A Convolutional Fuzzy Neural Network Architecture for Object Classification with Small Training Database

Min-Jie Hsu1 • Yi-Hsing Chien1 • Wei-Yen Wang1 • Chen-Chien Hsu1

International Journal of Fuzzy Systems 2020

Outline

Introduction

Motivation

Basic concepts

Proposed method

Results

Conclusion

Introduction

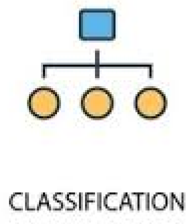
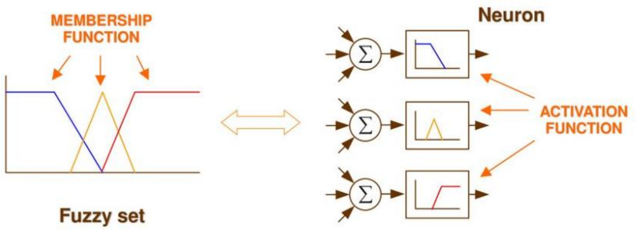


Image Classification – 90% precision rate

CNN

Fuzzy Neural Network – fuzzy values no crisp

Small data



Fully connected layers => Fuzzy neural network

Motivation

Data!

- Sufficient training data (quality, size)
- Not enough data – transfer learning
- Relevant data

- Crispy => fuzzy values
- Strengthen the ability to approximate function

Basic concepts

CNN

- Powerful technique
  - Some correlation between input data
- Weight sharing
  - Gradient vanishing, overfitting
- Extract features from input
- Backpropagation

$$a_j^l = f\left(\sum_{i=0}^{2}\left(x_i * w_{ij}^l\right) + b_j^l\right),\ j = 1, 2, \ldots, n^l$$

$$\partial \mathrm{loss} \Big/ \partial w_{ij}^l = a_j^l * \delta_j^{l+1}$$

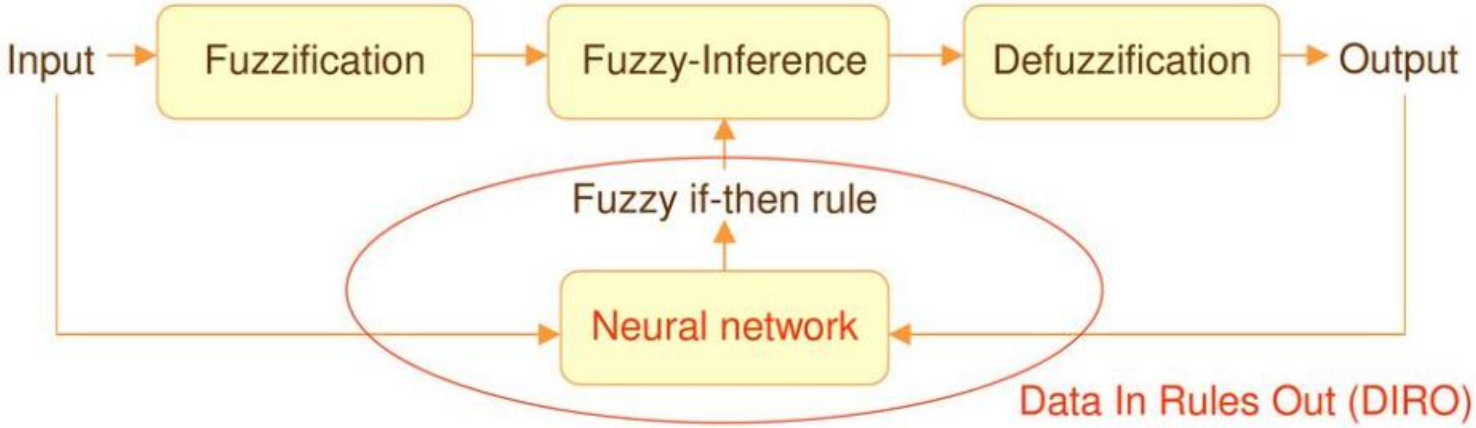
$$\delta_j^l = \begin{cases} f'\left(z_j^l\right) \otimes \text{upsample}\left(\delta_j^{l+1} * w_j^l\right), & \text{if } (l+1) \text{ layer is subsampling layer} \\ f'\left(z_j^l\right) \otimes \left(\delta_j^{l+1} * w_j^l\right), & \text{otherwise} \end{cases}$$

Fuzzy Neural Network (FNN)

- Input unit is graded – membership to fuzzy set
- Fuzzy value

$$R^l : \text{IF } x_1 \text{ is } A_1^l \text{ and} \cdots x_n \text{ is } A_n^l$$
$$\text{THEN } y_1 \text{ is } w_1^l \text{ and} \cdots y_m \text{ is } w_m^l$$

$$y_j = \frac{\sum_{l=1}^{h} w_j^l \left(\prod_{i=1}^{n} \mu_{A_i^l}(x_i) \right)}{\sum_{l=1}^{h} \left(\prod_{i=1}^{n} \mu_{A_i^l}(x_i) \right)}$$



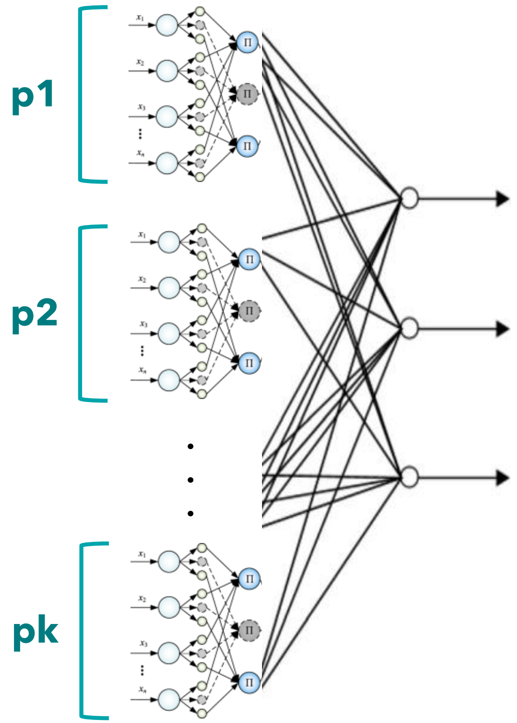
Fuzzy Neural Network (FNN)

When input number is too large!

- K independent inference engine

$$x_i^p, p = 1, 2, \ldots, k,\ i = 1, 2, \ldots, (n/k)$$

$$R^{p,l}: \text{IF } x_1^p \text{ is } A_1^l \text{ and} \cdots x_n^p \text{ is } A_n^l$$
$$\text{THEN } y_1 \text{ is } w_1^{p,l} \text{ and} \cdots y_m \text{ is } w_m^{p,l}$$



Proposed method

Proposed method

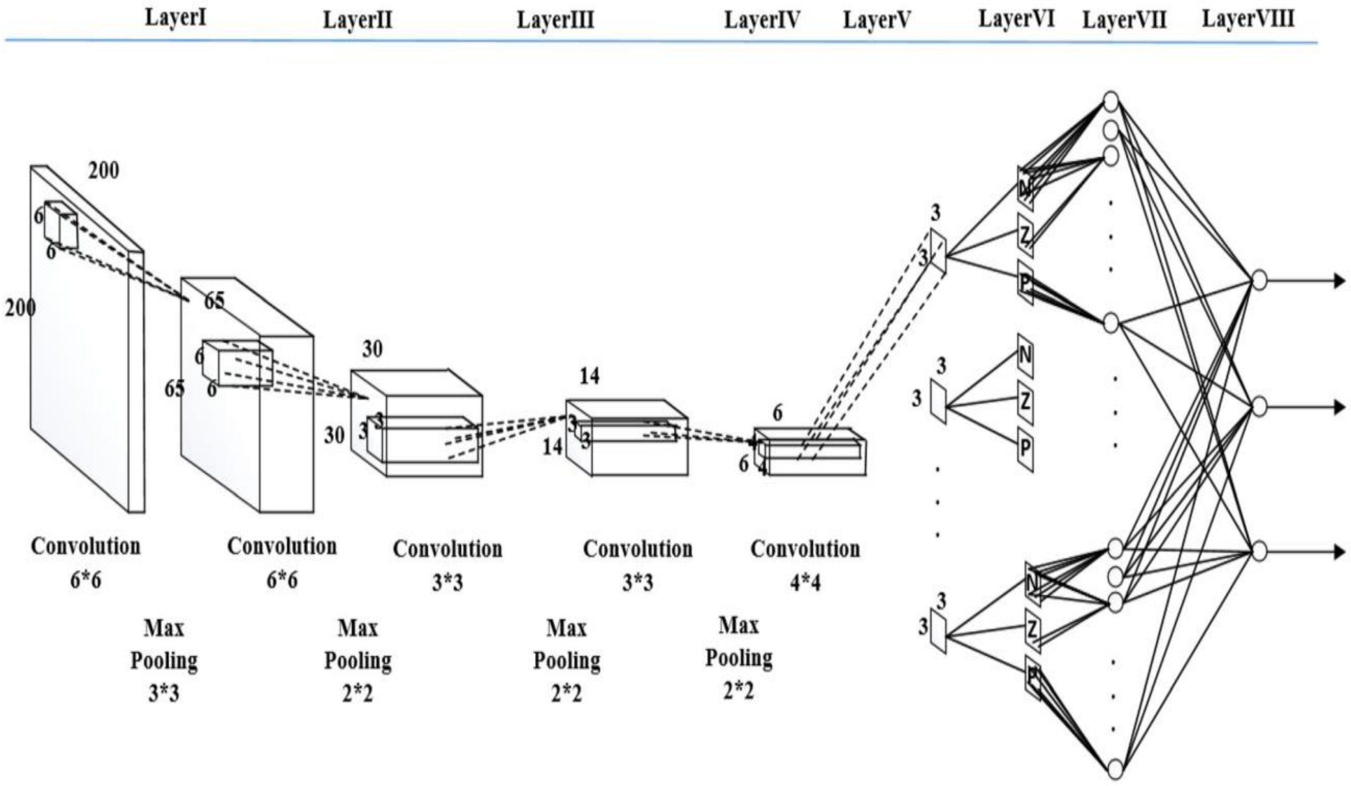
- Dropout – overfitting

$$\phi_q^p = \prod_{i=1}^{h\times w} \mu_{F_i^q}(x_i)$$

$$N = e^{-\frac{(x+m)^2}{2\sigma^2}},\ Z = e^{-\frac{x^2}{2\sigma^2}},\ P = e^{-\frac{(x-m)^2}{2\sigma^2}}$$

Fuzzy inference units:

$3^{3\times3\times80} \rightarrow 3^{3\times3}$



Proposed method

- - $\phi_q^p = \prod_{i=1}^{9} \mu_{F_i^q}^p(x_i)$

$$\mu_{F_i^q}^p\left(\zeta_{p,i}^5\right) = \begin{bmatrix} N\left(\zeta_{p,1}^5\right) \times \cdots N\left(\zeta_{p,i}^5\right) \times \cdots N\left(\zeta_{p,9}^5\right) \\ Z\left(\zeta_{p,1}^5\right) \times N\left(\zeta_{p,i}^5\right) \times \cdots N\left(\zeta_{p,9}^5\right) \\ \cdot \\ \cdot \\ P\left(\zeta_{p,1}^5\right) \times \cdots P\left(\zeta_{p,i}^5\right) \times \cdots P\left(\zeta_{p,9}^5\right) \end{bmatrix}$$

$$\psi = \begin{bmatrix} \phi_1^1 \\ \phi_2^1 \\ \cdot \\ \cdot \\ \phi_{19683}^{80} \end{bmatrix}, \ z^8 = \begin{bmatrix} z_1^8 \\ z_2^8 \\ z_3^8 \end{bmatrix}, \ w^8 = \begin{bmatrix} w_{1,1}^8 & \cdots & w_{1,19683}^8 \\ w_{2,1}^8 & \cdots & w_{2,19683}^8 \\ w_{3,1}^8 & \cdots & w_{3,19683}^8 \end{bmatrix}$$

$$z^8 = w^8 \times \psi, \qquad y = a^8 = \frac{1}{e^{z_1^8} + e^{z_2^8} + e^{z_3^8}} \begin{bmatrix} e^{z_1^8} \\ e^{z_2^8} \\ e^{z_3^8} \end{bmatrix}$$

Proposed method

CFNN	Feature numbers	Weight numbers
Layer I	65 × 65 × 20	6 × 6 × 3 × 20
Convolution 6 × 6		
Max pooling 3 × 3		
Layer II	30 × 30 × 40	6 × 6 × 20 × 40
Convolution 6 × 6		
Max pooling 2 × 2		
Layer III	14 × 14 × 40	3 × 3 × 40 × 40
Convolution 3 × 3		
Max pooling 2 × 2		
Layer IV	6 × 6 × 40	3 × 3 × 40 × 40
Convolution 3 × 3		
Max pooling 2 × 2		
Layer V	3 × 3 × 80	4 × 4 × 40 × 80
Convolution 3 × 3		
Layer VI	2160	N/A
Fuzzifier		
Layer VII	1,574,640	N/A
Inference Layer		
Layer VIII	3	1,574,640 × 3
Defuzzifier		

Proposed method

- Cross entropy

$$L = -\sum_{i=1}^{3} d_i \ln(y_i)$$

$$\partial L/\partial z^8 = \partial y/\partial z^8 \partial L/\partial y,$$

$$\frac{\partial y_i}{\partial z^8} = \begin{bmatrix} \frac{\partial y_1}{\partial z_1^8} & \frac{\partial y_2}{\partial z_1^8} & \frac{\partial y_3}{\partial z_1^8} \\ \frac{\partial y_1}{\partial z_2^8} & \frac{\partial y_2}{\partial z_2^8} & \frac{\partial y_3}{\partial z_2^8} \\ \frac{\partial y_1}{\partial z_3^8} & \frac{\partial y_3}{\partial z_3^8} & \frac{\partial y_3}{\partial z_3^8} \end{bmatrix} \text{ and } \frac{\partial L}{\partial y} = \begin{bmatrix} \frac{\partial L}{\partial y_1} \\ \frac{\partial L}{\partial y_2} \\ \frac{\partial L}{\partial y_3} \end{bmatrix} = \begin{bmatrix} \frac{d_1}{y_1} \\ \frac{d_2}{y_2} \\ \frac{d_3}{y_3} \end{bmatrix}$$

$$\partial y_i / \partial z_j^8,$$

$$\frac{\partial y_i}{\partial z_j^8} = \begin{cases} \dfrac{e^{z_j^8}\sum_{k=1}^{3} e^{z_k^8} - e^{z_j^8} e^{z_j^8}}{\left(\sum_{k=1}^{3} e^{z_k^8}\right)^2} = \dfrac{e^{z_j^8}}{\sum_{k=1}^{3} e^{z_k^8}}\left(1 - \dfrac{e^{z_j^8}}{\sum_{k=1}^{3} e^{z_k^8}}\right) = y_j(1-y_j), & \text{if } i = j \\ \dfrac{0 - e^{z_i^8} e^{z_j^8}}{\left(\sum_{k=1}^{3} e^{z_k^8}\right)^2} = \dfrac{-e^{z_i^8}}{\sum_{k=1}^{3} e^{z_k^8}} \dfrac{e^{z_j^8}}{\sum_{k=1}^{3} e^{z_k^8}} = -y_i y_j, & \text{if } i \neq j \end{cases}$$

$$\frac{\partial L}{\partial z_j^8} = -\sum_{i=1, i\neq j}^{3} \frac{d_i}{y_i}(-y_i y_j) - \frac{d_j}{y_j}(y_j(1-y_j))$$

$$= \sum_{i=1, i\neq j}^{3} d_i y_j - d_j + d_j y_j = -d_j + y_j \sum_{i=1}^{3} d_i.$$

Proposed method

- Cross entropy

$$\sum_{i=1}^{3} d_i = 1,$$

$$\boxed{\partial L / \partial z_j^8 = -d_j + y_j.}$$

$$\boxed{\partial L / \partial w^8 = \partial L / \partial z^8 \psi^T = (a^8 - d)\psi^T.}$$

$$\boxed{\partial L / \partial \phi^p = (w^8)^T \partial L / \partial z^8 = (w^8)^T (a^8 - d)}$$

$$\frac{\partial L}{\partial z_p^5} = \left(\frac{\partial(\phi_1^p, \ldots, \phi_{19688}^p)}{\partial(z_{p,1}^5, \ldots, z_{p,9}^5)} \right)^T \times \frac{\partial L}{\partial \phi^p}$$

$$= \begin{bmatrix} \frac{\partial \phi_1^p}{\partial z_{p,1}^5} & \frac{\partial \phi_2^p}{\partial z_{p,1}^5} & \cdots & \frac{\partial \phi_{19688}^p}{\partial z_{p,1}^5} \\ \frac{\partial \phi_1^p}{\partial z_{p,1}^5} & \cdots & \cdots & . \\ . & \cdots & \cdots & . \\ \frac{\partial \phi_1^p}{\partial z_{p,9}^5} & \cdots & \cdots & \frac{\partial \phi_{19688}^p}{\partial z_{p,9}^5} \end{bmatrix} \times \begin{bmatrix} \frac{\partial L}{\partial \phi_1^p} \\ \frac{\partial L}{\partial \phi_2^p} \\ . \\ \frac{\partial L}{\partial \phi_{19688}^p} \end{bmatrix}$$

$$\partial \phi_q^p / \partial z_{p,i}^5 = (\partial \phi_q^p / \partial \mu_{F_i^q}) \times (\partial \mu_{F_i^q} / \partial \zeta_{p,i}^5) \times (\partial \zeta_{p,i}^5 / \partial z_{p,i}^5),$$

$$\frac{\partial \mu_{F_i^q}}{\partial \zeta_{p,i}^5} = \frac{\partial e^{-\frac{\left(\zeta_{p,i}^5 + m_q\right)^2}{2\sigma^2}}}{\partial \zeta_{p,i}^5} = -e^{-\frac{\left(\zeta_{p,i}^5 + m_q\right)^2}{2\sigma^2}} \times \frac{\zeta_{p,i}^5}{\sigma^2} = \mu_{F_i^q} \times \frac{\zeta_{p,i}^5}{\sigma^2},$$

$$\frac{\partial \phi_q^p}{\partial \mu_{F_i^q}} = \frac{\prod_{j=1}^{9} \left(\mu_{F_j^q}\left(\zeta_{p,j}^5 \right) \right)}{\mu_{F_i^q}\left(\zeta_{p,i}^5 \right)} \qquad \frac{\partial \zeta_{p,i}^5}{\partial z_{p,i}^5} = \begin{cases} 0, & \text{if dropped out} \\ 1, & \text{if not dropped out} \end{cases}$$

Results

Case 1

TWGC (Taiwan GPU Cloud) - Tesla V100 GPUs - 16 GB RAM cluster

- 3 classes
- 700 image per class
- ImageNet
- 200x200 RGB
- K-fold validation – k* / k-1
- Dropout

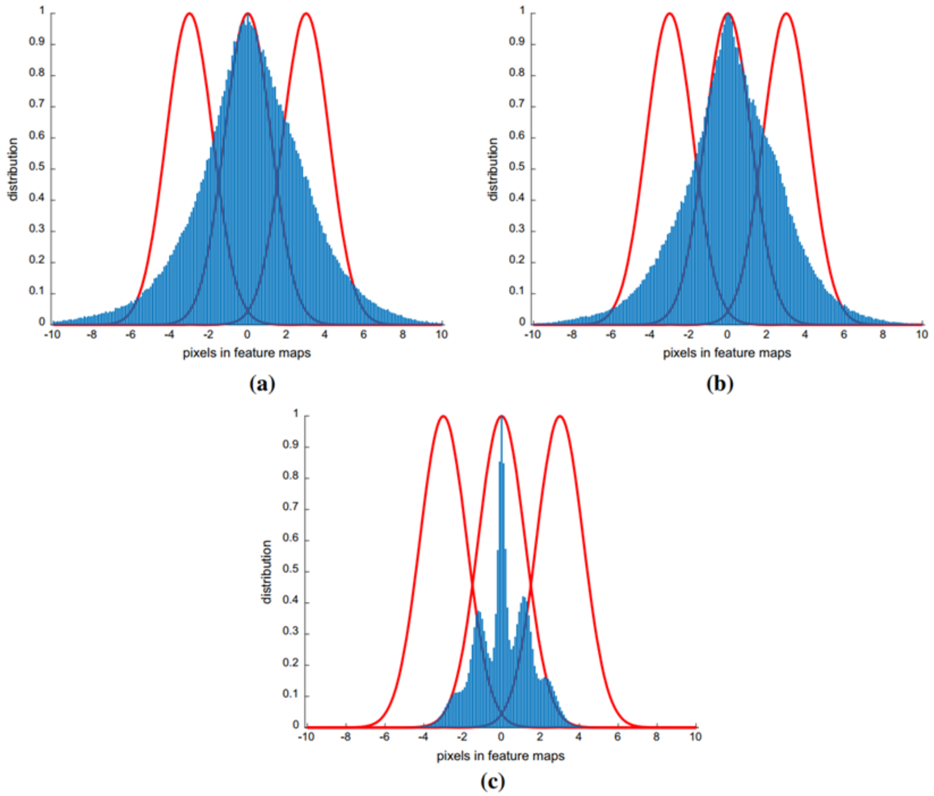
(a) (b) (c)

Case 1

- Expert rules

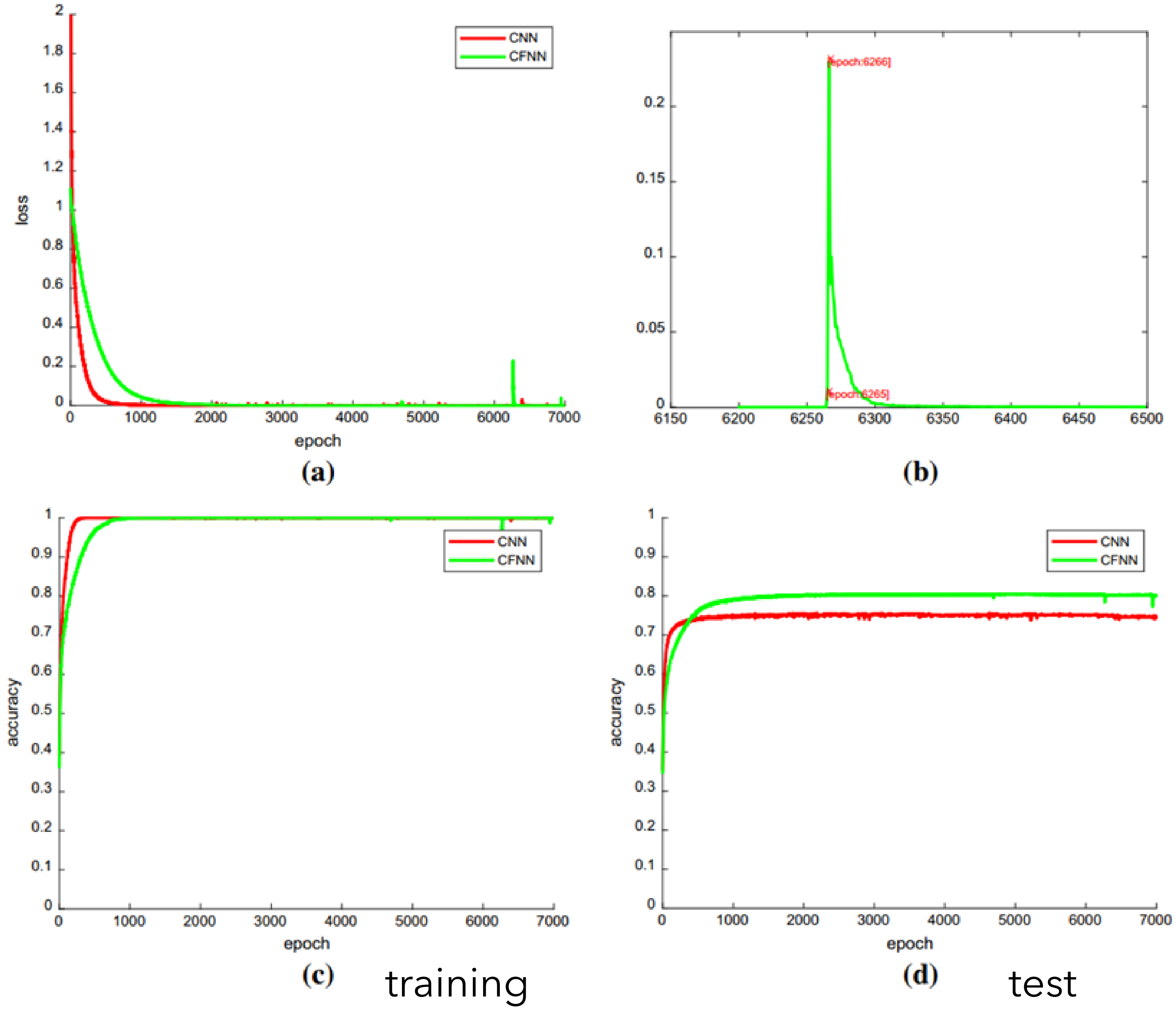
$$m = 3.0,\ \sigma = 1.2.$$

  - Gaussian distribution randomly



Case 1

- Dropout !



Case 2

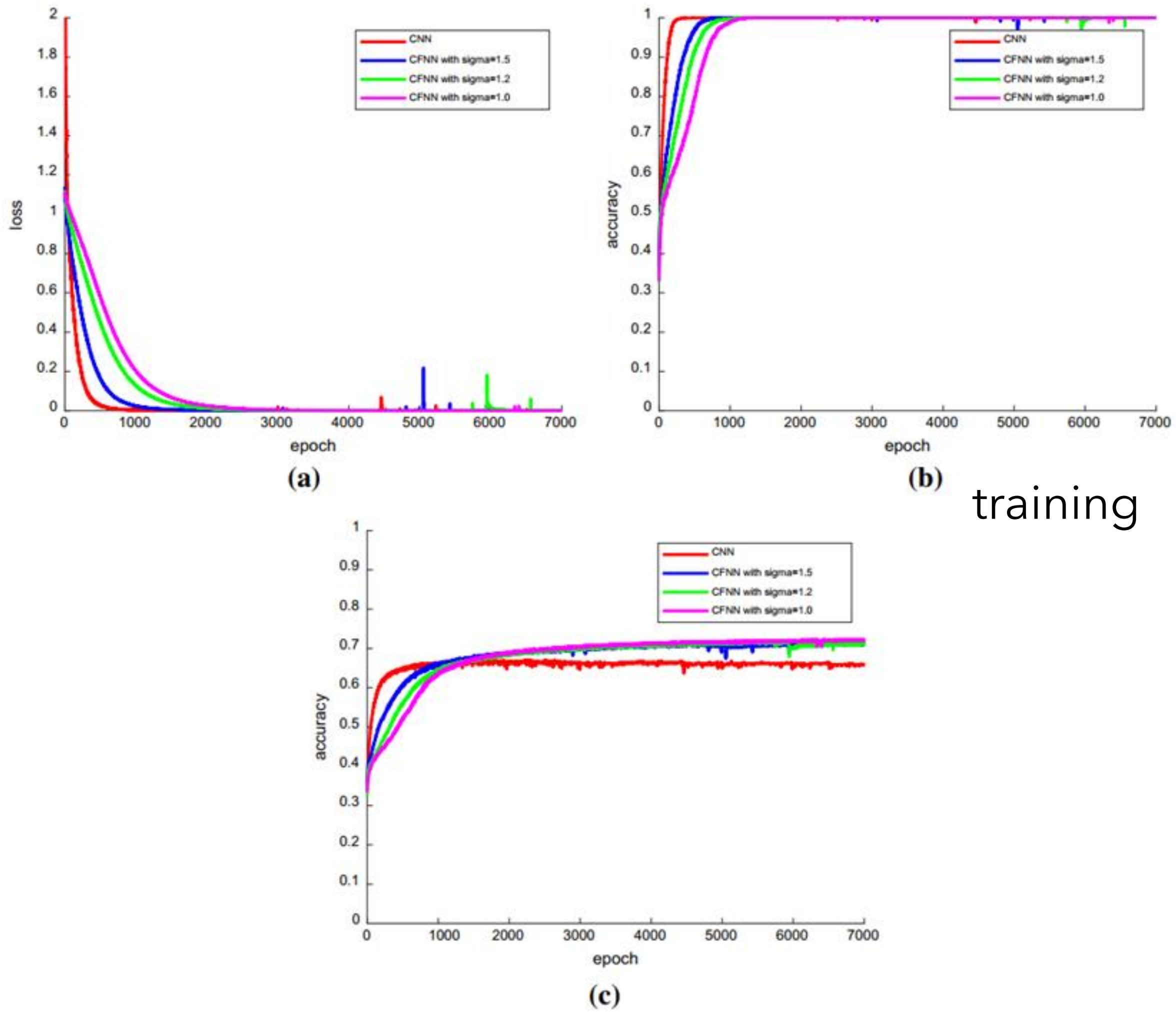
- 700 images per class - ImageNet
- Different membership function
- 10 fold – 210 training set – 1890 test set

(a) (b) (c)

Network architecture	The testing accuracy (%)
CNN	65.83
The proposed CFNN with $\sigma = 1.5$	70.95
The proposed CFNN with $\sigma = 1.2$	71.11
The proposed CFNN with $\sigma = 1.0$	71.98

Case 2

- CNN – fastest converge
- CNN – overfit rapidly
- CFNN
  - – better result – insufficient data



Conclusion

Conclusion and future work

Reduce overfitting

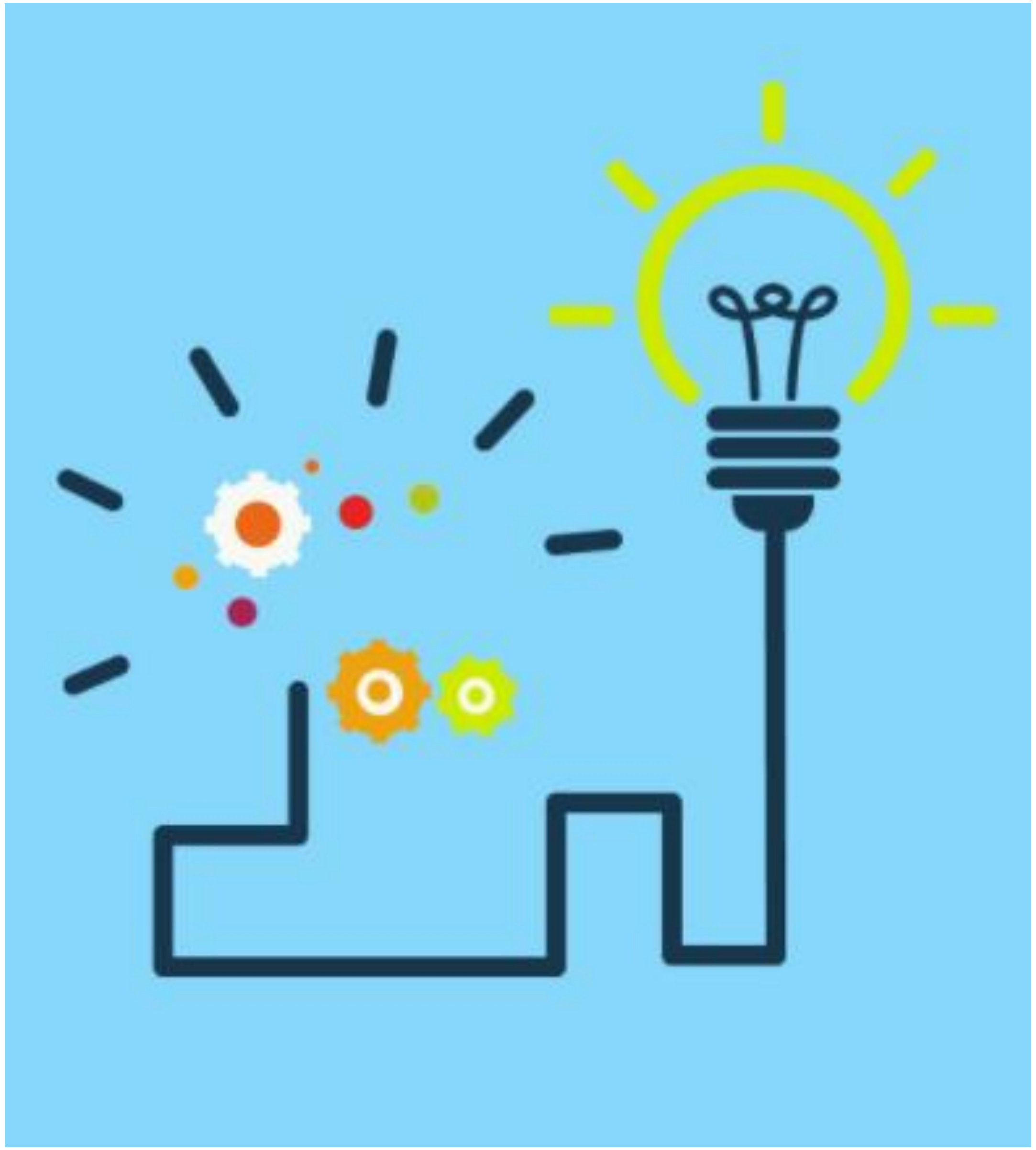
Sum-up feature information

Increase accuracy in tests

Possible to enhance testing accuracy by observing distribution of pixels – feature map – adjust membership function

Optimized membership function =>increase accuracy

Adaptive adjustment strategy



Thank you for your attention.