that the word "fuzzy" would make the theory controversial. Knowing how the real world functions, I submitted my paper to Information and Control because I was a member of the Editorial Board. There was just one review-which was very lukewarm.* ***I believe that my paper would have been rejected if I were not on the Editorial Board.*** (Zadeh L.A. (2011); My Life and Work - A Retrospective View, Applied and Computational Mathematics, Special Issue on Fuzzy Set Theory and Applications, Dedicated to the 90th Birthday of Prof. Lotfi A. Zadeh, 10(1), 4-9, 2011.)

INFORMATION AND CONTROL 8, 338–353 (1965)

129,325 Google Scholar citations, 4/13/2023

Fuzzy Sets*

L. A. Zadeh

Department of Electrical Engineering and Electronics Research Laboratory, University of California, Berkeley, California

A fuzzy set is a class of objects with a continuum of grades of membership. Such a set is characterized by a membership (characteristic) function which assigns to each object a grade of membership ranging between zero and one. The notions of inclusion, union, intersection, complement, relation, convexity, etc., are extended to such sets, and various properties of these notions in the context of fuzzy sets are established. In particular, a separation theorem for convex fuzzy sets is proved without requiring that the fuzzy sets be disjoint.

I. INTRODUCTION

More often than not, the classes of objects encountered in the real physical world do not have precisely defined criteria of membership. For example, the class of animals clearly includes dogs, horses, birds, etc. as its members, and clearly excludes such objects as rocks, fluids, plants, etc. However, such objects as starfish, bacteria, etc. have an ambiguous status with respect to the class of animals. The same kind of ambiguity arises in the case of a number such as 10 in relation to the "class" of all real numbers which are much greater than 1.

Clearly, the "class of all real numbers which are much greater than 1," or "the class of beautiful women," or "the class of tall men," do not constitute classes or sets in the usual mathematical sense of these terms. Yet, the fact remains that such imprecisely defined "classes" play an important role in human thinking, particularly in the domains of pattern recognition, communication of information, and abstraction.

The purpose of this note is to explore in a preliminary way some of the basic properties and implications of a concept which may be of use in

* This work was supported in part by the Joint Services Electronics Program (U.S. Army, U.S. Navy and U.S. Air Force) under Grant No. AF-AFOSR-139-64 and by the National Science Foundation under Grant GP-2413.

338

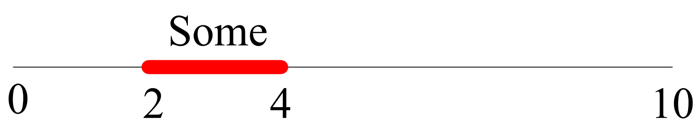
Most Cited Machine Learning Papers (4/13/2023)

1. **K. He**, X. Zhang, S. Ren and **J. Sun**, "Deep residual learning for image recognition," *CVPR* 2016. **160,317**
2. **D.P. Kingma** and J. Ba, "Adam: A method for stochastic optimization," *arXiv:1412.6980*, 2014. **141,178**
3. **A. Krizhevsky**, I. Sutskever and **G.E. Hinton**, "ImageNet classification with deep convolutional neural networks," *NeuRIPS* 2012. **130,639**
4. **L.A. Zadeh**, "Fuzzy sets," *Information and Control*, 1965. **129,325**
5. **L. Breiman**, "Random forests," *Machine Learning*, 2001. **106,831**
6. **V. Vapnik**, "The Nature of Statistical Learning Theory," *Data Mining and Knowledge Discovery*, 1995. **101,688**

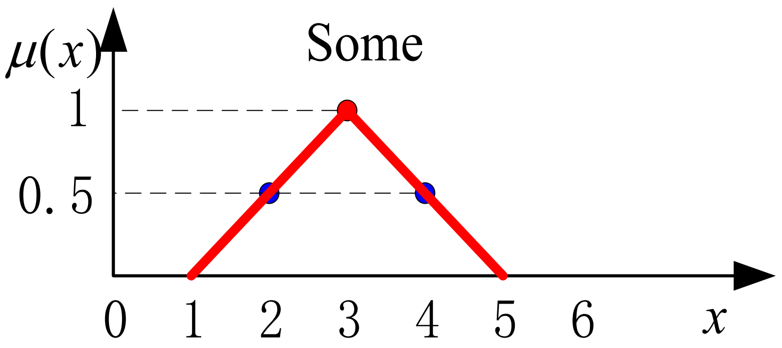
Fuzzy Sets

Model the word *“some ”*

Number

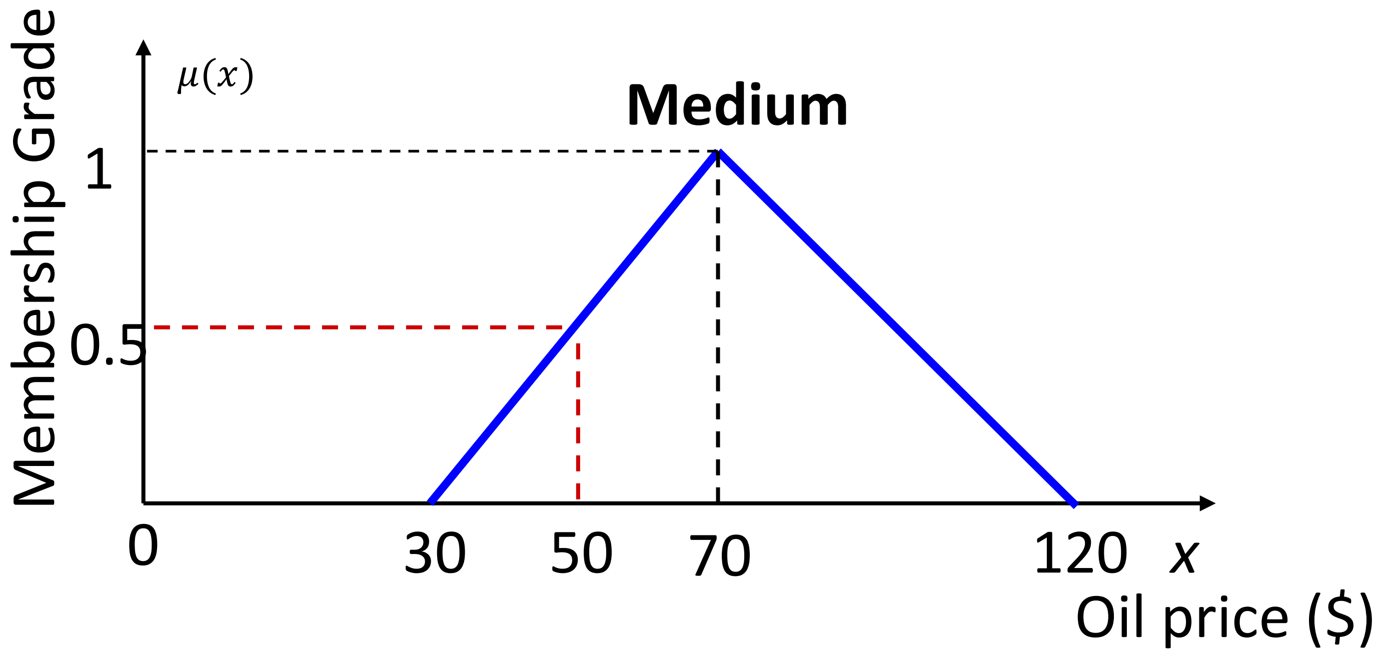


Interval



Fuzzy set to model linguistic uncertainty

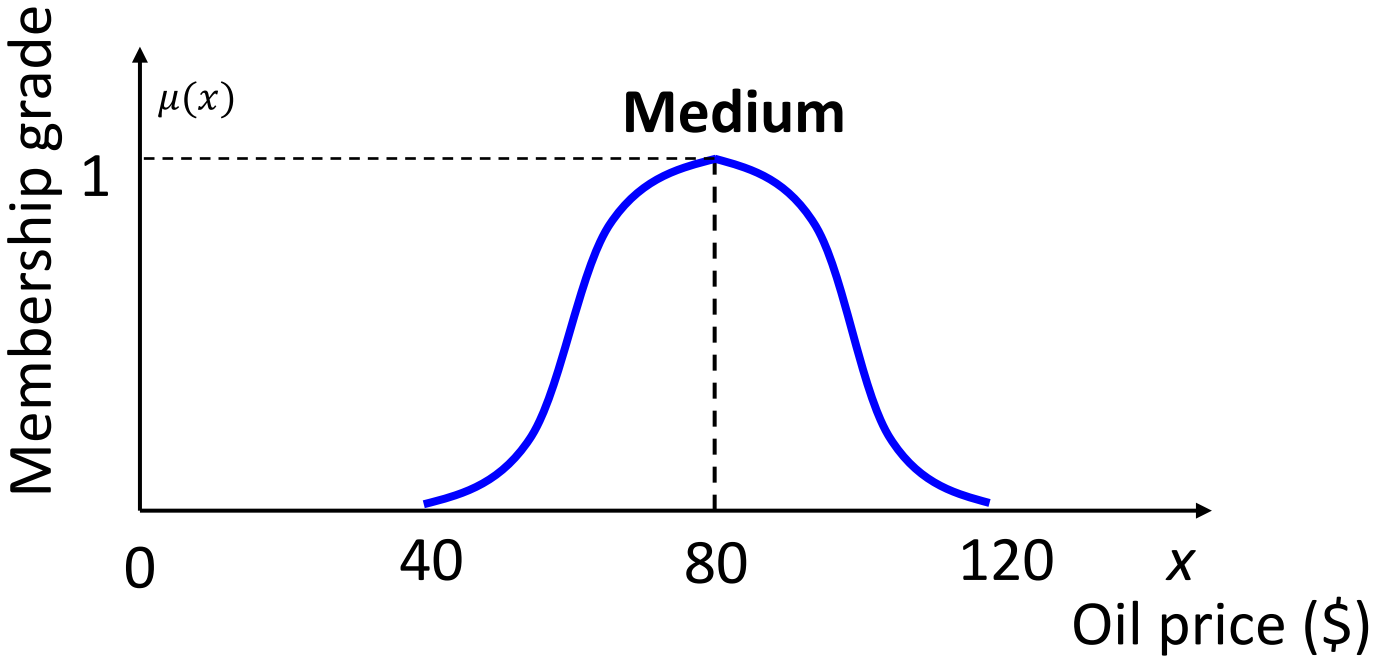
Triangular Fuzzy Set



Membership function (MF): $$\mu(x) = \begin{cases} 0, & x \le 30 \text{ or } x \ge 120 \\ (x-30)/40, & 30 < x \le 70 \\ (120-x)/50, & 70 < x \le 120 \end{cases}$$

Three numbers to determine a triangular MF

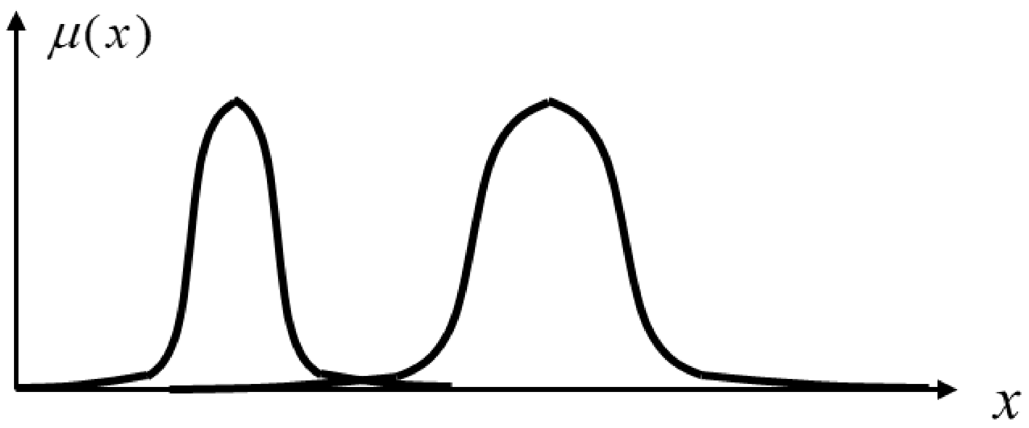
Gaussian Fuzzy Set

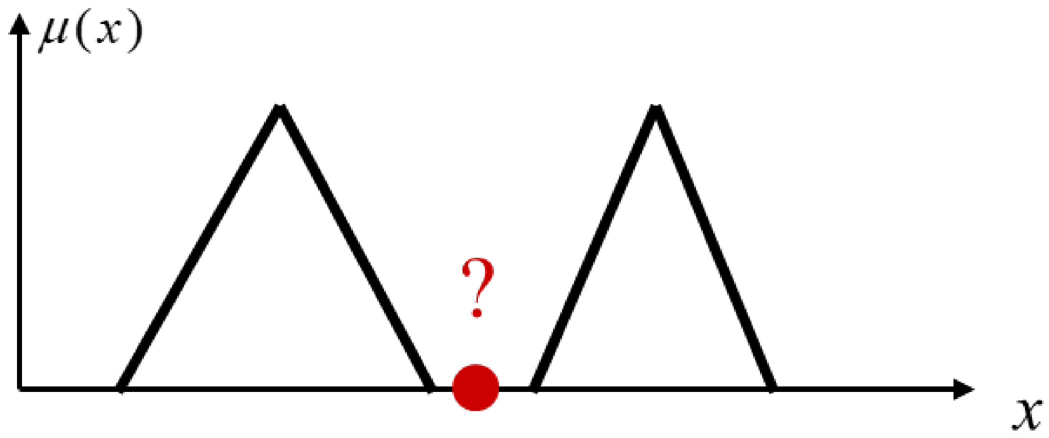


Membership function (MF): $$\mu(x) = \exp\left(-\frac{(x-m)^2}{2\sigma^2}\right)$$

Two numbers to determine a Gaussian MF

Gaussian Fuzzy Set: Property





Can cover the entire input domain with an arbitrary number of MFs

D. Wu and J. M. Mendel, “On the Continuity of Type-1 and Interval Type-2 Fuzzy Logic Systems,” *IEEE Trans. on Fuzzy Systems*, 19(1):179-192, 2011.

Outline

- Fuzzy Sets
- **TSK Fuzzy Systems (FSs)**
- Equivalence between TSK FSs and other Machine Learning Models
- Optimize TSK FSs for Regression Problems
- Optimize TSK FSs for Classification Problems
- Conclusions

Takagi-Sugeno-Kang (TSK) Rules

$$\text{IF } \overbrace{x_1 \text{ is } F_1 \text{ and } x_2 \text{ is } F_2}^{\text{Antecedents}}, \text{THEN } \overbrace{y = ax_1 + bx_2 + c}^{\text{Consequent}}$$

- y can also be a nonlinear function of x_1 and x_2.
- In practice usually the simplest approach is used, i.e., $y=c$, where c is a constant different from rule to rule.

TSK Rulebase

Oil price

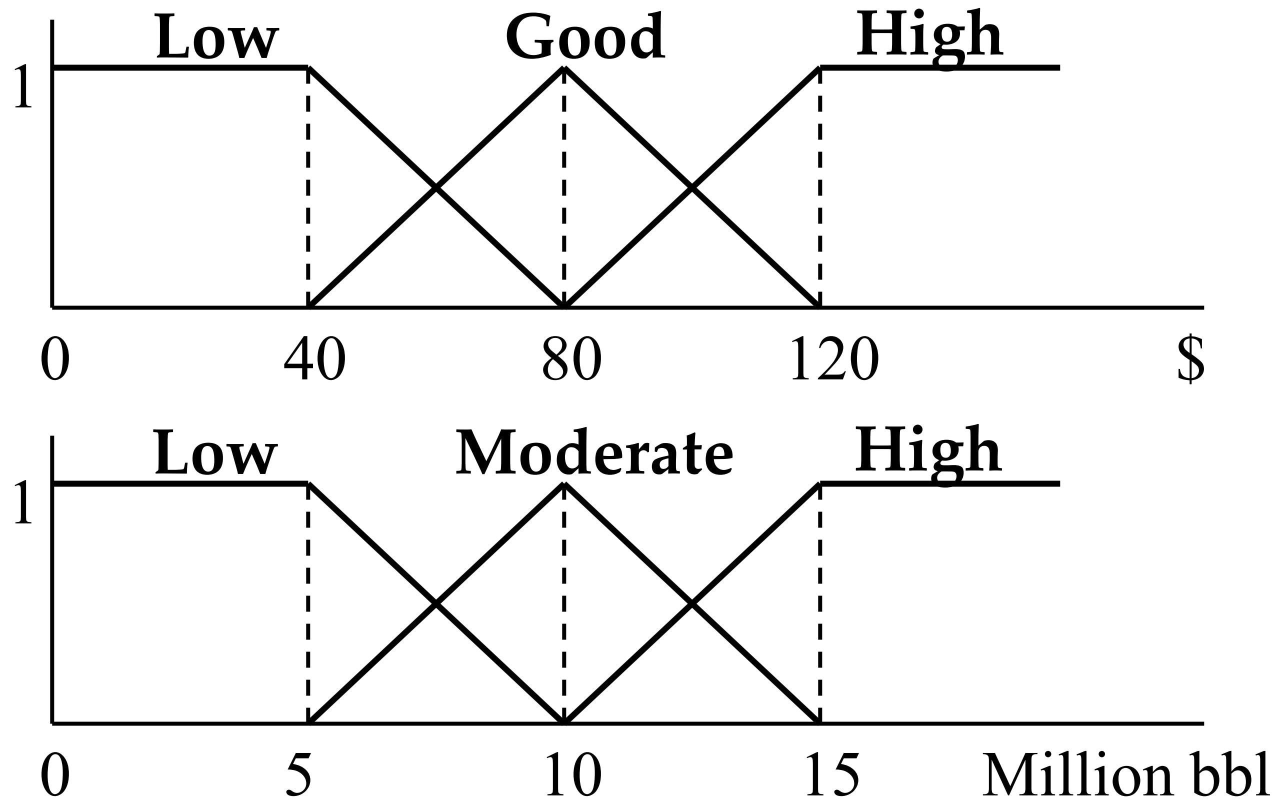
High	Rule 1 c_1	Rule 2 c_2	Rule 3 c_3
Good	Rule 4 c_4	Rule 5 c_5	Rule 6 c_6
Low	Rule 7 c_7	Rule 8 c_8	Rule 9 c_9
	High	Moderate	Low

Total proven reserves

Example: TSK Rules

Input 1: Oil price

Input 2: Total proven reserves



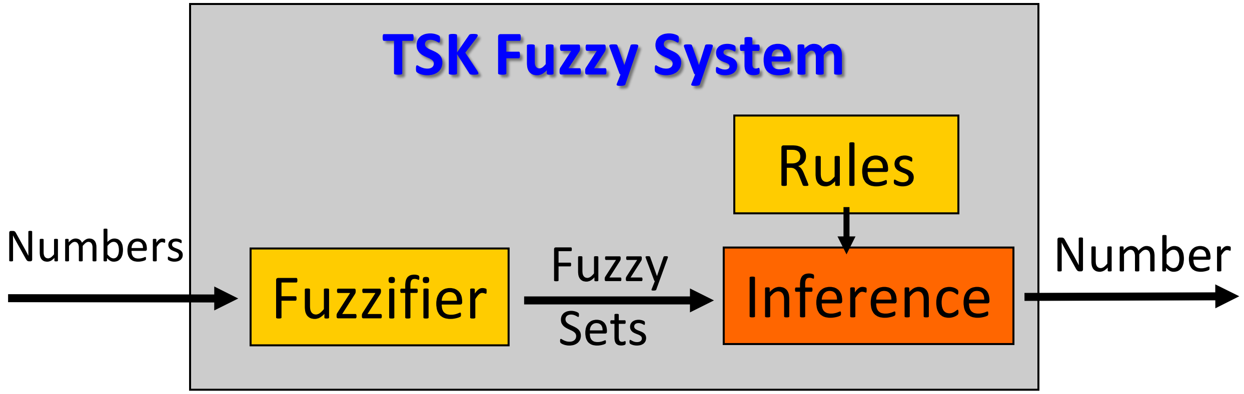
IF Oil price is High and Total proven reserves are High, THEN Enhanced recovery is 10.
IF Oil price is Good and Total proven reserves are High, THEN Enhanced recovery is 5.

Example: TSK Rulebase

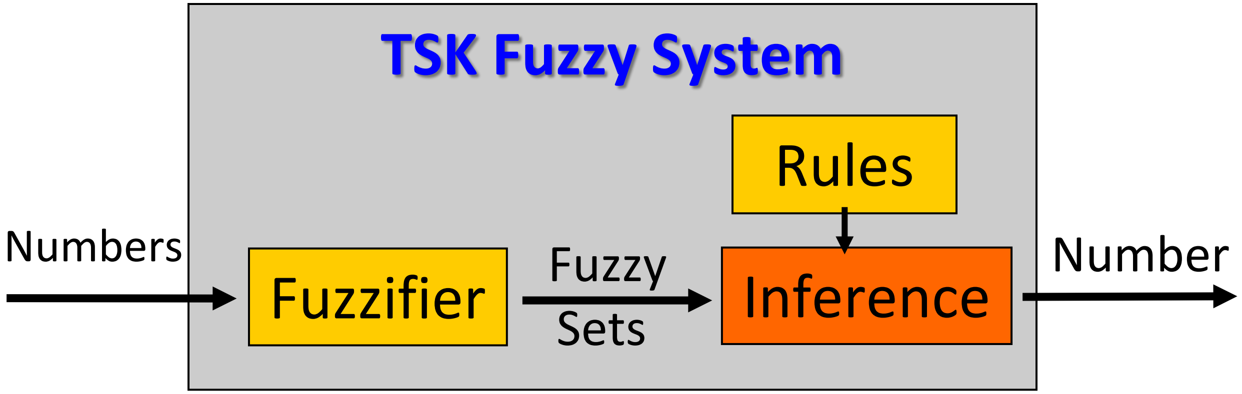
Oil price \ Total proven reserves	High	Moderate	Low
High	Rule 1: 10	Rule 2: 7	Rule 3: 5
Good	Rule 4: 5	Rule 5: 3	Rule 6: 3
Low	Rule 7: 3	Rule 8: 0	Rule 9: 0

TSK Inference

$$\text{IF } \overbrace{x_1 \text{ is } F_1 \text{ and } x_2 \text{ is } F_2}^{\text{Antecedents}}, \text{ THEN } \overbrace{y = ax_1 + bx_2 + c}^{\text{Consequent}}$$



TSK Inference



Inference: Combines the fuzzy IF-THEN rules.

1. Compute the firing level of each rule.
   ①Compute the firing level of each MF in the antecedent part of a rule.
   ②Combine these firing levels of antecedent MFs in a meaningful way to obtain the firing level of that rule.
2. Combine the fired rules using weighted average.

Inference: Example

Rule 1 10 $\mu_1 = 0$	Rule 2 7 $\mu_2 = 0.25$	Rule 3 5 $\mu_3 = 0.25$
Rule 4 5 $\mu_4 = 0$	Rule 5 3 $\mu_5 = 0.6$	Rule 6 3 $\mu_6 = 0.4$
Rule 7 3 $\mu_7 = 0$	Rule 8 0 $\mu_8 = 0$	Rule 9 0 $\mu_9 = 0$

$$y = \frac{\sum_{i=1}^{9} y_i \mu_i}{\sum_{i=1}^{9} \mu_i}$$

$$= \frac{7 \times 0.25 + 5 \times 0.25 + 3 \times 0.4 + 3 \times 0.6}{0.25 + 0.25 + 0.4 + 0.6}$$

$$= 4.0$$

A Short Video Tutorial

IEEE
Computational
Intelligence
Society

- This first-prize-winning video was sponsored by the IEEE Computational Intelligence Society (CIS) in its 2011 Fuzzy Logic Video Competition
- The IEEE CIS has free rights to use this video as it sees fit
- The video does not infringe any copyrighted materials

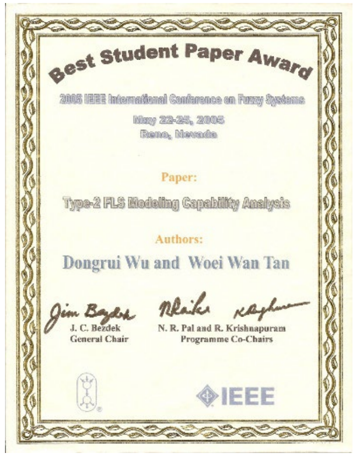
Another Short Video Tutorial

IEEE
Computational
Intelligence
Society

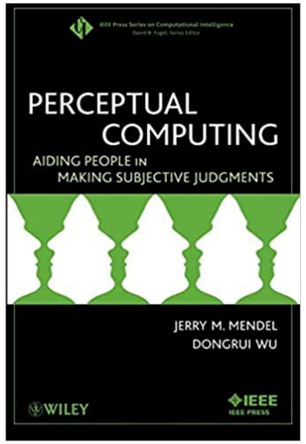
- This first-prize-winning video was sponsored by the IEEE Computational Intelligence Society (CIS) in its 2011 Fuzzy Logic Video Competition
- The IEEE CIS has free rights to use this video as it sees fit
- The video does not incorporate any copyrighted materials

Our Contributions on Fuzzy Systems

1. New optimization techniques for type-1 fuzzy systems
2. Theory and applications of interval type-2 fuzzy systems
3. Applications of fuzzy sets in signal processing and machine learning for brain-computer interfaces



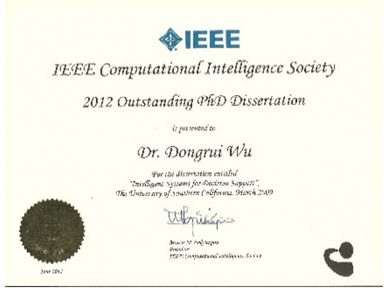
2005 FUZZ-IEEE Best Student Paper Award



2010 IEEE Press Monograph

2014 NAFIPS Early Career Award

2021 Chinese Association of Automation Young Scientist Award



2012 IEEE CIS Outstanding PhD Dissertation Award

2014 IEEE TFS Outstanding Paper Award

2017 IEEE SMC Early Career Award

IEEE TRANSACTIONS ON
FUZZY SYSTEMS
A PUBLICATION OF THE IEEE COMPUTATIONAL INTELLIGENCE SOCIETY

2023 IEEE TFS Editor-in-Chief

First AI-X paper on fuzzy systems: https://fuzzysystem.github.io

Type-2 Fuzzy Sets and Systems

Documentation | All | Examples | Functions | Blocks | Apps | Search R2019b Documentation | Documentation

CONTENTS Close | Trial Software | Product Updates

« Documentation Home

« Fuzzy Logic Toolbox

« Fuzzy Inference System Modeling

Type-2 Fuzzy Inference Systems

ON THIS PAGE

Interval Type-2 Membership Functions

Type-2 Fuzzy Inference Systems

Fuzzy Inference Process for Type-2 Fuzzy Systems

Type-Reduction Methods

See Also

Related Topics

MATLAB®

Type-Reduction Methods

Fuzzy Logic Toolbox software supports four built-in type-reduction methods. These algorithms differ in their initialization methods, assumptions, computational efficiency, and terminating conditions.

To set the type-reduction method for a type-2 fuzzy system, set the `TypeReduction` property of the `mamfistype2` or `sugfistype2` object.

Method	`TypeReduction` property Value	Description
Karnik-Mendel (KM) [2]	`"karnikmendel"`	First type-reduction method developed
Enhanced Karnik-Mendel (EKM) [3]	`"ekm"`	Modification of the Karnik-Mendel algorithm with an improved initialization, modified termination condition, and improved computational efficiency
Iterative algorithm with stop condition (IASC) [4]	`"iasc"`	Iterative improvement to brute force methods
Enhanced iterative algorithm with stop condition (EIASC) [5]	`"eiasc"`	Improved version of the IASC algorithm

In general, the computational efficiency of these methods improve as you move down the table.

You can also use your own custom type-reduction method. For more information, see Build Fuzzy Systems Using Custom Functions.

References

[1] Mendel, J.M., H. Hagras, W.-W. Tan, W.W. Melek, and H. Ying, *Introduction to Type-2 Fuzzy Logic Control*. Hoboken, NJ. Wiley and IEEE Press (2014)

[2] Karnik, N.N. and J.M. Mendel, "Centroid of a type-2 fuzzy set," *Information Sciences*, vol. 132, pp. 195-220. (2001)

[3] Wu, D. and J.M. Mendel, "Enhanced Karnik-Mendel algorithms," *IEEE Transactions on Fuzzy Systems*, vol. 17, pp. 923-934. (2009)

[4] Duran, K., H. Bernal, and M. Melgarejo, "Improved iterative algorithm for computing the generalized centroid of an interval type-2 fuzzy set," *Annual Meeting of the North American Fuzzy Information Processing Society*, pp. 190-194. (2008)

[5] Wu, D. and M. Nie, "Comparison and practical implementations of type-reduction algorithms for type-2 fuzzy sets and systems," *Proceedings of FUZZ-IEEE*, pp. 2131-2138 (2011)

伍冬睿*，曾志刚，莫红，王飞跃，“区间二型模糊集和模糊系统: 综述与展望," 自动化学报, 46(8):1539-1556, 2020.

Design a Fuzzy System

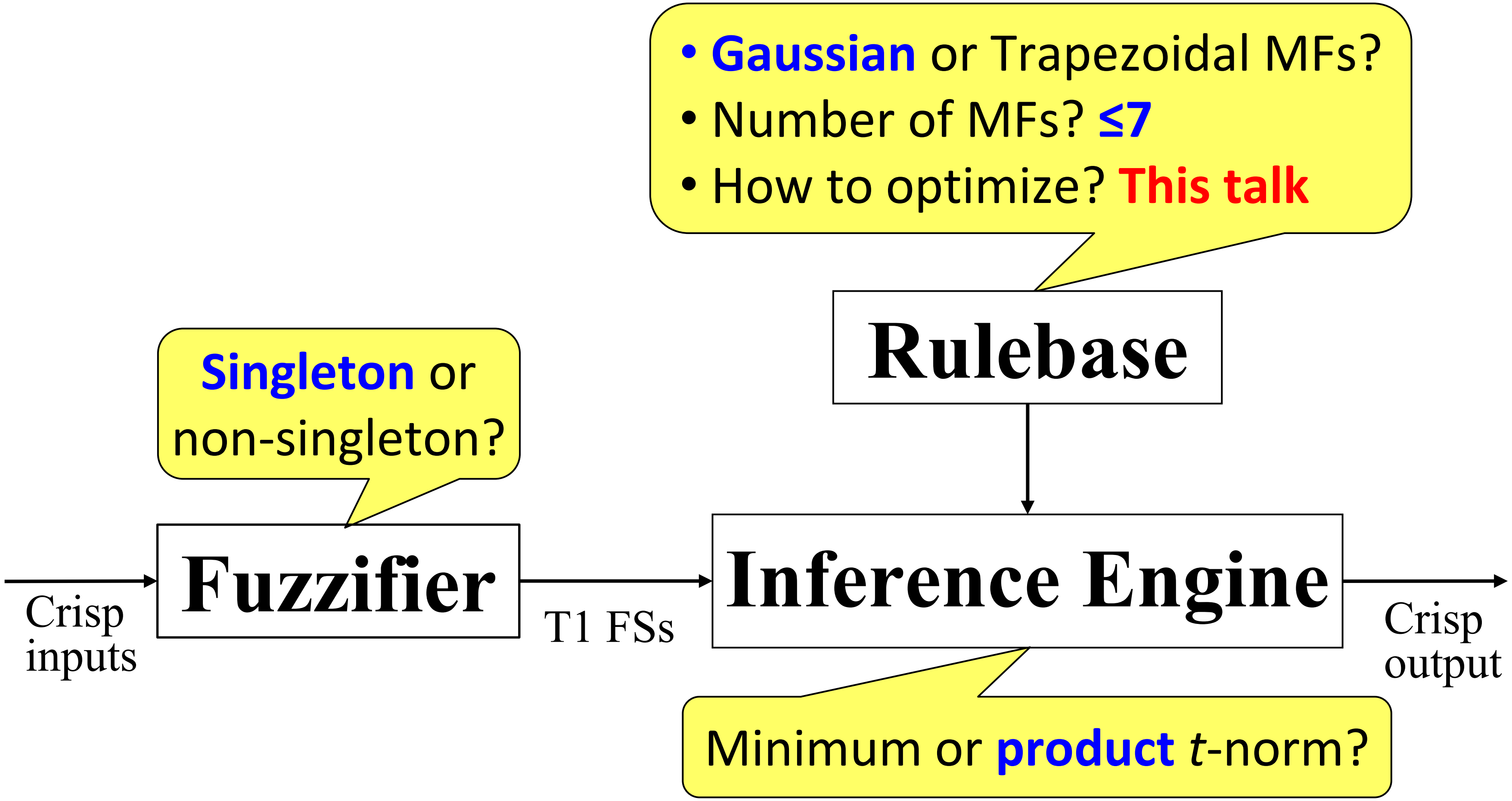
Many questions to be answered in designing a fuzzy system:

- ✓ **Should singleton or non-singleton fuzzifier be used?**
- ✓ **How many MFs should be used for each input?**
- ✓ **Should Gaussian or piecewise linear MFs be used?**
- ✓ **Should Mamdani or TSK inference be used?**
- ✓ **Should minimum or product *t*-norm be used?**
- ✓ **How to optimize the fuzzy system?**

In this talk, we use singleton fuzzification, Gaussian MFs, TSK rules and product *t*-norm, and assume that the user can specify the number of MFs in each input domain.

D. Wu and J. M. Mendel, "Recommendations on designing practical interval type-2 fuzzy systems," *Engineering Applications of Artificial Intelligence*, 95:182–193, 2019.

Design a TSK Fuzzy System



D. Wu and J. M. Mendel, "Recommendations on designing practical interval type-2 fuzzy systems," *Engineering Applications of Artificial Intelligence*, 95:182–193, 2019.

Design a TSK Fuzzy System

Challenges in designing a TSK fuzzy system:

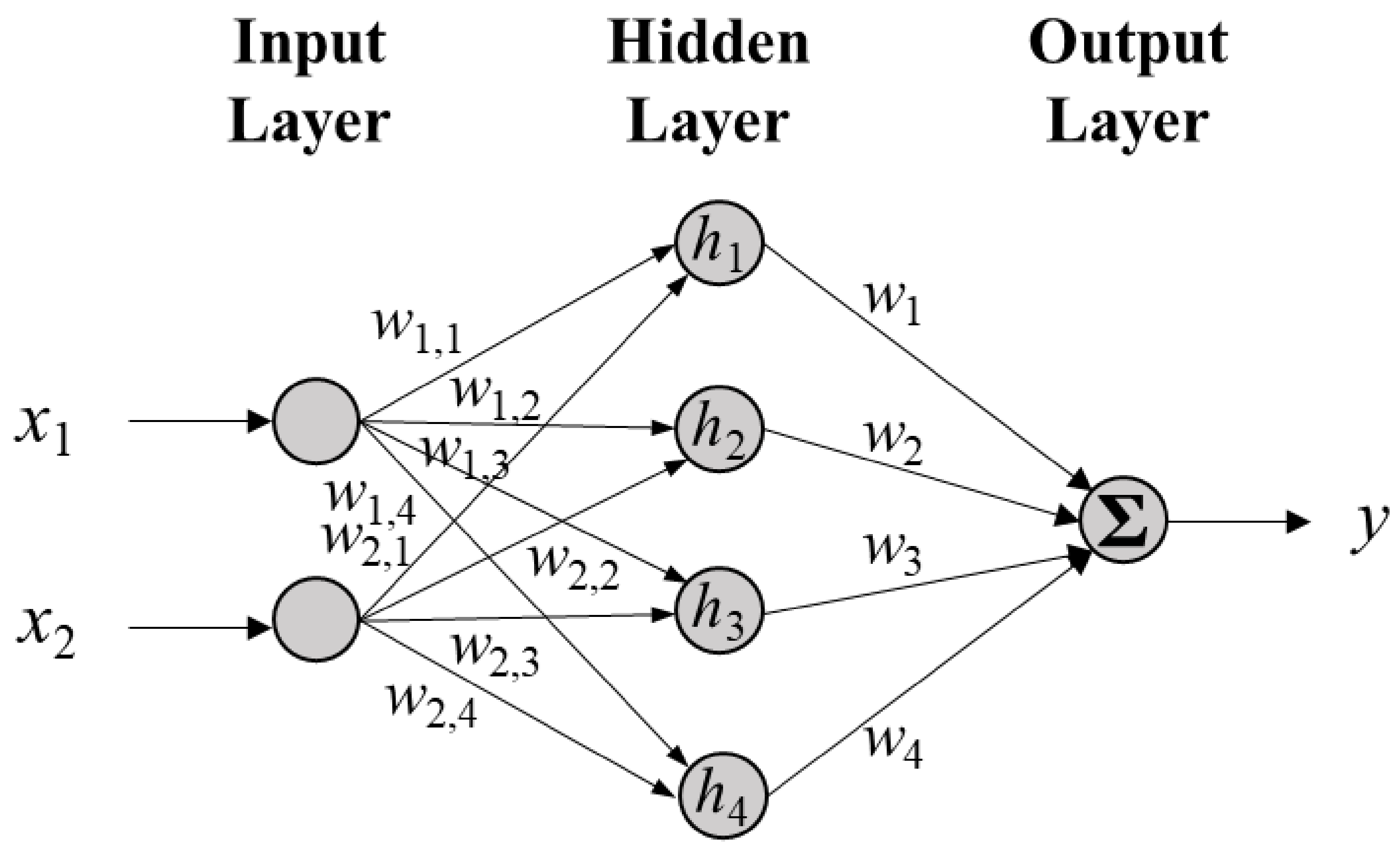
- ✓ **Optimization**. Evolutionary algorithms, gradient descent, and gradient descent plus least squares estimation (ANFIS).
- ✓ **Interpretability**. The interpretability decreases when the number of rules increases, and when each input activates too many rules.
- ✓ **Curse of dimensionality**. When each input has a few fuzzy partitions, the number of rules increases exponentially with the number of inputs. Clustering based initialization also suffers from curse of dimensionality (validity and interpretability).
- ✓ **Generalization**. Regularization can improve generalization, but not extensively explored in training fuzzy systems.

D. Wu, C-T Lin, J. Huang & Z. Zeng, "On the Functional Equivalence of TSK Fuzzy Systems to Neural Networks, Mixture of Experts, CART, and Stacking Ensemble Regression," *IEEE Trans. Fuzzy Systems*, 28(10): 2570-2580, 2020.

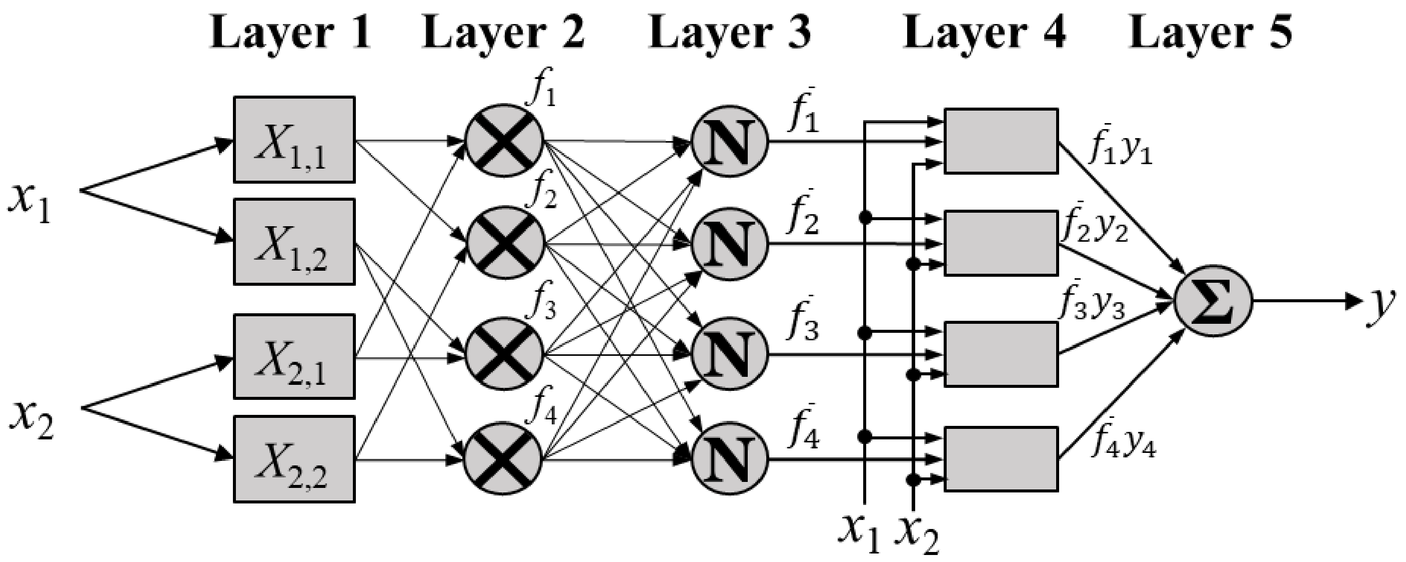
Outline

- Fuzzy Sets
- TSK Fuzzy Systems (FSs)
- **Equivalence between TSK FSs and other Machine Learning Models**
- Optimize TSK FSs for Regression Problems
- Optimize TSK FSs for Classification Problems
- Conclusions

Multi-Layer Perceptron (MLP)



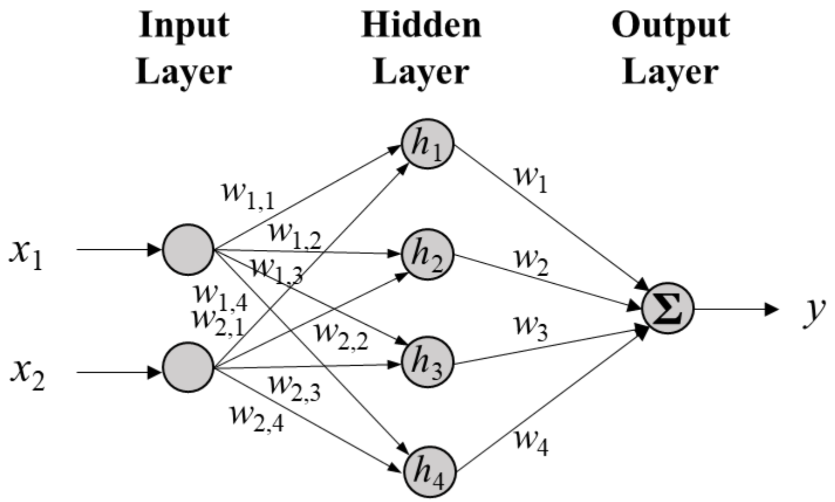
Adaptive-Network-based Fuzzy Inference System (ANFIS)

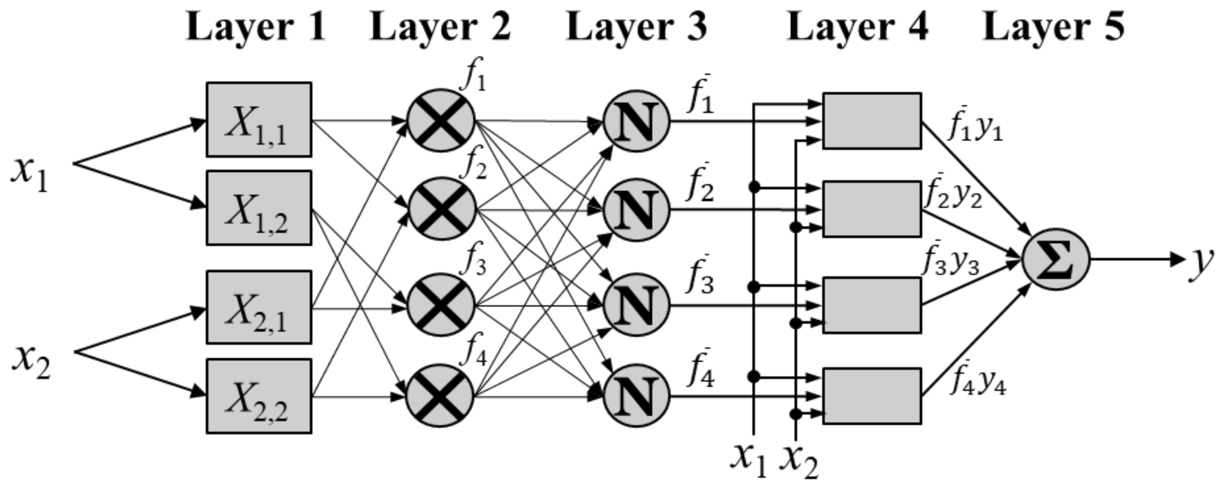


- **Layer 1**: Compute the membership grades.
- **Layer 2**: Compute the firing level of each rule.
- **Layer 3**: Compute the normalized firing levels of the rules.
- **Layer 4**: Multiply each normalized firing level by its corresponding rule consequent.
- **Layer 5**: Compute the output as a summation.

J.S. Jang, ANFIS: adaptive-network-based fuzzy inference system, *IEEE Trans. on Systems, Man, and Cybernetics*, *23*(3):665-685, 1993. (~20000 citations)

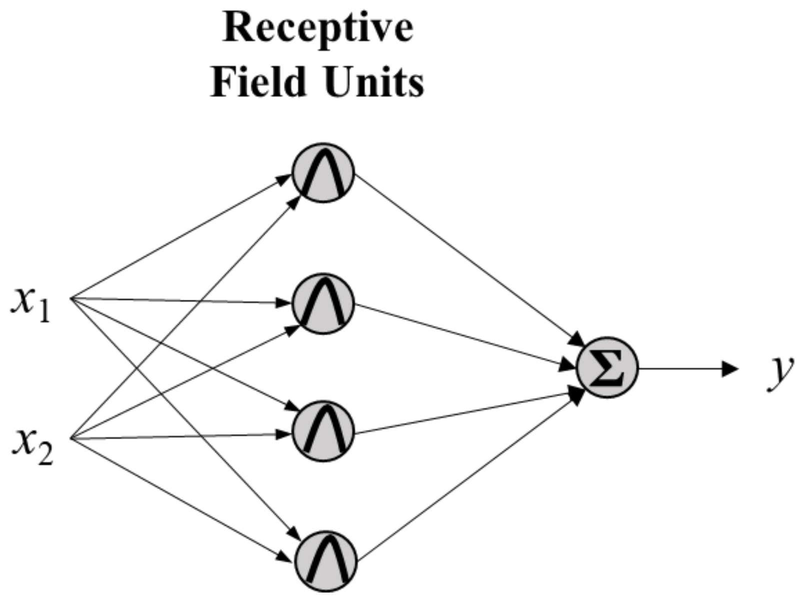
Differences between MLP & ANFIS





1. MLP always uses fully connected layers, whereas ANFIS is not.
2. In MLP, the output of a node in the hidden layer and output layer is always computed by a weighted sum followed by an activation function, whereas there are many different operations in an ANFIS.
3. Layer 4 of the ANFIS also uses **x** as an input (to represent the rule consequents), but usually an MLP does not have such connections. (skip-layer connection in ResNet?)
4. MLP is a black-box model, whereas ANFIS can be expressed by IF-THEN rules, which is easier to interpret and understand.

FS and Radial Basis Function Network (RBFN)



For input $\mathbf{x} = (x_1, x_2)$, the output of the kth ($k = 1, ..., K$) receptive field unit, using a Gaussian response function, is:

$$f_k(\mathbf{x}) = \exp\left(-\frac{(x_1 - m_{k,1})^2 + (x_2 - m_{k,2})^2}{\sigma_k^2}\right)$$

The output of the RBFN is:

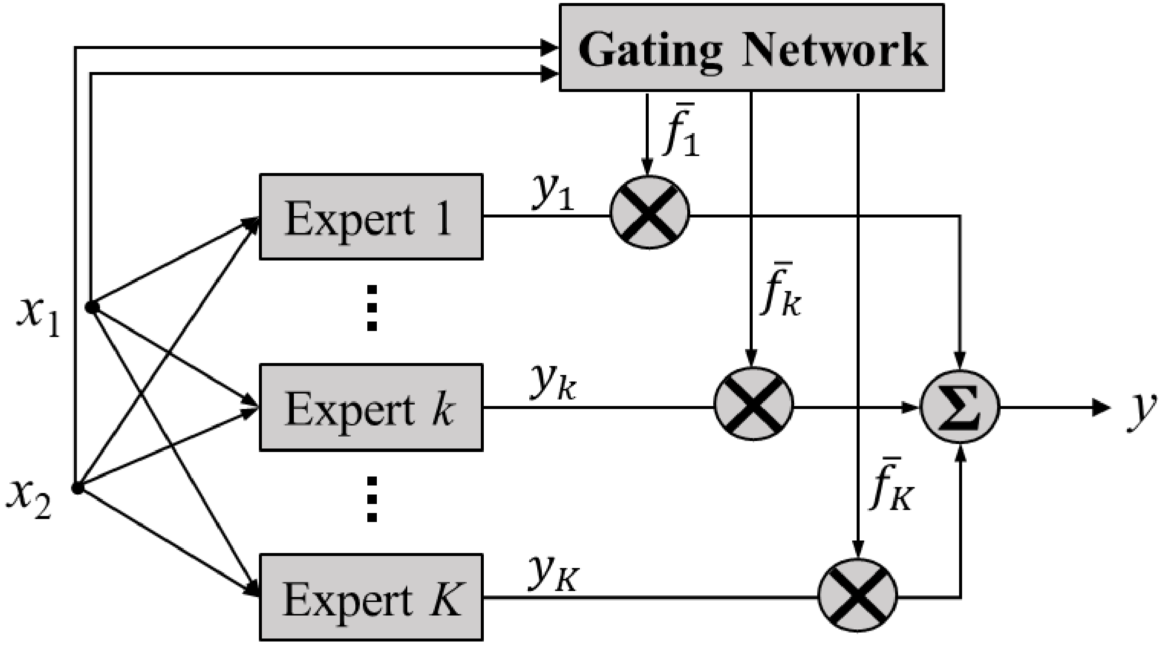
$$y(\mathbf{x}) = \frac{\sum_{k=1}^{K} f_k(\mathbf{x}) \cdot y_k}{\sum_{k=1}^{K} f_k(\mathbf{x})},$$

FS = RBFN, when:

- ✓ Number of receptive field units = Number of fuzzy rules
- ✓ The output of each fuzzy rule is a constant, instead of a function
- ✓ Antecedent MFs of each fuzzy rule: Gaussian with the same variance
- ✓ The product *t*-norm is used
- ✓ The FS and the RBFN use the same method (i.e., either weighted average or weighted sum) to compute the final output

J.-S. Jang and C.-T. Sun, "Functional equivalence between radial basis function networks and fuzzy inference systems," *IEEE Trans. on Neural Networks*, 4(1):156-159, 1993.

FS and Mixture of Experts (MoE)

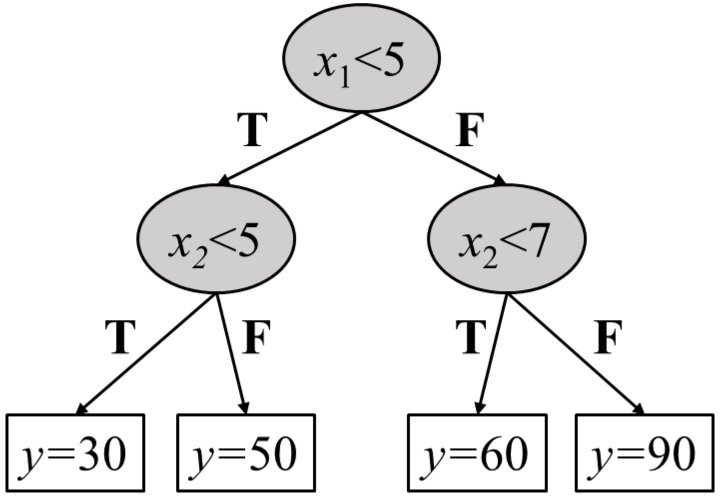


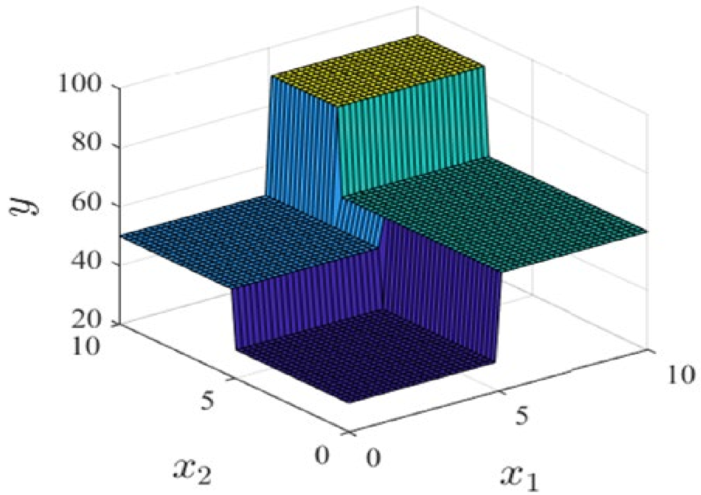
$$y_{ME}(\mathbf{x}) = \sum_{k=1}^{K} \bar{f}_k(\mathbf{x}) y_k(\mathbf{x})$$

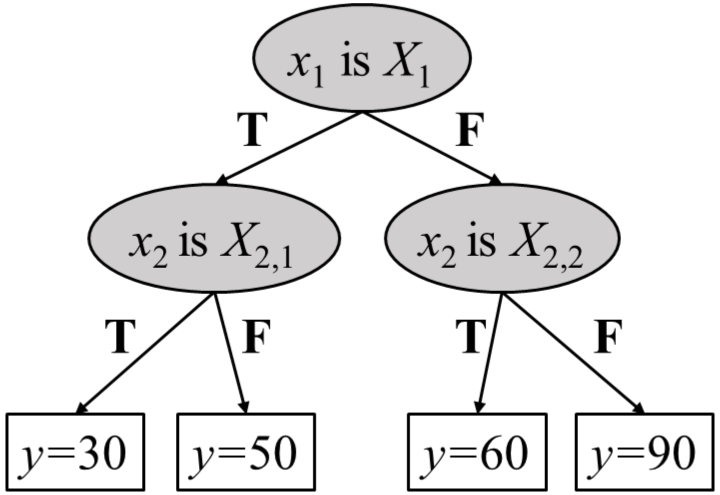
FS = MoE

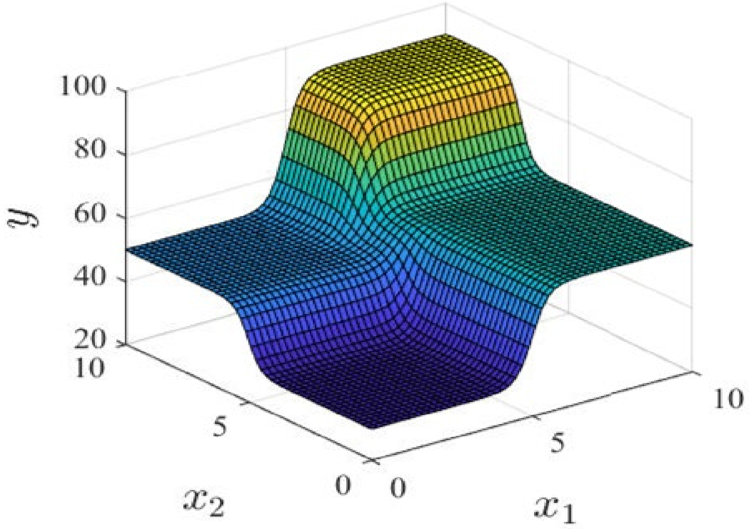
R. A. Jacobs, M. I. Jordan, S. J. Nowlan, and G. E. Hinton, "Adaptive mixtures of local experts," *Neural Computation*, 3(1):79-87, 1991.

FS and Classification and Regression Tree (CART)



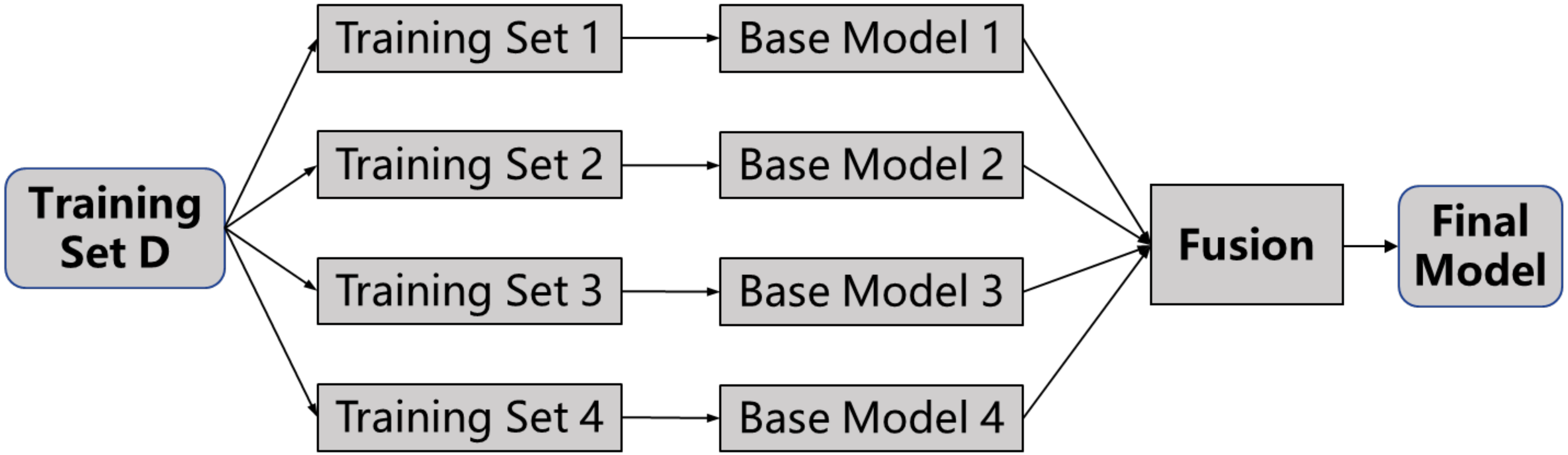






L. Breiman, J. Friedman, C. J. Stone, and R. Olshen, *Classification and Regression Trees*, 1st ed. Routledge, 2017.

FS and Stacking



- A TSK FS for regression can be viewed as a stacking model.
- Each rule consequent is a base regression model, and the rule antecedent MFs determine the weights of the base models in stacking.
- In stacking usually the aggregated output y is a function of y_k, k=1,…,K, only, but in a TSK FS the aggregation function also depends on the input **x**, as the weights are computed from them, and change with them.
- So, **a TSK FS is actually an adaptive stacking regression model**.

Inspirations

- **From neural networks**: Design more efficient training algorithms for TSK fuzzy systems.
- **From MoE**: Achieve a better trade-off between cooperation and competitions of the rules in a TSK fuzzy system.
- **From CART**: Better initialize a TSK fuzzy system for high-dimensional problems.
- **From stacking ensemble regression**: Design better stacking models, and increase the generalization ability of a TSK fuzzy model.

D. Wu, C-T Lin, J. Huang & Z. Zeng, "On the Functional Equivalence of TSK Fuzzy Systems to Neural Networks, Mixture of Experts, CART, and Stacking Ensemble Regression," *IEEE Trans. Fuzzy Systems*, 28(10): 2570-2580, 2020.

Outline

- Fuzzy Sets
- TSK Fuzzy Systems (FSs)
- Equivalence between TSK FSs and other Machine Learning Models
- **Optimize TSK FSs for Regression Problems**
- Optimize TSK FSs for Classification Problems
- Conclusions

Takagi-Sugeno-Kang (TSK) Fuzzy System (FS)

Assume the input $\mathbf{x} = (x_1, ..., x_M)^T \in \mathbb{R}^{M\times 1}$, and the TSK fuzzy system has R rules:

$$\text{Rule}_r : \text{IF } x_1 \text{ is } \boxed{X_{r,1}} \text{ and } \cdots \text{ and } x_M \text{ is } \boxed{X_{r,M}},$$

$$\text{THEN } y_r(\mathbf{x}) = \boxed{b_{r,0}} + \sum_{m=1}^{M} \boxed{b_{r,m}} x_m,$$

where $X_{r,m}$ $(r = 1, ..., R;\ m = 1, ..., M)$ are fuzzy sets, and $b_{r,0}$ and $b_{r,m}$ are consequent parameters.

Big Data

- At least three Vs:
  1. Volume (the size of the data): The number of training examples (N) is very large, and/or the dimensionality of the input (M) is very high.
  2. Velocity (the speed of the data)
  3. Variety (the types of data)
- FSs suffer from the curse of dimensionality, i.e., the number of rules (parameters) increases exponentially with M.
- We assume that the dimensionality can be reduced effectively to just a few, e.g., using PCA.
- We mainly consider how to deal with large N.

Optimize TSK FS

- **Evolutionary algorithms**: Very high memory and computing power requirement on big data.

- **Gradient descent (GD)**: Focus of this talk.

Steps in Optimizing a TSK Fuzzy System

1. Define the objective function
2. Initialize the rules
3. Fine-tune the rules
    - How to handle big data (big size & high dimensionality)?
    - How to speed-up the training?
    - How to improve the generalization performance?

Regularization in the Objective Function

1. **Define objective function**
2. Initialize rules
3. Fine-tune rules
   - Handle big data
   - Speed-up training
   - Improve generalization

ℓ_2 regularized loss function:

$$L = \frac{1}{2}\sum_{n=1}^{N_{bs}}[y_n - y(\mathbf{x}_n)]^2 + \frac{\lambda}{2}\sum_{r=1}^{R}\sum_{m=1}^{M} b_{r,m}^2,$$

where $N_{bs} \in [1, N]$, and $\lambda \geq 0$ is a regularization parameter.

Note that $b_{r,0}$ $(r = 1, ..., R)$ are not regularized.

Semi-Random Initialization of the Rules

1. Define objective function
2. **Initialize rules**
3. Fine-tune rules
   - Handle big data
   - Speed-up training
   - Improve generalization

For $m = 1, \ldots, M$:

1. Compute the minimum and maximum of all $\{x_{n,m}\}_{n=1}^{N}$
2. Initialize the centers of the Gaussian MFs uniformly between the minimum and the maximum
3. Initialize the standard deviation of all Gaussian MFs as the standard deviation of $\{x_{n,m}\}_{n=1}^{N}$
4. Initialize the rule consequent parameters as 0

Mini-Batch Gradient Descent (MBGD) for Big Data

1. Define objective function
2. Initialize rules
3. Fine-tune rules
   - **Handle big data**
   - Speed-up training
   - Improve generalization

Randomly sample $N_{bs} \in [1, N]$ training examples $\Rightarrow$ Compute the gradients from them $\Rightarrow$ update the parameters of the TSK fuzzy system.

Let $\boldsymbol{\theta}_k$ be the model parameter vector in the kth iteration, and $\partial L / \partial \boldsymbol{\theta}_k$ be their first-order gradients. Then, the update rule is:

$$\boldsymbol{\theta}_k = \boldsymbol{\theta}_{k-1} - \alpha \frac{\partial L}{\partial \boldsymbol{\theta}_{k-1}},$$

where $\alpha > 0$ is the learning rate.

When $N_{bs} = 1$, MBGD degrades to stochastic GD.

When $N_{bs} = N$, it becomes batch GD.

Adam/AdaBound for Speeding-up the Training

1. Define objective function
2. Initialize rules
3. Fine-tune rules
   - Handle big data
   - **Speed-up training**
   - Improve generalization

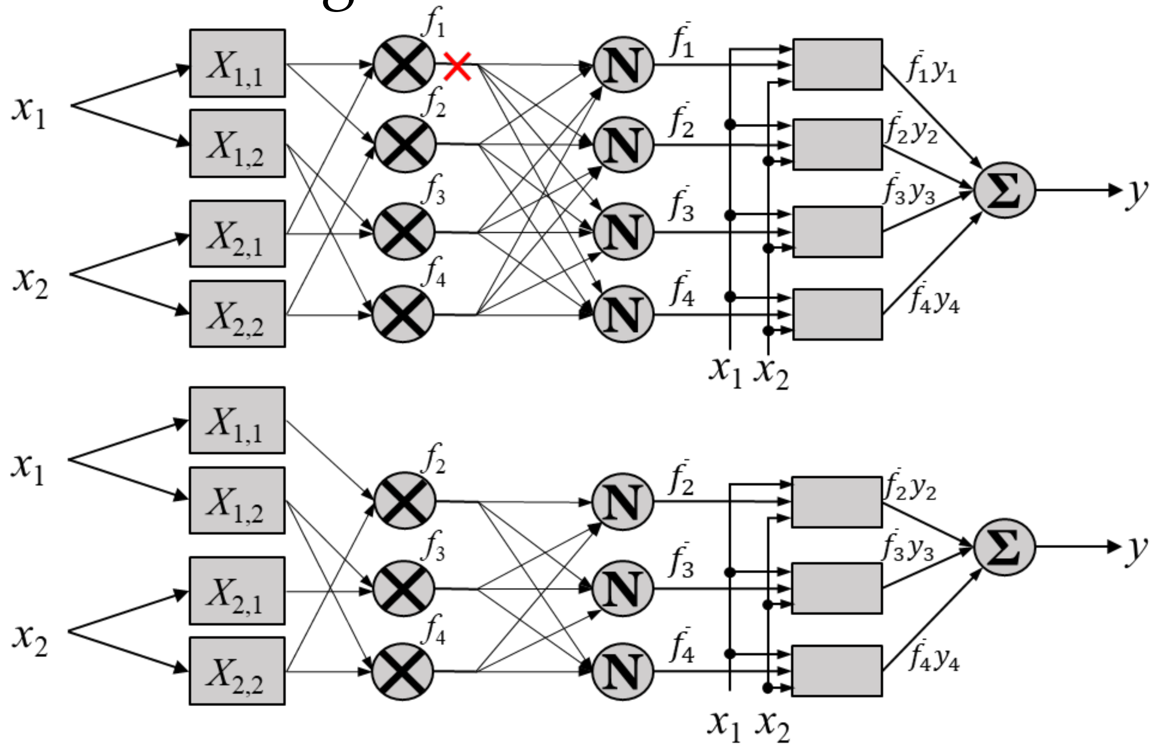
- **Adam** in NN training computes an individualized adaptive learning rate for each different model parameter from the estimates of the 1st and 2nd moments of the gradient.
- **AdaBound** bounds the individualized adaptive learning rate from the upper and the lower, so that extremely large or small learning rate cannot occur. Additionally, the bounds become tighter as the number of iterations increases, which forces the learning rates to approach a constant (as in stochastic GD).

Kingma, D.P. and Ba, J., Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980, 2014.
L. Luo, Y. Xiong, Y. Liu, and X. Sun, “Adaptive gradient methods with dynamic bound of learning rate,” ICLR 2019.

DropRule for Better Generalization

1. Define objective function
2. Initialize rules
3. Fine-tune rules
    - Handle big data
    - Speed-up training
    - **Improve generalization**

- **DropOut** randomly discards some neurons and their connections during the training of NNs.
- **DropRule** randomly discards a small number of rules during the training of FSs, but uses all rules when the training is done.



N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: A simple way to prevent neural networks from overfitting," *Journal of Machine Learning Research*, 15(1):1929–1958, 2014.

MBGD-RDA for Big Data

Algorithm 1: The mini-batch gradient descent with regularization, DropRule and AdaBound (MBGD-RDA) algorithm for TSK fuzzy system optimization. Typical values of some hyper-parameters are: $\beta_1 = 0.9$, $\beta_2 = 0.999$, $\epsilon = 10^{-8}$.

Input: N labeled training examples $\{\mathbf{x}_n, y_n\}_{n=1}^N$, where $\mathbf{x}_n = (x_{n,1}, ..., x_{n,M})^T \in \mathbb{R}^{M\times 1}$;
$L(\boldsymbol{\theta})$, the loss function for the TSK fuzzy model parameter vector $\boldsymbol{\theta}$;
M_m, the number of Gaussian MFs in the mth input domain;
$N_{bs} \in [1, N]$, the mini-batch size;
K, the maximum number of training iterations;
$P \in [0.5, 1]$, the DropRule rate;
α, the initial learning rate (step size);
$\beta_1, \beta_2 \in [0, 1)$, exponential decay rates for the moment estimates;
ϵ, a small positive number;
$l(k)$ and $u(k)$, the lower and upper bound functions in AdaBound;
Output: The final $\boldsymbol{\theta}$.

```
// Initialization
for m = 1, ..., M do
    Compute the minimum and maximum of all {x_{n,m}}_{n=1}^N;
    Initialize the center of the M_m Gaussian MFs uniformly between the minimum and the maximum;
    Initialize the standard deviation of all M_m Gaussian MFs as the standard deviation of {x_{n,m}}_{n=1}^N;
end
Initialize the consequent parameters of all R rules as 0;
θ_0 is the concatenation of all Gaussian MF centers, standard deviations, and rule consequent parameters;
// Update θ
m_0 = 0; v_0 = 0;
for k = 1, ..., K do
    // MBGD
    Randomly select N_bs training examples;
    for n = 1, ..., N_bs do
        for r = 1, ..., R do
            Compute f_r(x_n), the firing level of x_n on Rule_r;
            // DropRule
            Generate p, a uniformly distributed random number in [0, 1];
            if p > P then
                f_r(x_n) = 0;
            end
        end
        Compute y(x_n), the TSK fuzzy system output for x_n, by (4);
    end
    // Compute the gradients
    g_k = ∂L(θ_{k-1}) / ∂θ_{k-1};
    // AdaBound
    m_k = β_1 m_{k-1} + (1 - β_1) g_k;   v_k = β_2 v_{k-1} + (1 - β_2) g_k^2;
    m̂_k = m_k / (1 - β_1^k);   v̂_k = v_k / (1 - β_2^k);
    α̂ = max[ l(k), min( u(k), α / (sqrt(v̂_t) + ε) ) ];
    θ_k = θ_{k-1} - α̂ ⊙ m̂_k;
end
Return θ_K
```

Key formulas:

$$\mathbf{g}_k = \frac{\partial L(\boldsymbol{\theta}_{k-1})}{\partial \boldsymbol{\theta}_{k-1}};$$

$$\mathbf{m}_k = \beta_1 \boldsymbol{m}_{k-1} + (1-\beta_1)\boldsymbol{g}_k; \quad \boldsymbol{v}_k = \beta_2 \boldsymbol{v}_{k-1} + (1-\beta_2)\boldsymbol{g}_k^2;$$

$$\hat{\boldsymbol{m}}_k = \frac{\boldsymbol{m}_k}{1-\beta_1^k}; \quad \hat{\boldsymbol{v}}_k = \frac{\boldsymbol{v}_k}{1-\beta_2^k};$$

$$\hat{\boldsymbol{\alpha}} = \max\left[l(k), \min\left(u(k), \frac{\alpha}{\sqrt{\hat{\boldsymbol{v}}_t}+\epsilon}\right)\right];$$

$$\boldsymbol{\theta}_k = \boldsymbol{\theta}_{k-1} - \hat{\boldsymbol{\alpha}} \odot \hat{\boldsymbol{m}}_k;$$

Experiments: Datasets

TABLE II
SUMMARY OF THE 10 REGRESSION DATASETS.

Dataset	Source	No. of examples	No. of raw features	No. of numerical features	No. of used features	No. of TSK model parameters
PM10[1]	StatLib	500	7	7	5	212
NO2[1]	StatLib	500	7	7	5	212
Housing[2]	UCI	506	13	13	5	212
Concrete[3]	UCI	1,030	8	8	5	212
Airfoil[4]	UCI	1,503	5	5	5	212
Wine-Red[5]	UCI	1,599	11	11	5	212
Abalone[6]	UCI	4,177	8	7	5	212
Wine-White[5]	UCI	4,898	11	11	5	212
PowerPlant[7]	UCI	9,568	4	4	4	96
Protein[8]	UCI	45,730	9	9	5	212

[1] http://lib.stat.cmu.edu/datasets/
[2] https://archive.ics.uci.edu/ml/machine-learning-databases/housing/
[3] https://archive.ics.uci.edu/ml/datasets/Concrete+Compressive+Strength
[4] https://archive.ics.uci.edu/ml/datasets/Airfoil+Self-Noise
[5] https://archive.ics.uci.edu/ml/datasets/Wine+Quality
[6] https://archive.ics.uci.edu/ml/datasets/Abalone
[7] https://archive.ics.uci.edu/ml/datasets/Combined+Cycle+Power+Plant
[8] https://archive.ics.uci.edu/ml/datasets/Physicochemical+Properties+of+Protein+Tertiary+Structure

Results: RMSEs

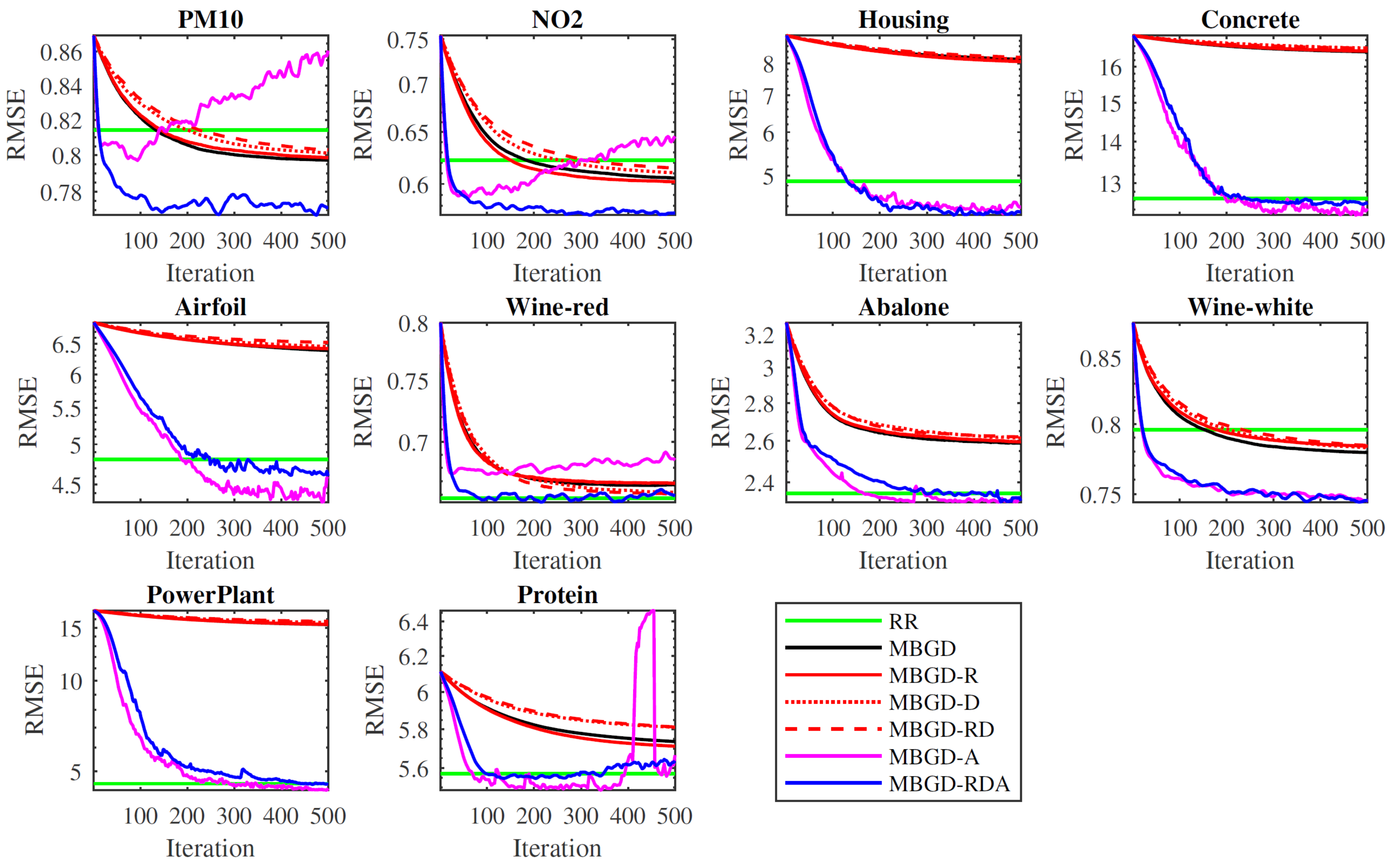


Fig. 2. The average test RMSEs of the seven algorithms on the 10 datasets.

Improvements over MBGD

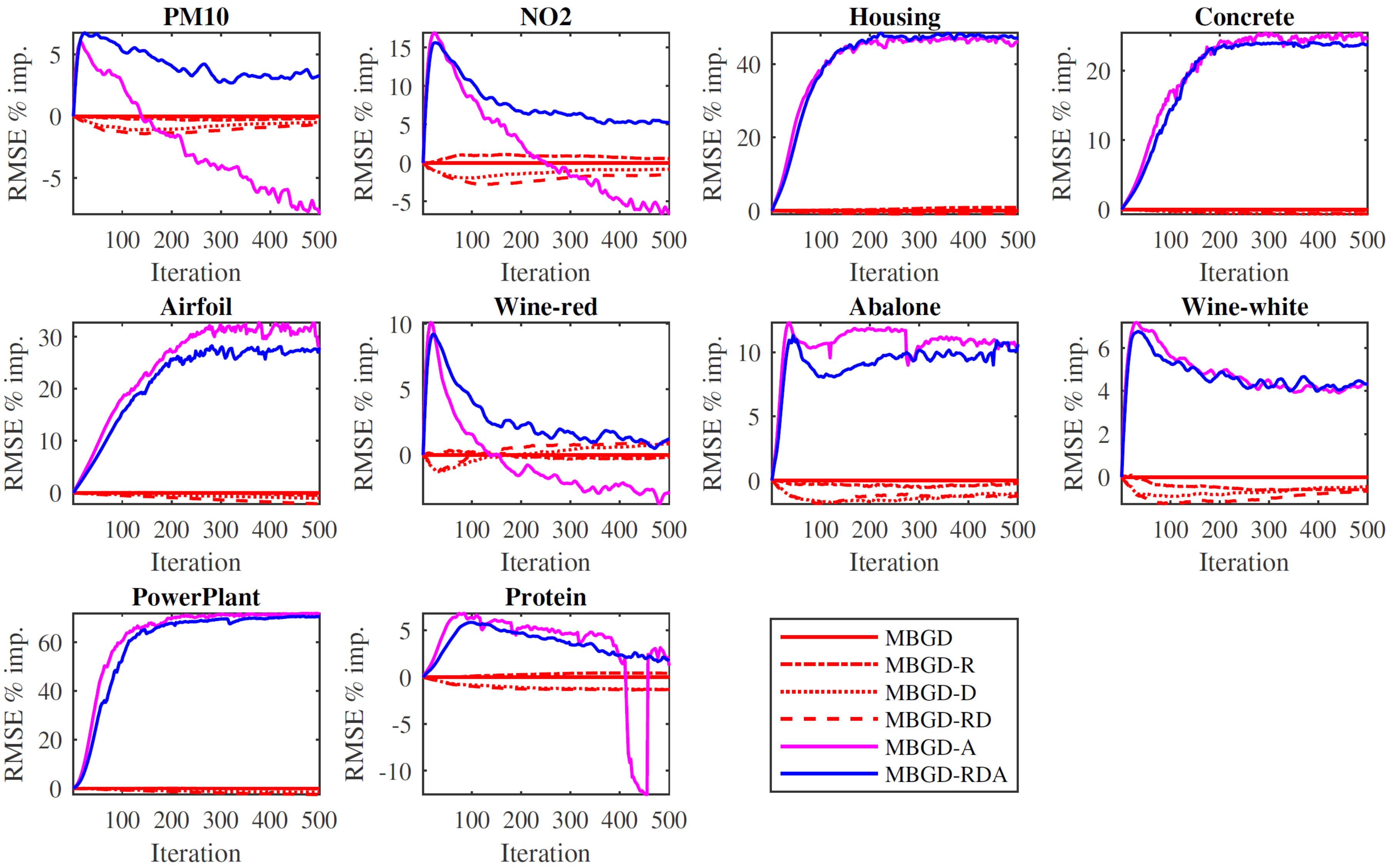
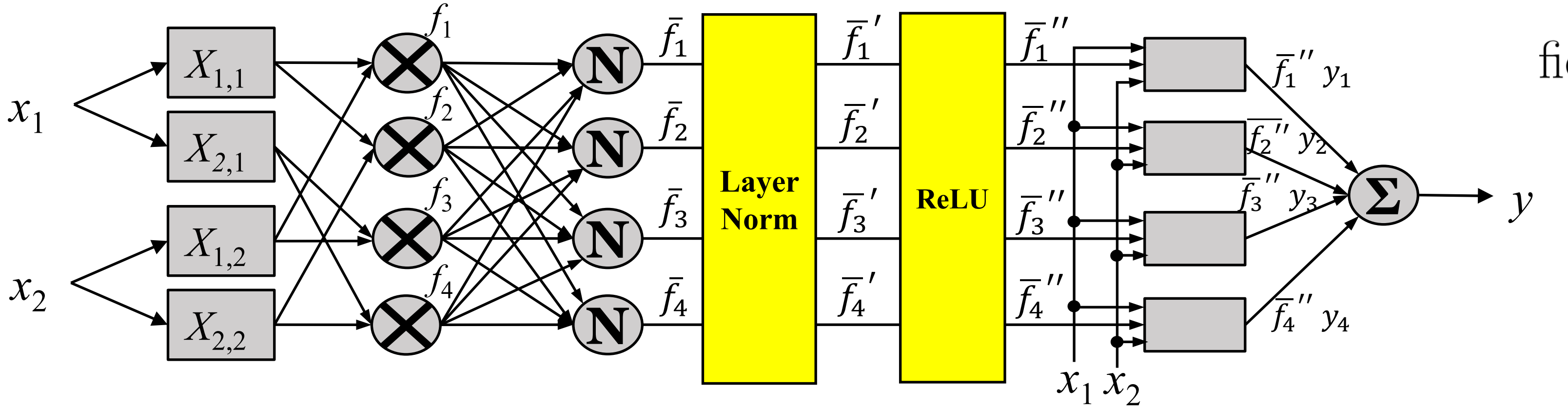


Fig. 3. Percentage improvement of the test RMSEs of MBGD-R, MBGD-D, MBGD-RD, MBGD-A and MBGD-RDA over MBGD.

High-dimensional TSK with Layer Normalization and Rectified Linear Unit (HTSK-LN-ReLU)



High-dimensional TSK (HTSK): Defuzzification using

$$\bar{f}_r(\boldsymbol{x}_i) = \frac{\exp(-Z_r')}{\sum_{r=1}^R \exp(-Z_r')},$$

$$Z_r' = \frac{1}{D}\sum_{d=1}^{D}\frac{(x_{i,d}-m_{r,d})^2}{\sigma_{r,d}^2},$$

Layer Normalization (LN) computes the sample-wise mean and standard deviation of all neurons in a specific layer, and scales the neuron outputs $\boldsymbol{z}$ to:

$$\mathrm{LN}(\boldsymbol{z}) = \boldsymbol{\gamma}\frac{\boldsymbol{z}-\tilde{\mu}}{\sqrt{\tilde{\sigma}^2+\epsilon}} + \boldsymbol{\beta},$$

where $\tilde{\mu}$ and $\tilde{\sigma}$ are respectively the mean and standard deviation of all neuron outputs in that layer, and $\boldsymbol{\gamma}$ and $\boldsymbol{\beta}$ are parameters to be tuned.

We add a ReLU function after HTSK-LN to filter out negative NFLs:

$$\overline{\boldsymbol{f}}''(\boldsymbol{x}_i) = \max\left(0, \overline{\boldsymbol{f}}'(\boldsymbol{x}_i)\right) = \max\left(0, LN(\overline{\boldsymbol{f}}(\boldsymbol{x}_i))\right).$$

Y. Cui, D. Wu*, Y. Xu and R. Peng, "Layer Normalization for TSK Fuzzy System Optimization in Regression Problems," *IEEE Trans. on Fuzzy Systems*, 2023.

Experiments: Datasets

	Abbr.	Train size	Test size	Features	Comment
Scikit-digits	SD	1,258	539	64	
Space GA	SG	2,174	933	6	
Abalone	ABA	2,923	1,254	8	
Park Motor UPDRS	PM	4,112	1,763	16	
Puma 32h	PUM	5,734	2,458	32	
Power Plant	PP	6,697	2,871	4	
Naval	NAV	8,353	3,581	16	
UTK Face	UTK	16,595	7,113	28	Resnet50 + PCA
Steel Industry	SI	24,528	10,512	9	
Diamonds	DIA	27,449	11,764	10	
Microsoft	MIC	34,903	14,959	136	Removed outliers
Year Prediction MSD	YP	36,073	15,461	90	Down-sampled

Results: RMSEs

		Ridge	SVR	CART	RF	XGBoost	MLP	HTSK	HTSK-ConsBN	HTSK-ConsBN-UR	HTSK-LN	HTSK-LN-ReLU
SD	Mean	0.6658	0.6787	0.6751	0.4084	0.3805	**0.2815**	0.4332	0.3091	0.2996	0.3122	0.2979
	STD	0.0000	0.0000	0.0000	0.0068	0.0123	0.0148	0.1743	0.0483	0.0230	0.0326	0.0077
SG	Mean	0.6060	0.6077	0.6715	0.5622	0.5522	0.4893	0.4985	0.4958	0.5108	0.4795	**0.4781**
	STD	0.0000	0.0000	0.0000	0.0028	0.0090	0.0120	0.0110	0.0103	0.0143	0.0092	0.0114
ABA	Mean	0.6695	0.6892	0.7176	0.6683	0.6718	**0.6359**	0.6646	0.6589	0.6546	0.6520	0.6560
	STD	0.0000	0.0000	0.0000	0.0033	0.0085	0.0083	0.0085	0.0080	0.0081	0.0074	0.0062
PM	Mean	0.9763	0.9731	0.9216	0.8162	0.8561	**0.7856**	0.8289	0.8245	0.8252	0.8276	0.8257
	STD	0.0000	0.0000	0.0000	0.0046	0.0061	0.0183	0.0081	0.0075	0.0096	0.0137	0.0183
PUM	Mean	0.8951	0.8993	0.3242	0.2641	0.2980	0.2507	0.2248	**0.1976**	0.1998	0.2121	0.2145
	STD	0.0000	0.0000	0.0000	0.0015	0.0235	0.0028	0.0049	0.0023	0.0041	0.0046	0.0049
PP	Mean	0.2619	0.2620	0.2380	0.1928	**0.1893**	0.2315	0.2230	0.2240	0.2216	0.2240	0.2211
	STD	0.0000	0.0000	0.0000	0.0009	0.0062	0.0015	0.0019	0.0014	0.0010	0.0013	0.0015
NAV	Mean	0.3938	0.4263	0.1039	0.0734	0.0881	0.1070	0.0760	0.0787	0.0597	0.0375	**0.0313**
	STD	0.0000	0.0000	0.0000	0.0031	0.0032	0.0142	0.0435	0.0221	0.0040	0.0041	0.0034
UTK	Mean	0.8831	0.8929	0.9282	0.8433	0.8531	**0.8010**	0.8099	0.8108	0.8057	0.8027	0.8074
	STD	0.0000	0.0000	0.0000	0.0021	0.0026	0.0064	0.0037	0.0038	0.0027	0.0030	0.0019
SI	Mean	0.1400	0.1419	0.0467	0.0335	0.0384	0.0537	0.0355	0.0429	0.0438	0.0298	**0.0287**
	STD	0.0000	0.0000	0.0000	0.0007	0.0048	0.0016	0.0023	0.0024	0.0022	0.0014	0.0013
DIA	Mean	0.2388	0.2479	0.0364	**0.0199**	0.0302	0.0636	0.0372	0.0380	0.0377	0.0364	0.0350
	STD	0.0000	0.0000	0.0000	0.0006	0.0106	0.0047	0.0025	0.0032	0.0078	0.0055	0.0051
MIC	Mean	0.9711	1.2943	0.9408	**0.9177**	0.9180	0.9711	1.0621	1.1274	1.0525	0.9435	0.9471
	STD	0.0000	0.0000	0.0000	0.0006	0.0018	0.0250	0.0964	0.2867	0.1212	0.0219	0.0160
YP	Mean	0.8808	0.9105	0.9247	0.8788	0.8577	**0.8310**	0.8454	0.8362	0.8360	0.8406	0.8407
	STD	0.0000	0.0000	0.0000	0.0008	0.0011	0.0039	0.0055	0.0036	0.0032	0.0042	0.0029
Average		0.6319	0.6687	0.5441	0.4732	0.4778	0.4585	0.4783	0.4703	0.4623	0.4498	**0.4486**
Average Rank		9.25	10.58	8.67	4.75	5.83	4.58	5.92	5.17	4.33	3.50	**3.17**

Outline

- Fuzzy Sets
- TSK Fuzzy Systems (FSs)
- Equivalence between TSK FSs and other Machine Learning Models
- Optimize TSK FSs for Regression Problems
- **Optimize TSK FSs for Classification Problems**
- Conclusions

Uniform Regularization in the Objective Function

1. **Define objective function**
2. Initialize rules
3. Fine-tune rules
   - Handle big data
   - Speed-up training
   - Improve generalization

For each mini-batch with M training samples,

$$\mathcal{L} = \ell + \alpha\ell_2 + \lambda\sum_{r=1}^{R}\left(\frac{1}{M}\sum_{n=1}^{M}\overline{f}_r(\boldsymbol{x}_n) - \frac{1}{R}\right)^2$$

where:

- ℓ is the cross-entropy loss between the estimated class probabilities [obtained by applying a *softmax* operation to $\boldsymbol{y}(\boldsymbol{x})$] and the true class probabilities
- ℓ_2 the L2 regularization of the rule consequent parameters
- The last term forces the rules to be fired at similar degrees in the input space, so that each rule contributes about equally to the output. It reduces the "rich get richer" problem.

k-Means Clustering Initialization of the Rules

1. Define objective function
2. **Initialize rules**
3. Fine-tune rules
   - Handle big data
   - Speed-up training
   - Improve generalization

$$\text{Rule}_r : \text{IF } x_1 \text{ is } X_{r,1} \text{ and } \cdots \text{ and } x_M \text{ is } X_{r,M},$$

$$\text{THEN } y_r(\mathbf{x}) = b_{r,0} + \sum_{m=1}^{M} b_{r,m} x_m,$$

- **Antecedents**: k-means clustering to initialize the means of the Gaussian MFs, and random standard deviation in $N(1,0.2)$.
- **Consequents**: bias = 0, coefficients $U(-1,1)$.

Mini-Batch Gradient Descent (MBGD) for Big Data

1. Define objective function
2. Initialize rules
3. Fine-tune rules
   - **Handle big data**
   - Speed-up training
   - Improve generalization

Randomly sample $N_{bs} \in [1, N]$ training examples $\Rightarrow$ Compute the gradients from them $\Rightarrow$ update the parameters of the TSK fuzzy system.

Let $\boldsymbol{\theta}_k$ be the model parameter vector in the kth iteration, and $\partial L / \partial \boldsymbol{\theta}_k$ be their first-order gradients. Then, the update rule is:

$$\boldsymbol{\theta}_k = \boldsymbol{\theta}_{k-1} - \alpha \frac{\partial L}{\partial \boldsymbol{\theta}_{k-1}},$$

where $\alpha > 0$ is the learning rate.

When $N_{bs} = 1$, MBGD degrades to stochastic GD.

When $N_{bs} = N$, it becomes batch GD.

Adam/AdaBound for Speeding-up the Training

1. Define objective function
2. Initialize rules
3. Fine-tune rules
   - Handle big data
   - **Speed-up training**
   - Improve generalization

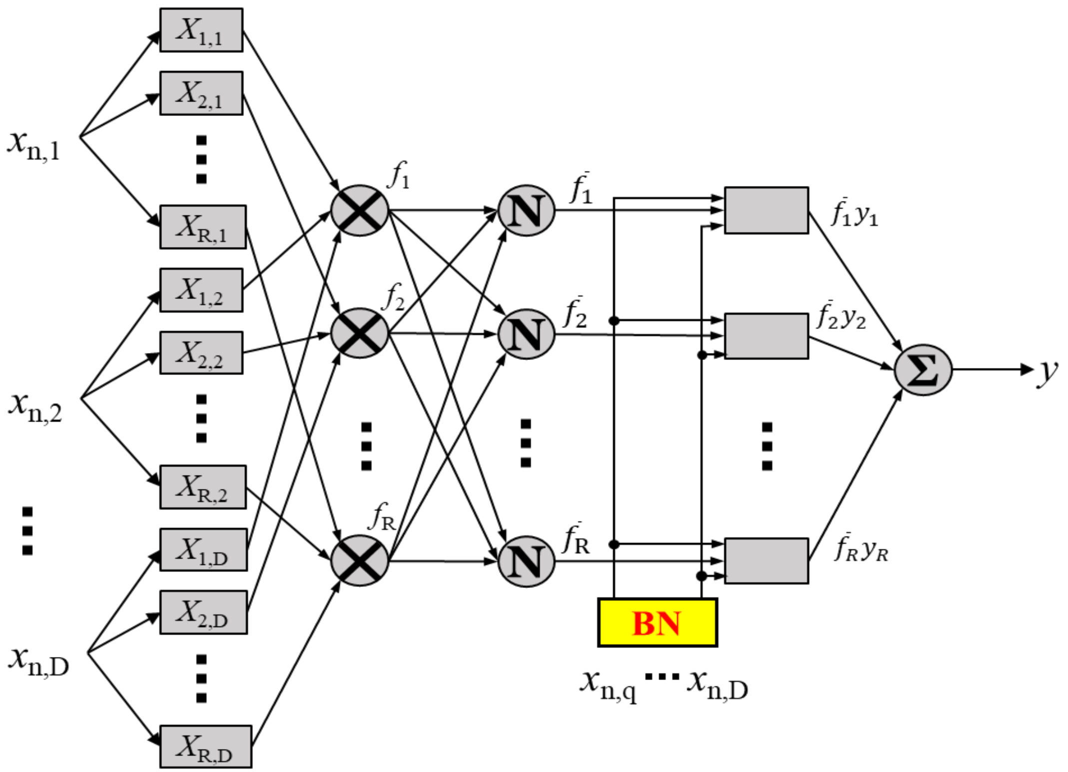
- **Adam** in NN training computes an individualized adaptive learning rate for each different model parameter from the estimates of the 1st and 2nd moments of the gradient.
- **AdaBound** bounds the individualized adaptive learning rate from the upper and the lower, so that extremely large or small learning rate cannot occur. Additionally, the bounds become tighter as the number of iterations increases, which forces the learning rates to approach a constant (as in stochastic GD).

Kingma, D.P. and Ba, J., Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980, 2014.
L. Luo, Y. Xiong, Y. Liu, and X. Sun, "Adaptive gradient methods with dynamic bound of learning rate," ICLR 2019.

Batch Normalization (BN) for Better Generalization

1. Define objective function
2. Initialize rules
3. Fine-tune rules
   - Handle big data
   - Speed-up training
   - **Improve generalization**

BN normalizes the data distribution in each mini-batch to accelerate the training & improve the generalization.



S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," ICML 2015.

Experiments: Datasets

SUMMARY OF THE 12 DATASETS.

Index	Dataset	No. of Samples	No. of Features	No. of Classes
1	Vehicle[1]	846	18	4
2	Biodeg[2]	1,055	41	2
3	DRD[3]	1151	19	2
4	Yeast[4]	1,484	8	10
5	Steel[5]	1,941	27	7
6	IS[6]	2,310	19	7
7	Abalone[7]	4,177	10	3
8	Waveform21[8]	5,000	21	3
9	Page-blocks[9]	5,473	10	5
10	Satellite[10]	6,435	36	6
11	Clave[11]	10,798	16	4
12	MAGIC[12]	19,020	10	2

Experimental Results: Classification Accuracy

AVERAGE BCAS OF THE NINE ALGORITHMS ON THE 12 DATASETS.

Dataset	CART	RF	JRip	PART	TSK-FCM-LSE	TSK-MBGD	TSK-MBGD-BN	TSK-MBGD-UR	TSK-MBGD-UR-BN
Vehicle	0.6936	0.744	0.6939	0.7131	0.7443	0.7010	0.7380	0.7127	**0.7930**
Biodeg	0.7973	0.8306	0.7899	0.8122	0.8205	0.8368	0.8318	0.8390	**0.8439**
DRD	0.634	0.6624	0.6227	0.6422	**0.6845**	0.6642	0.6634	0.6717	0.6729
Yeast	0.3998	0.4867	0.5203	0.4889	0.5102	0.4951	0.5184	0.4946	**0.5332**
Steel	0.7005	0.6937	0.7129	0.7267	0.6319	0.5933	0.7258	0.7245	**0.7515**
IS	0.932	0.9529	0.9481	**0.9607**	0.9571	0.5762	0.7557	0.8559	0.9501
Abalone	0.5319	0.5362	0.5371	0.5280	0.5402	0.4567	0.5236	0.4791	**0.5402**
Waveform21	0.7637	0.8365	0.7905	0.7844	**0.8645**	0.6784	0.8003	0.8362	0.8233
Page-blocks	0.7986	0.7385	**0.8192**	0.8162	0.6003	0.5129	0.5609	0.6033	0.671
Satellite	0.8204	0.8480	0.8308	0.834	0.8558	0.4337	0.7651	0.8679	**0.8700**
Clave	0.4701	0.4878	0.4985	**0.6507**	0.4825	0.5876	0.6468	0.6374	0.6421
MAGIC	0.8058	0.8108	0.8052	0.8135	0.7886	0.6325	0.7128	**0.8225**	0.7934
Average	0.6956	0.7190	0.714	0.7309	0.7067	0.5974	0.6869	0.7120	**0.7404**

Experimental Results: Rank of Classification Accuracy

BCA RANKS OF THE NINE ALGORITHMS ON THE 12 DATASETS.

Dataset	CART	RF	JRip	PART	TSK-FCM-LSE	TSK-MBGD	TSK-MBGD-BN	TSK-MBGD-UR	TSK-MBGD-UR-BN
Vehicle	9	3	8	5	2	7	4	6	1
Biodeg	8	5	9	7	6	3	4	2	1
DRD	8	6	9	7	1	4	5	3	2
Yeast	9	8	2	7	4	5	3	6	1
Steel	6	7	5	2	8	9	3	4	1
IS	6	3	5	1	2	9	8	7	4
Abalone	5	4	3	6	1	9	7	8	2
Waveform21	8	2	6	7	1	9	5	3	4
Page-blocks	3	4	1	2	7	9	8	6	5
Satellite	7	4	6	5	3	9	8	2	1
Clave	9	7	6	1	8	5	2	4	3
MAGIC	4	3	5	2	7	9	8	1	6
Average	6.8	4.7	5.4	4.3	4.2	7.3	5.4	4.3	2.6

Outline

- Fuzzy Sets
- TSK Fuzzy Systems (FSs)
- Equivalence between TSK FSs and other Machine Learning Models
- Optimize TSK FSs for Regression Problems
- Optimize TSK FSs for Classification Problems
- **Conclusions**

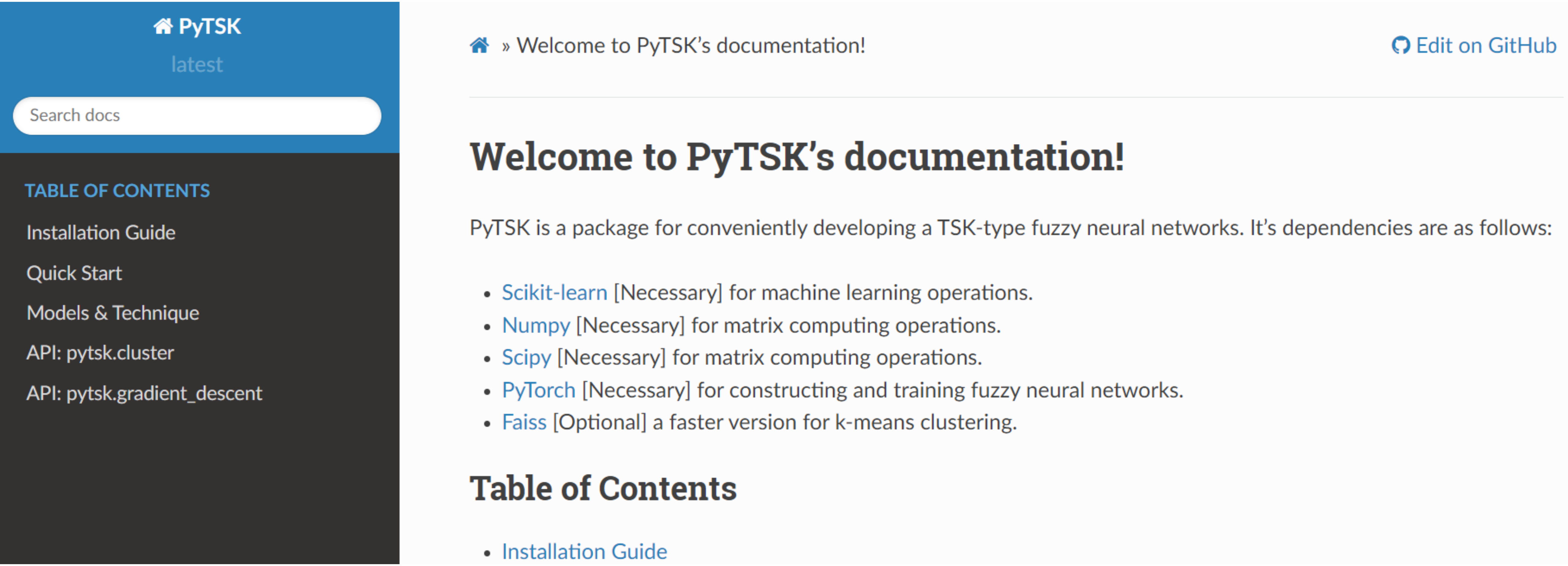
Conclusions

- TSK FS is functionally equivalent to RBFN, MoE, CART and stacking.
- Techniques for the latter models can be used to optimize TSK FSs.

Step	Regression	Classification
Define the objective function	L2 regularization	L2 regularization + Uniform Regularization
Initialize the rules	Semi-Random Initialization	*k*-means Clustering Initialization
Fine-tune the rules: Handle big data	Mini-batch Gradient Descent (MBGD)	Mini-batch Gradient Descent (MBGD)
Fine-tune the rules: Speed-up training	AdaBound AdaBelief	AdaBound
Fine-tune the rules: Improve generalization	DropRule Layer Normalization	Batch Normalization

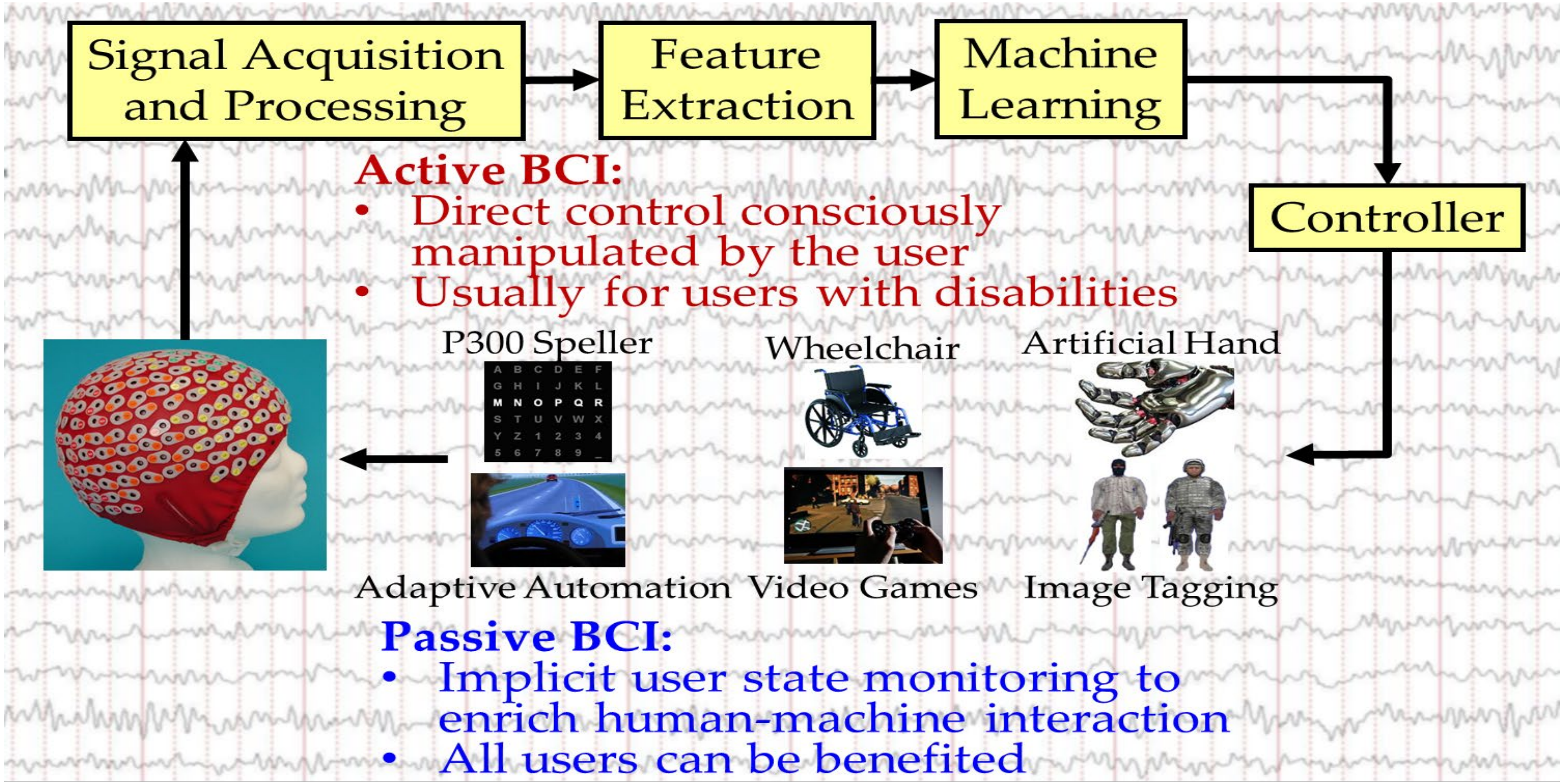
Source Code

- Matlab: https://github.com/drwuHUST
- PyTSK: https://github.com/YuqiCui/PyTSK



Brain-Computer Interface (BCI)

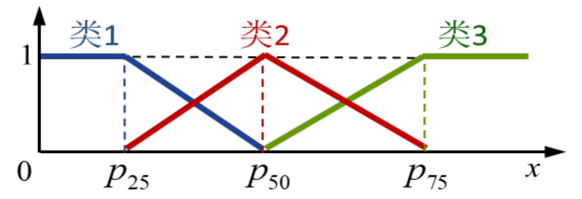
- A direct communication pathway between the brain and an external device.
- Research, assist, augment, or repair cognitive or sensory-motor functions.



Fuzzy Sets in BCIs

挑战：脑机接口分类问题研究多，回归问题算法少

创新：提出模糊类概念，推广分类算法至回归，实现脑机接口精准回归



使用模糊集定义模糊类

信号处理

$$\mathbf{W}_k^* = \arg\max_{\mathbf{W}\in\mathbb{R}^{C\times F}} \frac{\mathrm{Tr}(\mathbf{W}^T \boxed{\bar{\Sigma}_k} \mathbf{W})}{\mathrm{Tr}[\mathbf{W}^T \boxed{(\sum_{i\neq k}\bar{\Sigma}_i)} \mathbf{W}]}$$

共同空间模式滤波器

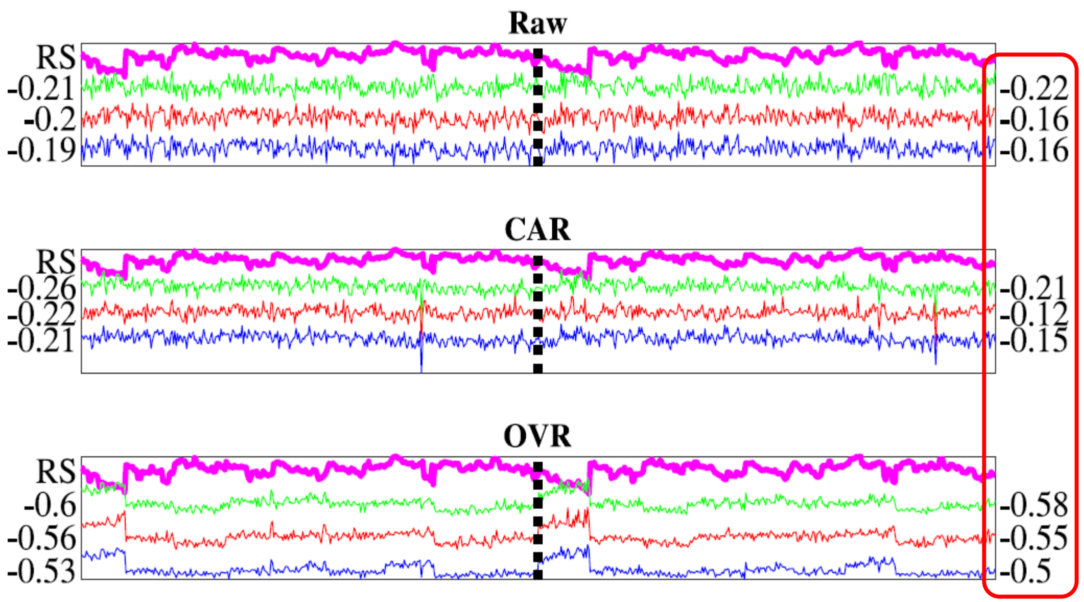
机器学习

$$f = \arg\min_f \sum_{i=1}^{n} (y_i - f(\mathbf{x}_i))^2 + w^t \sum_{i=n+1}^{n+m} (y_i - f(\mathbf{x}_i))^2 - \gamma \tilde{r}^2(y, f(\mathbf{x})) + \lambda[d(P^z, P^t) + \boxed{d(Q^z, Q^t)}]$$

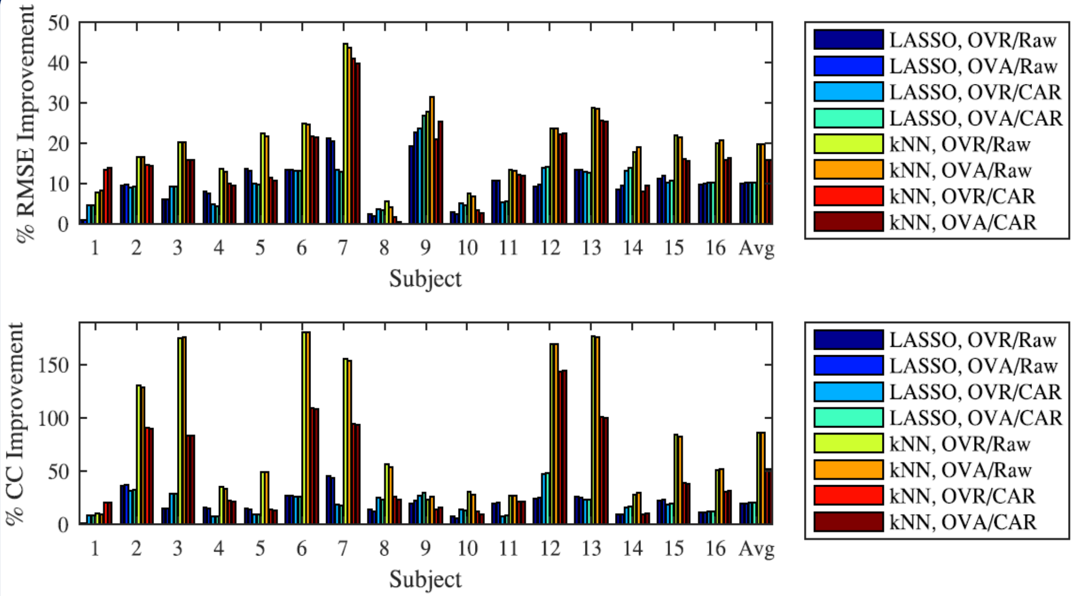
域适配迁移学习

1. D. Wu, V. Lawhern, S. Gordon, B. Lance and C-T Lin, "Driver Drowsiness Estimation from EEG Signals Using Online Weighted Adaptation Regularization for Regression (OwARR)," *IEEE Trans. on Fuzzy Systems*, 25(6):1522-1535, 2017.
2. D. Wu, J-T King, C-C Chuang, C-T Lin and T-P Jung, "Spatial Filtering for EEG-Based Regression Problems in Brain-Computer Interface (BCI)," *IEEE Trans. on Fuzzy Systems*, 26(2):771-781, 2018.

Fuzzy Sets in BCIs



回归滤波 提高特征相关性 2.6 倍



回归滤波 降低估计误差 10-20%

Fuzzy sets can be integrated with state-of-the-art signal processing and machine learning approaches for EEG-based BCIs

1. D. Wu, V. Lawhern, S. Gordon, B. Lance and C-T Lin, "Driver Drowsiness Estimation from EEG Signals Using Online Weighted Adaptation Regularization for Regression (OwARR)," *IEEE Trans. on Fuzzy Systems*, 25(6):1522-1535, 2017.
2. D. Wu, J-T King, C-C Chuang, C-T Lin and T-P Jung, "Spatial Filtering for EEG-Based Regression Problems in Brain-Computer Interface (BCI)," *IEEE Trans. on Fuzzy Systems*, 26(2):771-781, 2018.

References

1. D. Wu and J. M. Mendel, “Recommendations on designing practical interval type-2 fuzzy systems,” *Engineering Applications of Artificial Intelligence*, 95:182–193, 2019.
2. D. Wu, C-T Lin, J. Huang & Z. Zeng, "On the Functional Equivalence of TSK Fuzzy Systems to Neural Networks, Mixture of Experts, CART, and Stacking Ensemble Regression," *IEEE Trans. on Fuzzy Systems*, 28(10): 2570-2580, 2020.
3. D. Wu, Y. Yuan, J. Huang & Y. Tan, "Optimize TSK Fuzzy Systems for Regression Problems: Mini-Batch Gradient Descent with Regularization, DropRule and AdaBound," *IEEE Trans. on Fuzzy Systems*, 28(5): 1003-1015, 2020.
4. Y. Cui, J. Huang and D. Wu, "Optimize TSK Fuzzy Systems for Classification Problems: Mini-Batch Gradient Descent with Uniform Regularization and Batch Normalization," *IEEE Trans. on Fuzzy Systems*, 28(12), 2020.
5. Z. Shi, D. Wu*, C. Guo, C. Zhao, Y. Cui and F-Y Wang*, "FCM-RDpA: TSK Fuzzy Regression Model Construction Using Fuzzy C-Means Clustering, Regularization, DropRule, and Powerball AdaBelief,” *Information Sciences*, 574: 490-504, 2021.
6. Y. Cui, D. Wu*, Y. Xu and R. Peng, "Layer Normalization for TSK Fuzzy System Optimization in Regression Problems," *IEEE Trans. on Fuzzy Systems*, 31(1):254-264, 2023.
7. Y. Cui, D. Wu and Y. Xu, "Curse of Dimensionality for TSK Fuzzy Neural Networks: Explanation and Solutions," *Int'l Joint Conf. on Neural Networks (IJCNN),* Shenzhen, China, July 2021.
8. D. Wu, V. Lawhern, S. Gordon, B. Lance and C-T Lin, "Driver Drowsiness Estimation from EEG Signals Using Online Weighted Adaptation Regularization for Regression (OwARR)," *IEEE Trans. on Fuzzy Systems*, 25(6):1522-1535, 2017.
9. D. Wu, J-T King, C-C Chuang, C-T Lin and T-P Jung, "Spatial Filtering for EEG-Based Regression Problems in Brain-Computer Interface (BCI)," *IEEE Trans. on Fuzzy Systems*, 26(2):771-781, 2018.

Thank you!