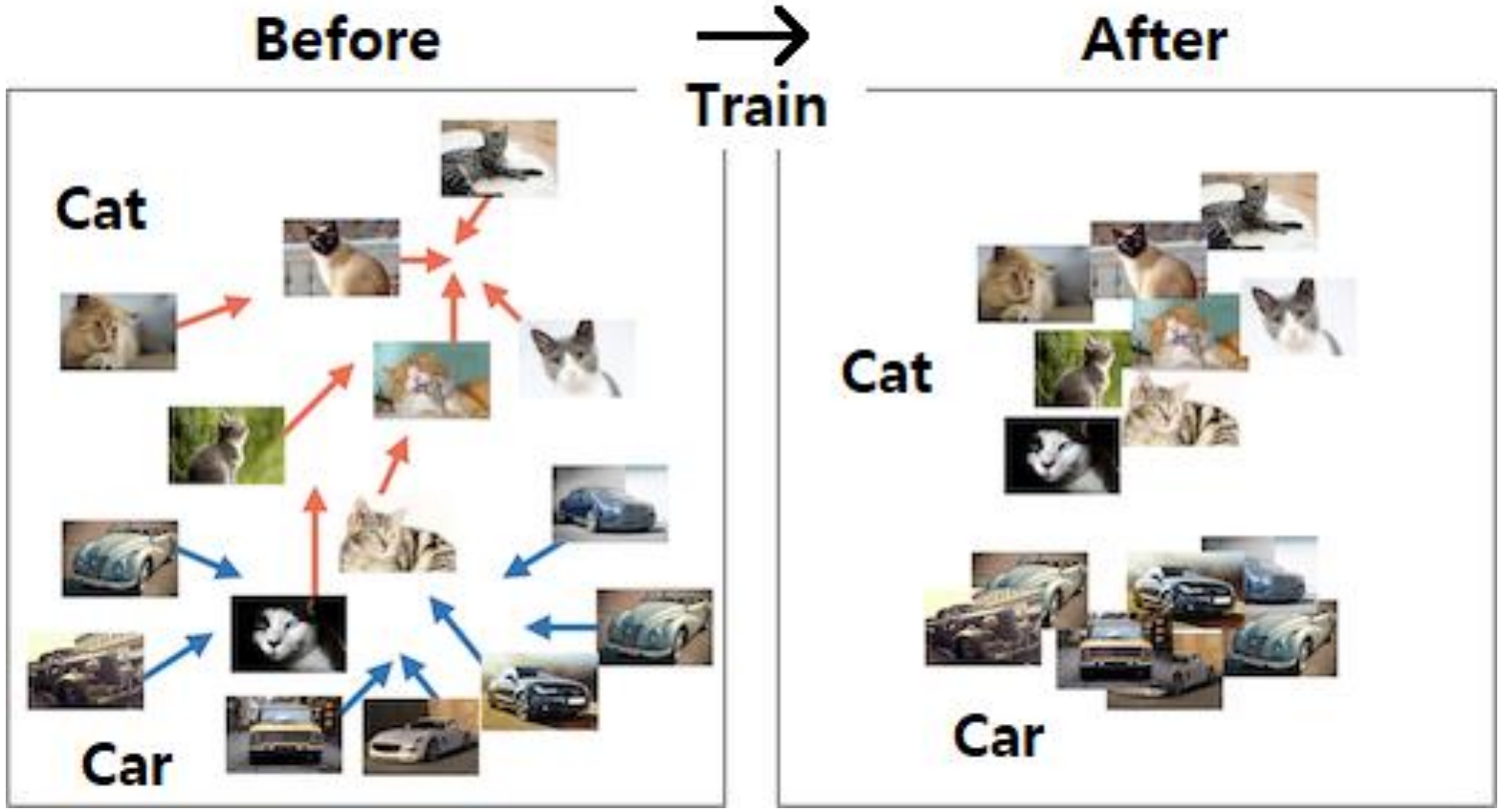
SIAMESE NEURAL NETWORKS FOR ONE-SHOT IMAGE RECOGNITION

JuHyeong Kim

SIAMESE NEURAL NETWORKS

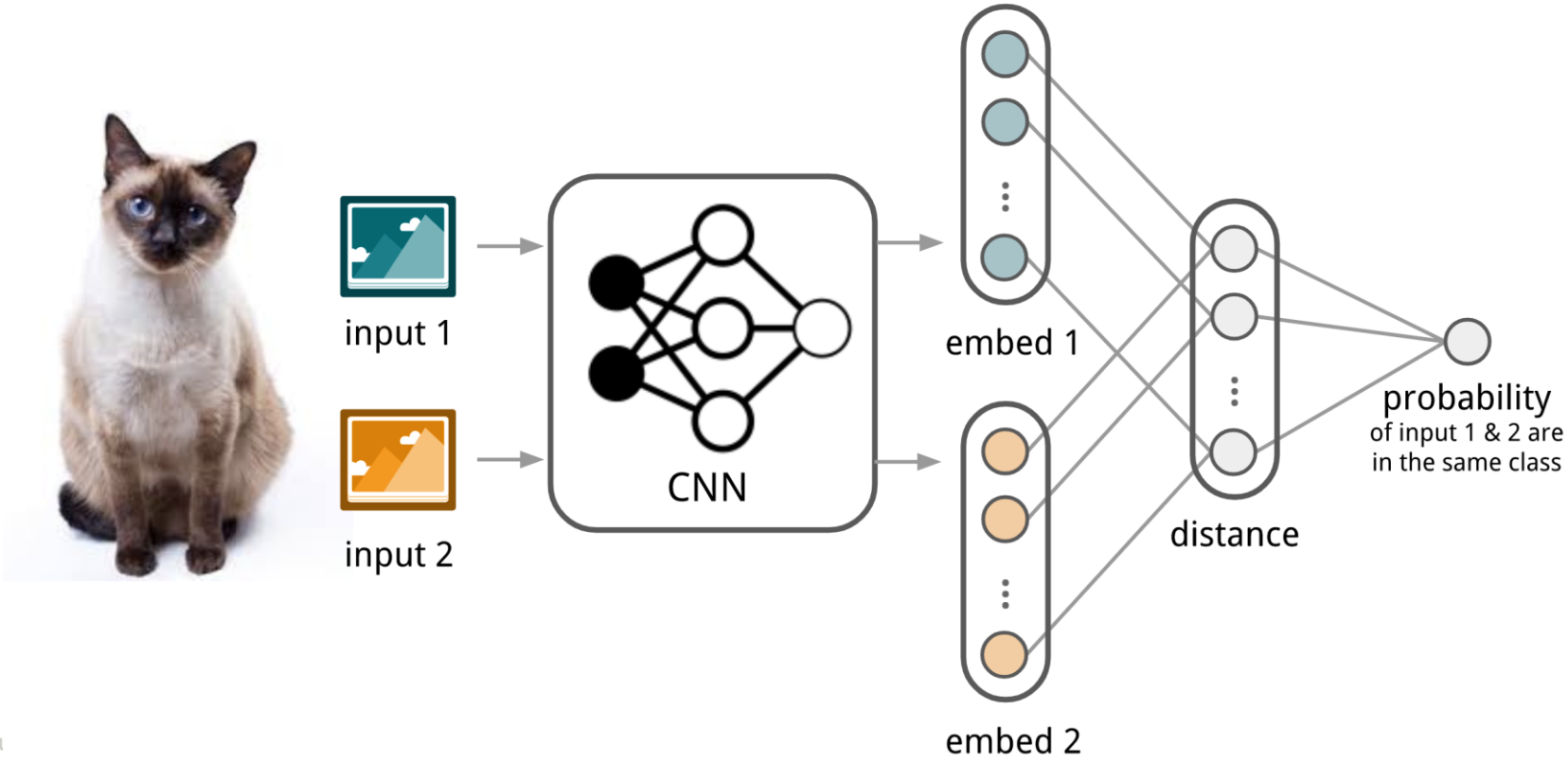
- Siamese neural networks are Metric-based learning.
  - Calculate the distance for each label and measure the same label closely and the other label far away.



https://www.kakaobrain.com/blog/106

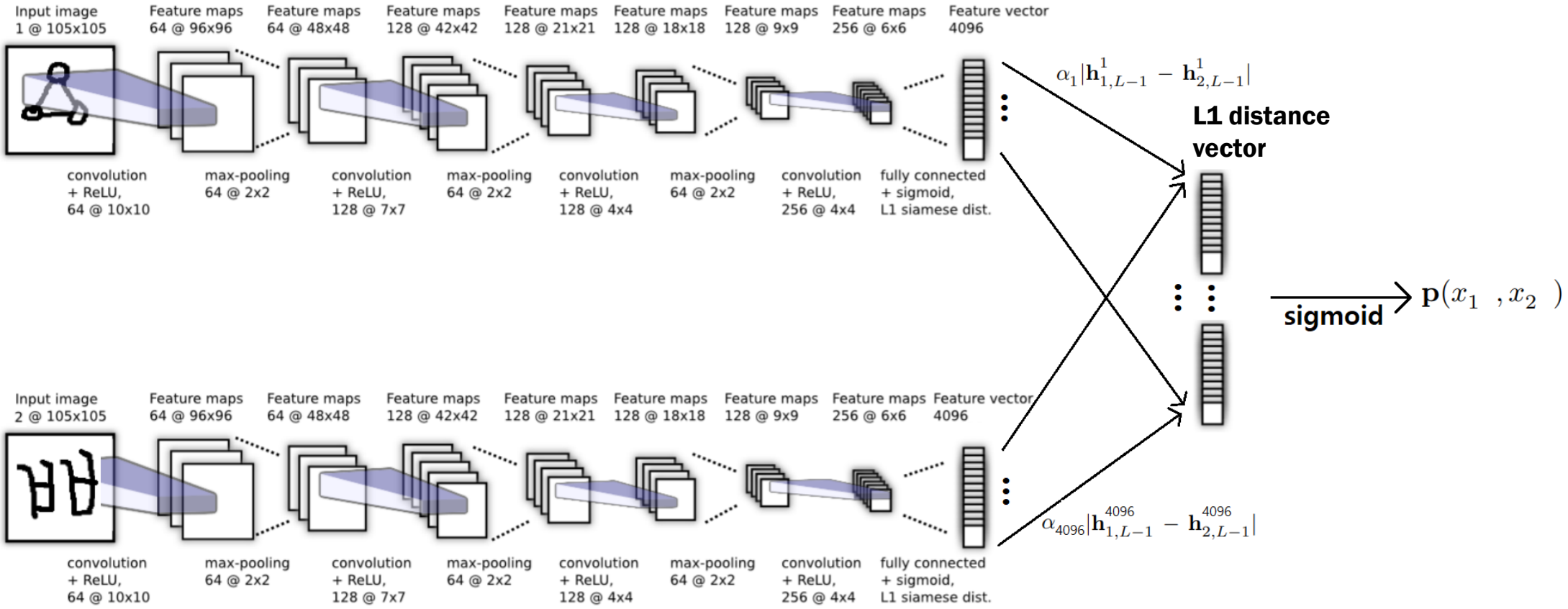
SIAMESE NEURAL NETWORKS

- Siamese Neural network for one-shot image recognition(ICML deep learning workshop, 2015)
- Measure the distance similarity by comparing the two inputs.
  - embed1 and embed2 represent features for input 1 and input2.
  - The last probability represents the similarity between the two inputs.



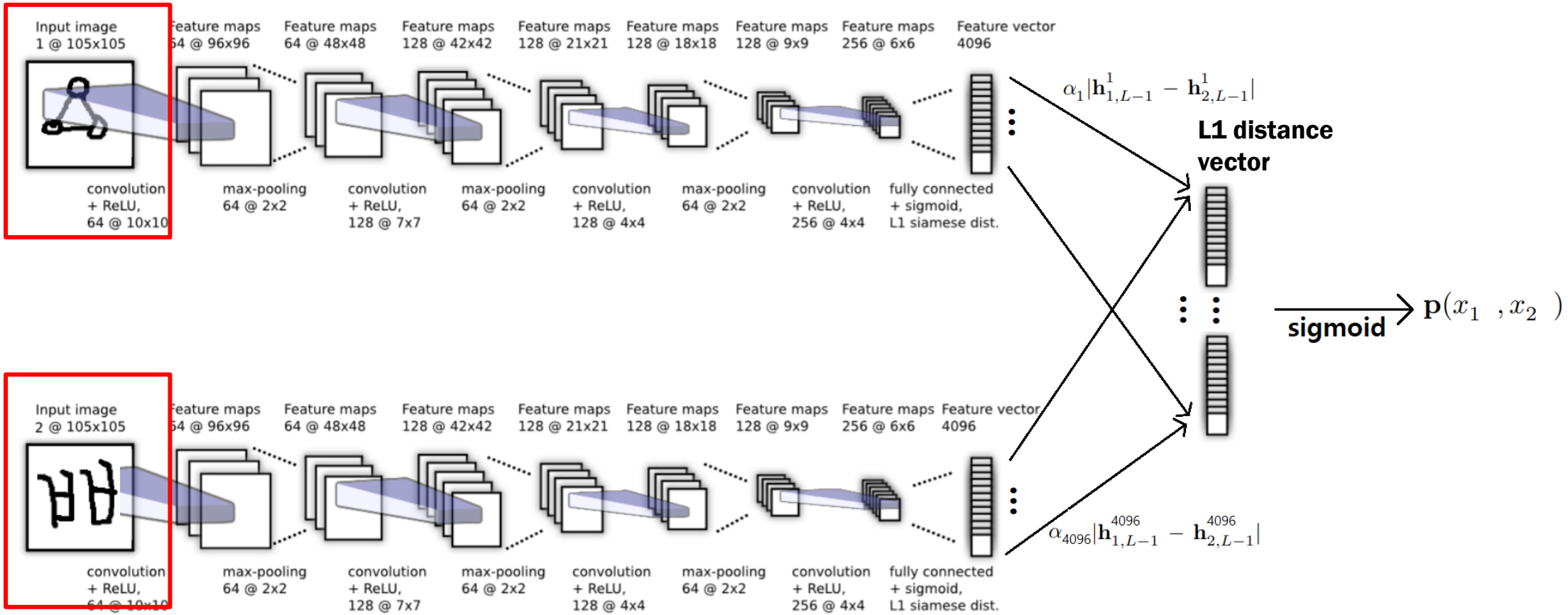
SIAMESE NEURAL NETWORKS STRUCTURE

- The figure below is a specific structure.



SIAMESE NEURAL NETWORKS STRUCTURE

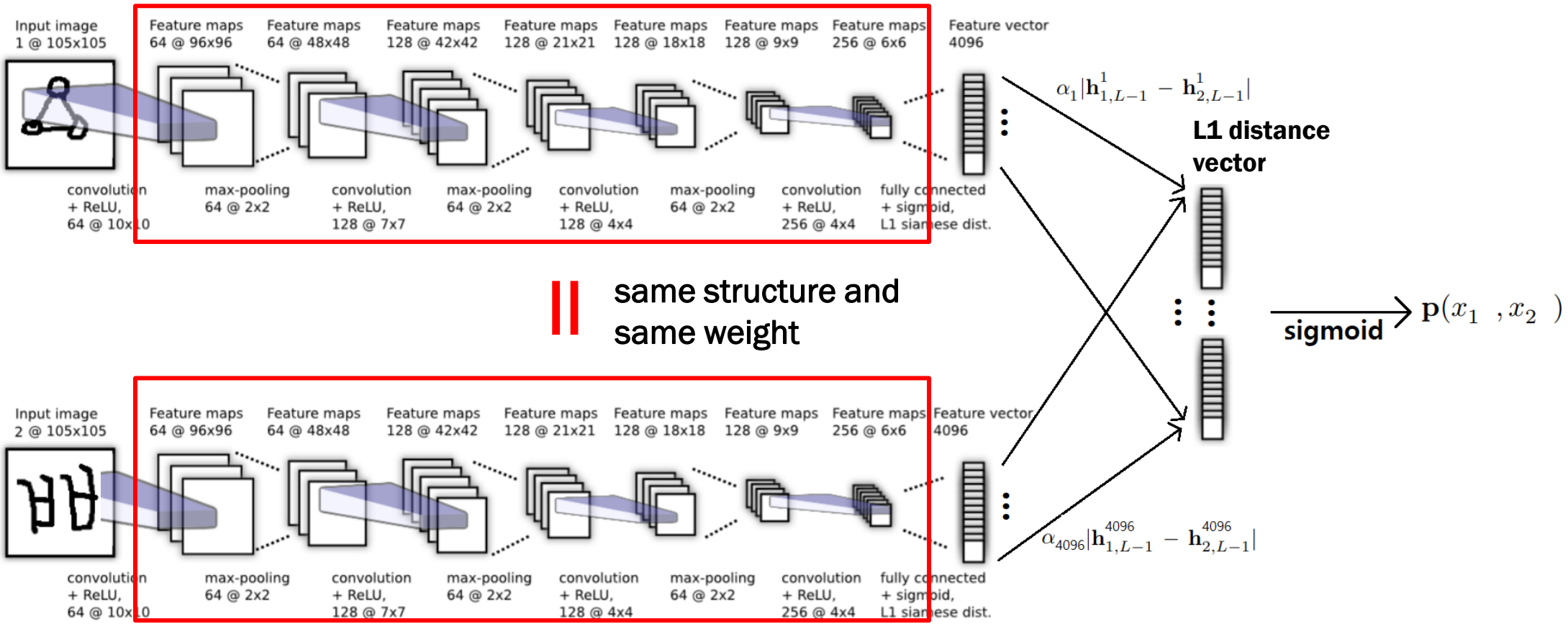
- This part is input data.



SIAMESE NEURAL NETWORKS STRUCTURE

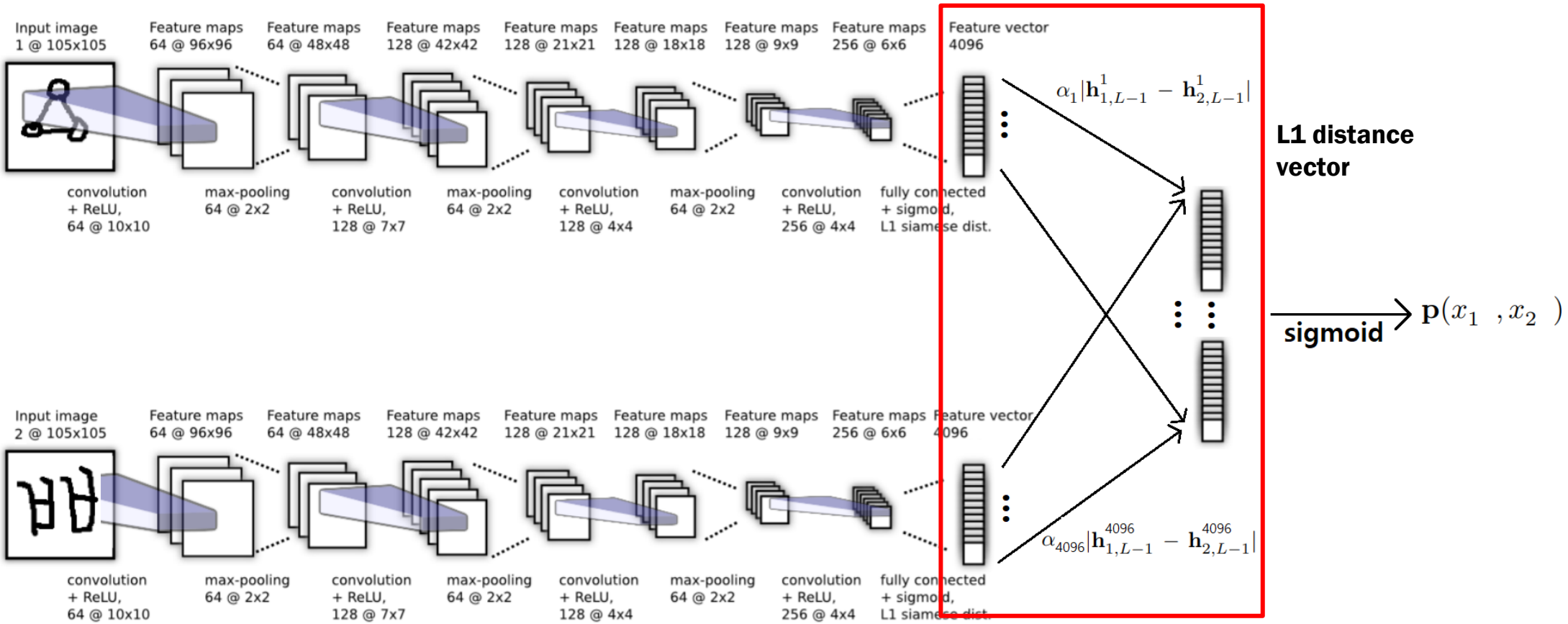
- This part is a CNN structure that extracts characteristics.
- * Same structure, same weight.

  → The same characteristics are extracted.



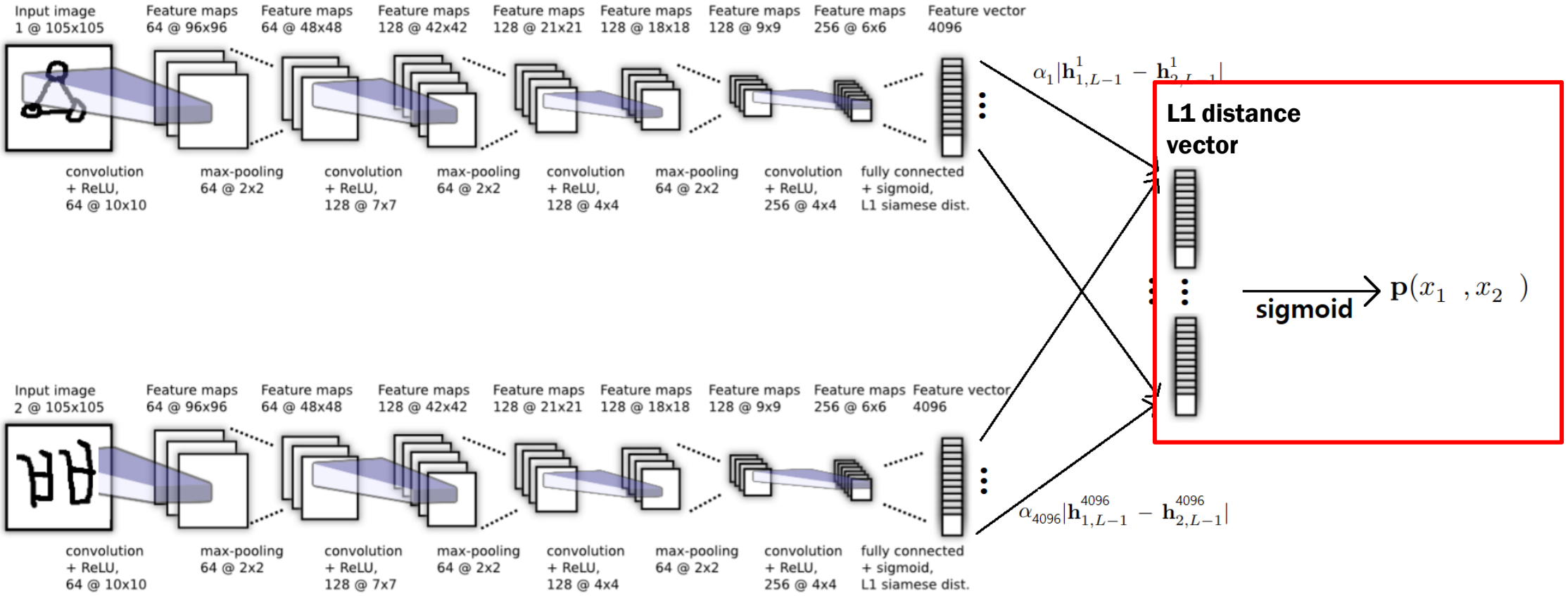
SIAMESE NEURAL NETWORKS STRUCTURE

- The L1 distance values of the feature vectors for two images are calculated and made into one vector.
- It is expressed as $\alpha_j \left| h_{1,L-1}^{(j)} - h_{2,L-1}^{(j)} \right|$
  - *j is the j-th vector, and α is the weight for the L1 distance.*



SIAMESE NEURAL NETWORKS STRUCTURE

- The sum of the L1 distance vector is output as a sigmoid.
  - It is expressed as $\mathbf{p} = \sigma(\sum j\, \alpha_j |h^{(j)}_{1,L-1} - h^{(j)}_{2,L-1}|)$
  - *σ is sigmoid function.*
- The closer the value of $\mathbf{p}$ is to 1, the two images are the same, and the closer to 0, the two images are different.



LEARNING

- This objective is combined with standard backpropagation algorithm
  - where the gradient is additive across the twin networks due to the tied weights
- The loss function is cross entropy and is as follows.
  - $\mathcal{L}\left(x_1^{(i)}, x_2^{(i)}\right) = y\left(x_1^{(i)}, x_2^{(i)}\right)\log\mathbf{p}\left(x_1^{(i)}, x_2^{(i)}\right)+(1- y\left(x_1^{(i)}, x_2^{(i)}\right))\log(1- \mathbf{p}\left(x_1^{(i)}, x_2^{(i)}\right)),$ also $\mathbf{p} = \sigma(\sum j\, \alpha_j |h_{1,L-1}^{(j)} - h_{2,L-1}^{(j)}|)$
  - if same label, $y\left(x_1^{(i)}, x_2^{(i)}\right) = 1$
  - if difference label, $y\left(x_1^{(i)}, x_2^{(i)}\right) = 0$

EXPERIMENTS

- The dataset used The Omniglot Dataset.
  - There are 50 alphabets and there are 15-50 characters for each alphabet. The total number of characters is 1623.
  - It was written by handwriting by 20 people. → drawer (Total 1623 * 20 characters)
- There were two types of verification network and one-shot training/testing.

50 kinds of alphabet
= 1623 characters

x

20
drawer

VERIFICATION NETWORK

- We set aside sixty, twenty, twenty percent of the total data for training, validation, testing
  - Training set : **30 alphabets** out of 50 and **12 drawers** out of 20.
  - Validation set (evaluation set) : **10 alphabets** out of 50 and **4 drawers** out of 20.
  - Test set : **10 alphabets** out of 50 and **4 drawers** out of 20.
- **To train our verification network,** we put together three different data set sizes with 30,000, 90,000, and 150,000 training examples by sampling random same and different pairs.

no distortions : 30k, 90k, 150k of data

8 transforms were used.

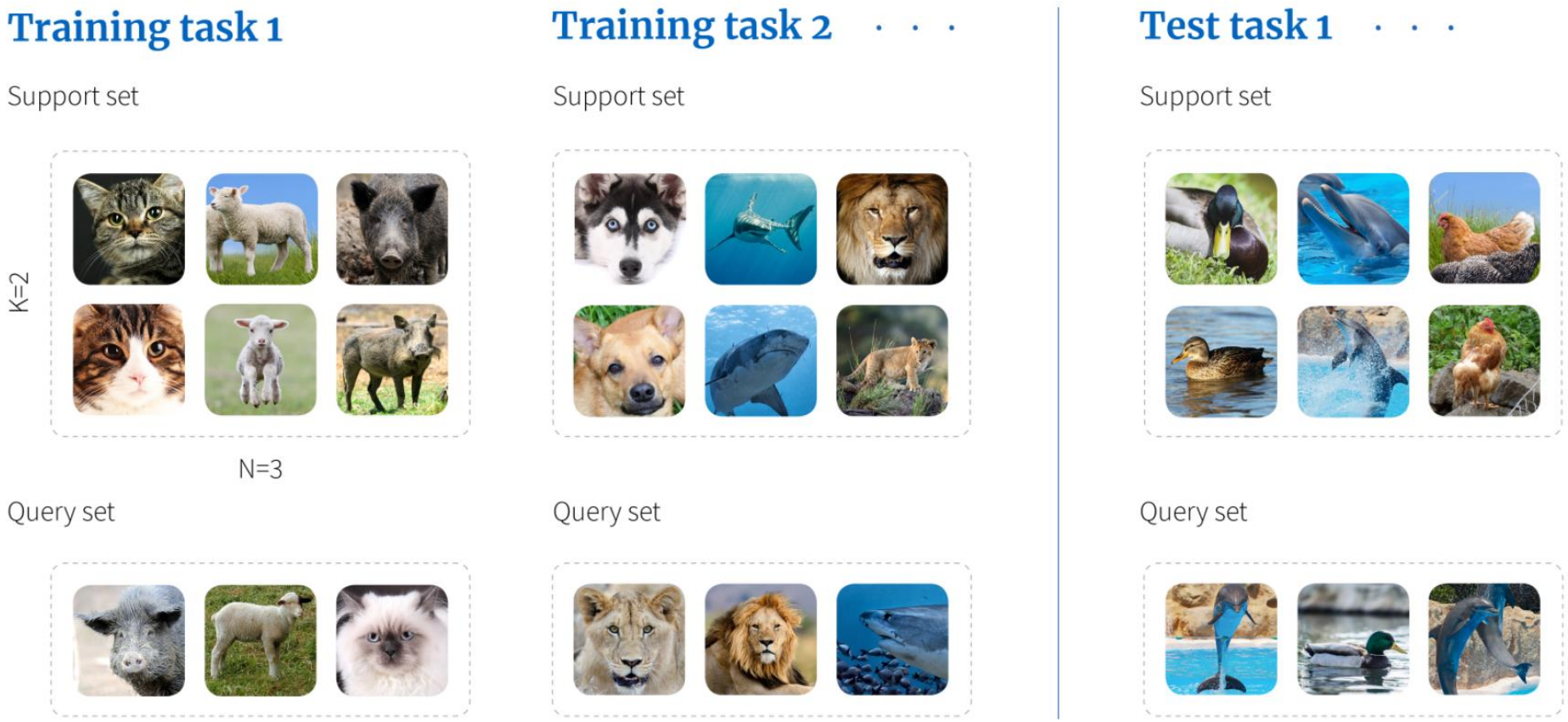
affine distortions x8: 270k, 810k, 1350k of data

Table 1. Accuracy on Omniglot verification task (siamese convolutional neural net)

Method	Test
30k training	
no distortions	90.61
affine distortions x8	91.90
90k training	
no distortions	91.54
affine distortions x8	93.15
150k training	
no distortions	91.63
affine distortions x8	**93.42**

ONE-SHOT LEARNING

- N-way K-shot learning is to choose one of the support sets that has K for each N class.
  - Support Set is a dataset that supports training and testing.
  - Query Set is a dataset for training and testing.

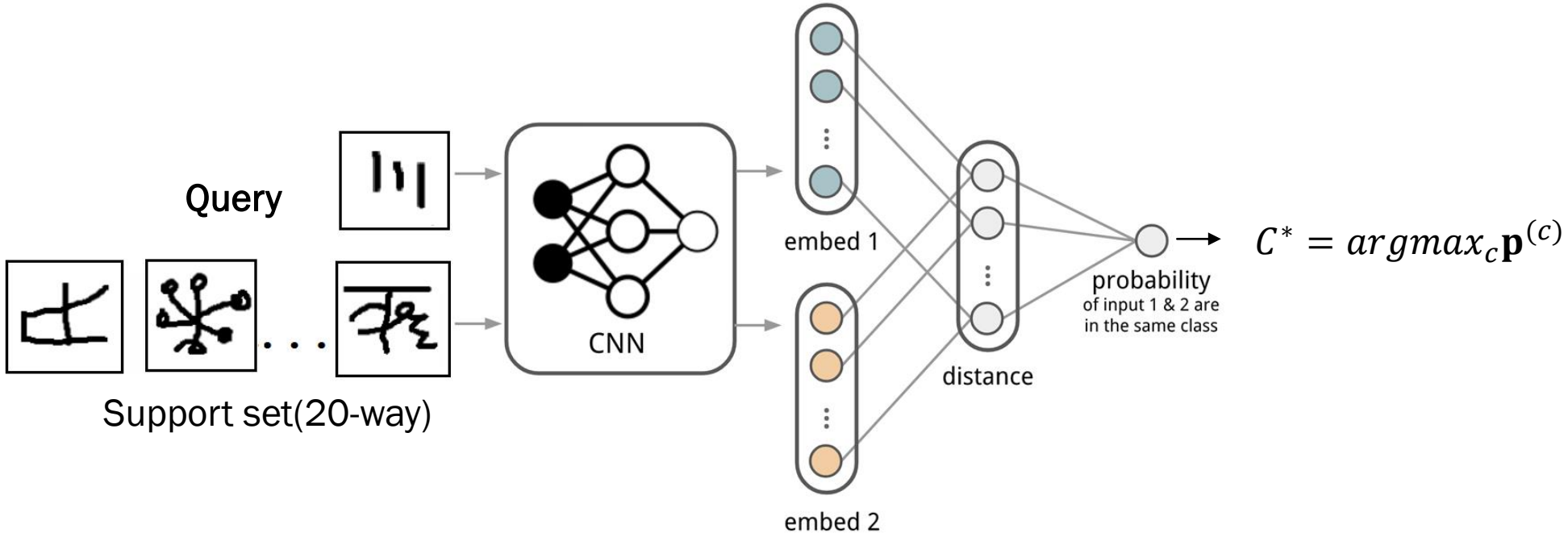


ONE-SHOT LEARNING DATASET

- In this paper, they do 20-way one-shot learning.
- Like the Verification network, 60% of the total data is used as training data.
  - Training set : 30 alphabets out of 50
  - Validation set(evaluation set) : 10 alphabets out of 50
  - Test set : 10 alphabets out of 50

ONE-SHOT LEARNING DATASET

- Validation and test is as follows:
  - Select 2 out of 20 drawers.
  - This is a method of selecting the most similar image by comparing the image created by the first drawer (Query) and the image set (Support set, 20-way) of the second drawer.
  - Repeat the above procedure twice for all alphabets.
  - That is, try 40 times per alphabet.
  - There are 10 evaluation alphabets and test alphabets → 400 trials and tests.



EXPERIMENTS

- 3.5% less accurate than human measurements.
- 3.2% less accurate than Hierarchical Bayesian Program Learning.
  - " While HBPL exhibits stronger results overall, our top-performing convolutional network did not include any extra prior knowledge about characters or strokes such as generative information about the drawing process. This is the primary advantage of our model."

Table 2. Comparing best one-shot accuracy from each type of network against baselines.

Method	Test
Humans	95.5
Hierarchical Bayesian Program Learning	95.2
Affine model	81.8
Hierarchical Deep	65.2
Deep Boltzmann Machine	62.0
Simple Stroke	35.2
1-Nearest Neighbor	21.7
Siamese Neural Net	58.3
Convolutional Siamese Net	92.0

→ 논문의 알고리즘

MNIST ONE-SHOT TRIAL

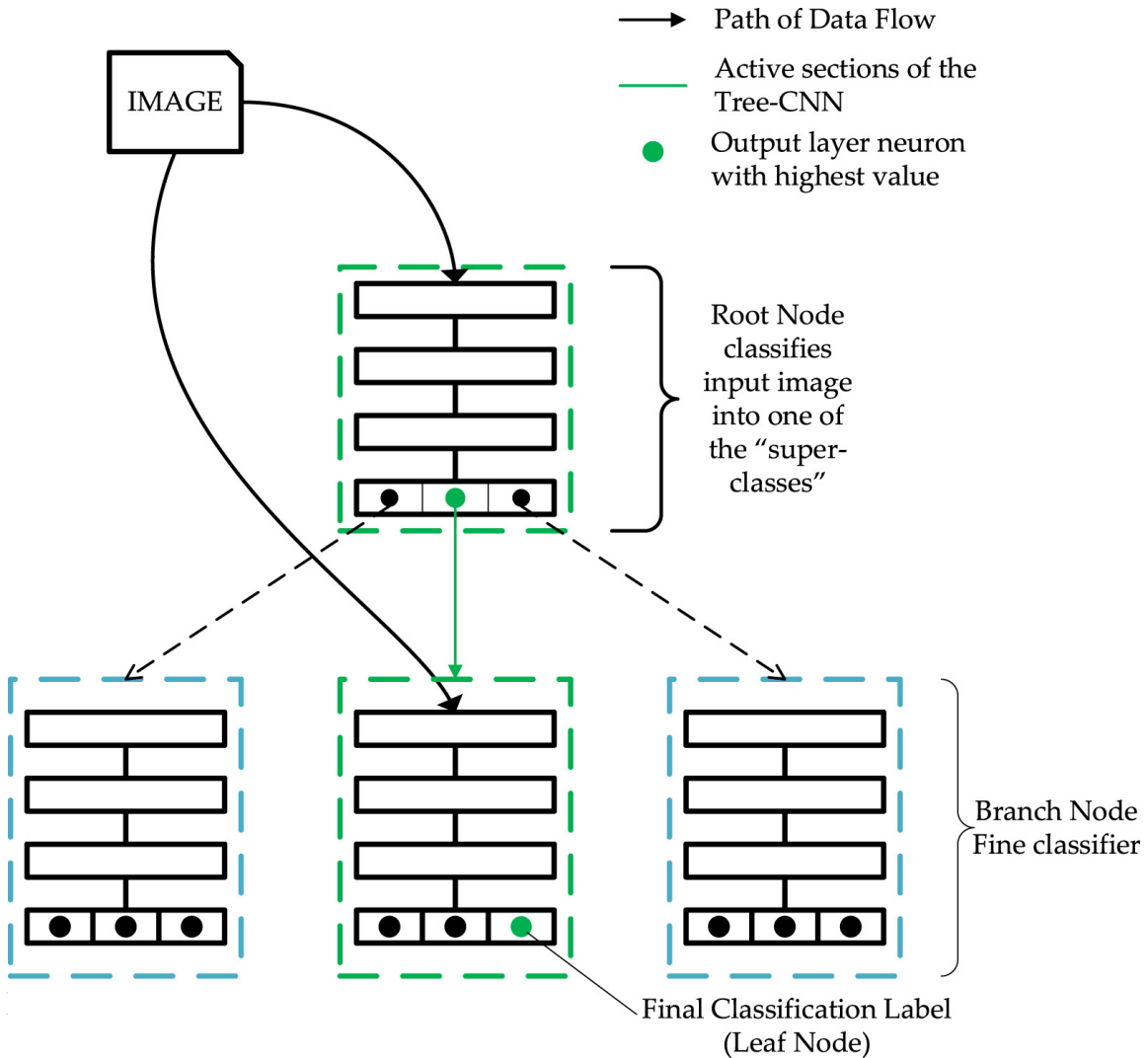
- This is to check how much MNIST generalized models learned with Omniglot.
- In the training set, no fine-tunning was performed and only the MNIST test set was used.

Table 3. Results from MNIST 10-versus-1 one-shot classification task.

Method	Test
1-Nearest Neighbor	26.5
Convolutional Siamese Net	70.3

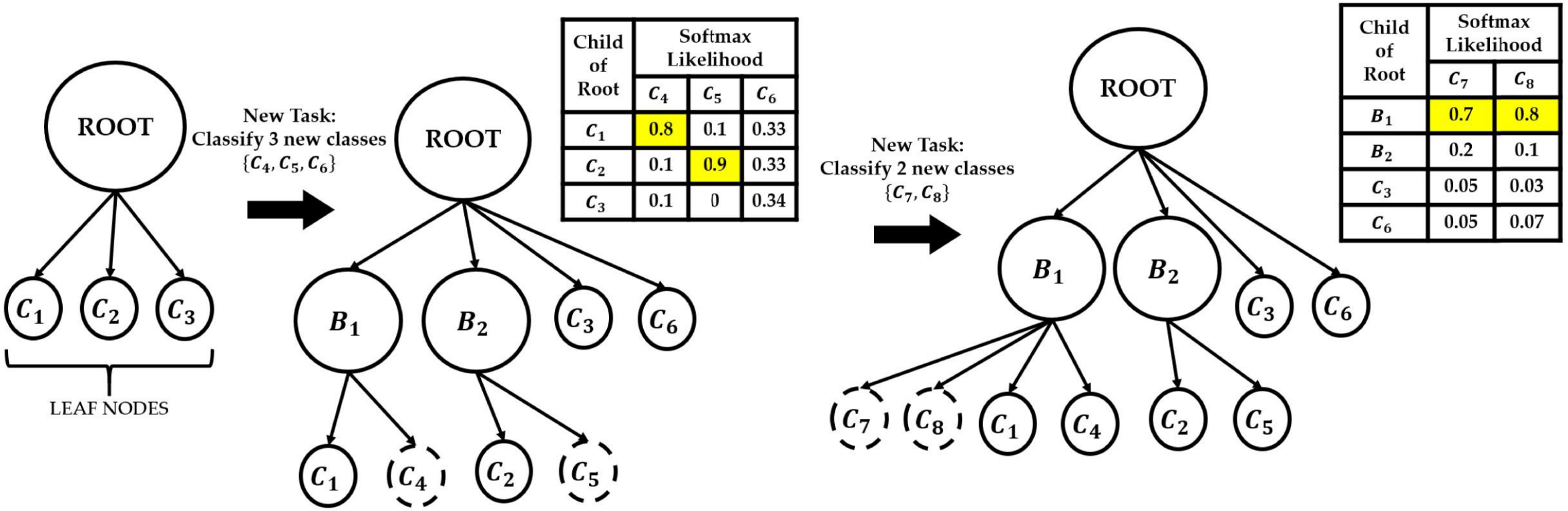
IDEA

- 위의 metric based few-shot learning과 Tree-CNN을 결합하면 더 나은 결과가 나올 것 같다.



IDEA

- 위의 metric based few-shot learning과 Tree-CNN을 결합하면 더 나은 결과가 나올 것 같다.



Child of Root	Softmax Likelihood		
	C_4	C_5	C_6
C_1	0.8	0.1	0.33
C_2	0.1	0.9	0.33
C_3	0.1	0	0.34

Child of Root	Softmax Likelihood	
	C_7	C_8
B_1	0.7	0.8
B_2	0.2	0.1
C_3	0.05	0.03
C_6	0.05	0.07

Initial Stage 0:
A network trained to classify 3 classes $\{C_1, C_2, C_3\}$

Incremental Learning Stage 1:
The Tree-CNN adds the 3 new classes as new nodes to the Root depending on their Softmax Likelihood value for each of the leaf nodes. B_1 and B_2 are branch nodes created.

Incremental Learning Stage 2:
The Tree-CNN adds the 2 new classes as new nodes to the Root depending on their Softmax Likelihood value for each of the leaf nodes. Only Root and B_1 needs to be retrained. B_2 does not change

IDEA

- 기존 Tree-CNN은 incremental learning과 데이터 분류에서 의미를 갖는다.
- 하지만 많은 데이터가 필요하며, 다음 branch node로 이동할 때 CNN dence layer의 단점을 고스란히 전수받는다.(under/overfitting, 많은 layer 필요 등등)

IDEA

- 위의 단점을 보완하여 metric based few-shot learning과 결합하면 적은 데이터로 학습이 되며, 다음 branch node로 이동할 때 데이터의 거리를 기준으로 다음 branch node로 이동하게 된다.
- 또한 branch node마다 서로 다른 metric based 알고리즘을 적용할 수 있다.
- *추상적으로 Tree-CNN에서 다음 branch node로 이동할 때마다 이미 학습된 데이터들의 거리가 가까워지기 때문에 서로 다른 class들끼리 서로 거리를 조절하게 되면 성능이 개선될 것 같다.