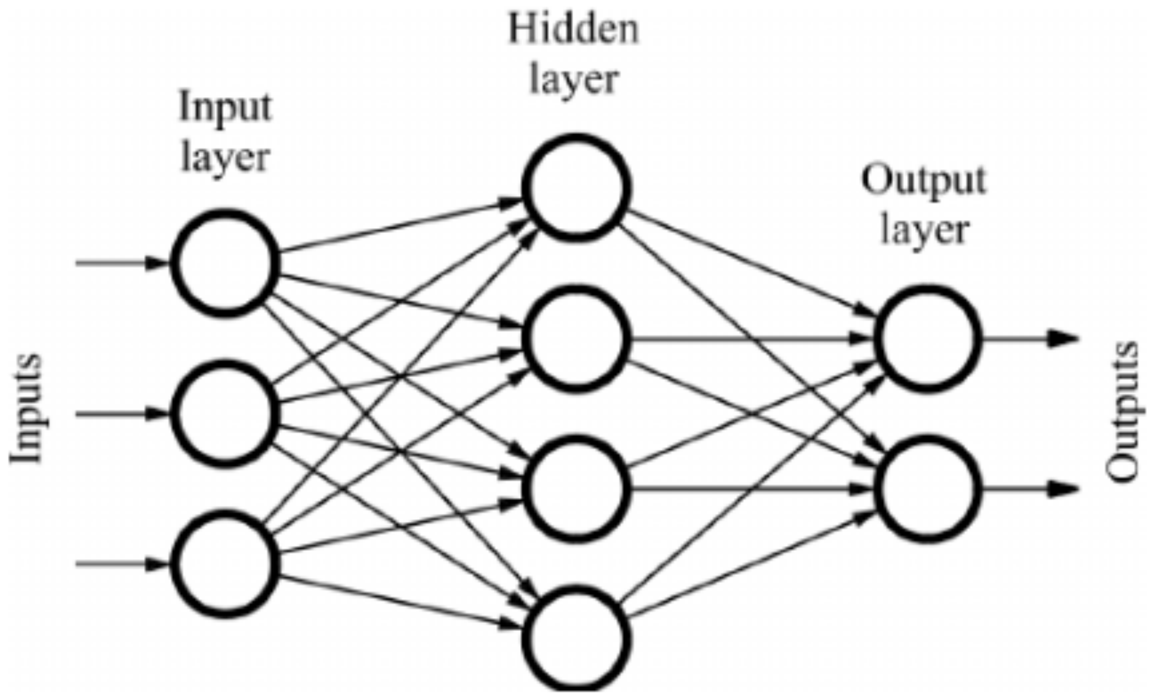
Convolutional Neural Network

Lecturer：Shi Changtai

CONTENTS

01 Neural Network and BP Algorithm

01 Neural Network and BP Algorithm



✓ Artificial neural networks consist of three layers (input layer, hidden layer and output layer), each layer is made up of adaptive processing units, called neurons, which are interconnected.

✓ A neuron is a generic computational unit that takes n inputs and produces a single output. What differentiates the outputs of different neurons is their parameters (also referred to as their weights). One of the most popular choices for neurons is the "sigmoid"

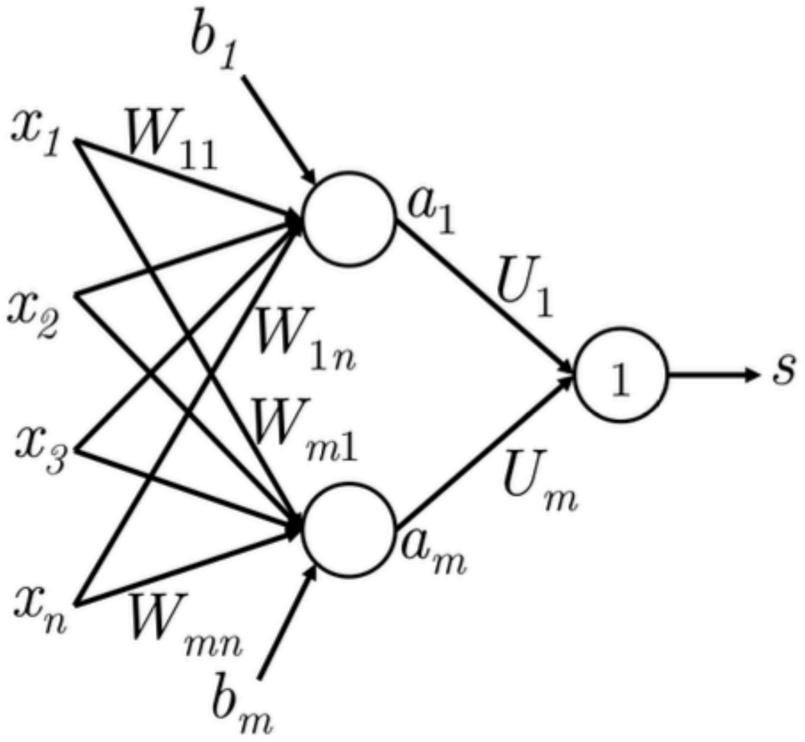
$$a = \frac{1}{1 + \exp(-(w^T x + b))}$$

Neural Network and BP Algorithm

$$z = Wx + b$$

$$a = \sigma(z)$$

$$s = U^T a$$



Like most machine learning models, neural networks also need an optimization objective, a measure of error or goodness which we want to minimize or maximize respectively. Here, we use a popular error metric known as the maximum margin objective. The idea behind using this objective is to ensure that the score computed for "true" labeled data points is higher than the score computed for "false" labeled data points.

Minimize J = $S_{false} - S_{true}$

Minimize J = max($S_{false} - S_{true}$, 0)

Minimize J = max($S_{false} - S_{true} + \Delta$, 0)

Minimize J = max($S_{false} - S_{true} + 1$, 0)

Only for binary classification!

How about the multi-classification

Neural Network and BP Algorithm

Back Propagation Algorithm

Chain-Rule based

A kind of Gradient Descent

$$\theta^{(t+1)} = \theta^{(t)} - \alpha \nabla_{\theta^{(t)}} J$$

$$\frac{\partial s}{\partial W_{ij}^{(1)}} = \frac{\partial W^{(2)} a^{(2)}}{\partial W_{ij}^{(1)}} = \frac{\partial W_i^{(2)} a_i^{(2)}}{\partial W_{ij}^{(1)}} = W_i^{(2)} \frac{\partial a_i^{(2)}}{\partial W_{ij}^{(1)}}$$

$$\Rightarrow W_i^{(2)} \frac{\partial a_i^{(2)}}{\partial W_{ij}^{(1)}} = W_i^{(2)} \frac{\partial a_i^{(2)}}{\partial z_i^{(2)}} \frac{\partial z_i^{(2)}}{\partial W_{ij}^{(1)}}$$

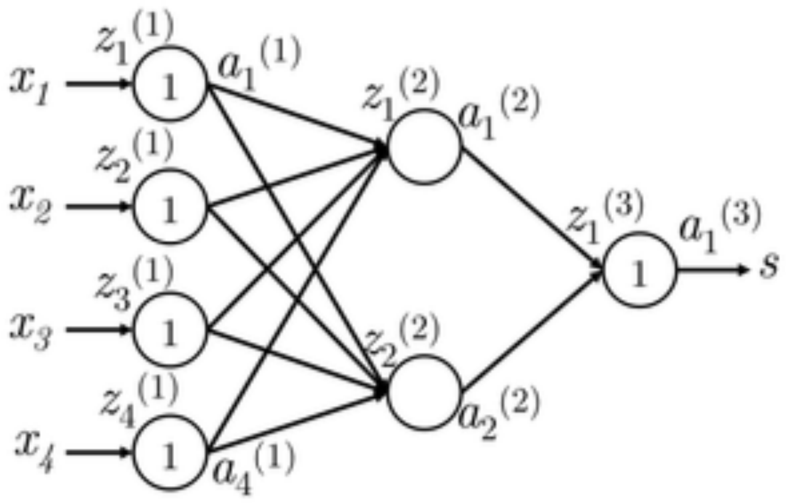
$$= W_i^{(2)} \frac{f(z_i^{(2)})}{\partial z_i^{(2)}} \frac{\partial z_i^{(2)}}{\partial W_{ij}^{(1)}}$$

$$= W_i^{(2)} f'(z_i^{(2)}) \frac{\partial z_i^{(2)}}{\partial W_{ij}^{(1)}}$$

$$= W_i^{(2)} f'(z_i^{(2)}) \frac{\partial}{\partial W_{ij}^{(1)}} (b_i^{(1)} + a_1^{(1)} W_{i1}^{(1)} + a_2^{(1)} W_{i2}^{(1)} + a_3^{(1)} W_{i3}^{(1)} + a_4^{(1)} W_{i4}^{(1)})$$

$$= W_i^{(2)} f'(z_i^{(2)}) \frac{\partial}{\partial W_{ij}^{(1)}} (b_i^{(1)} + \sum_k a_k^{(1)} W_{ik}^{(1)})$$

$$= W_i^{(2)} f'(z_i^{(2)}) a_j^{(1)}$$

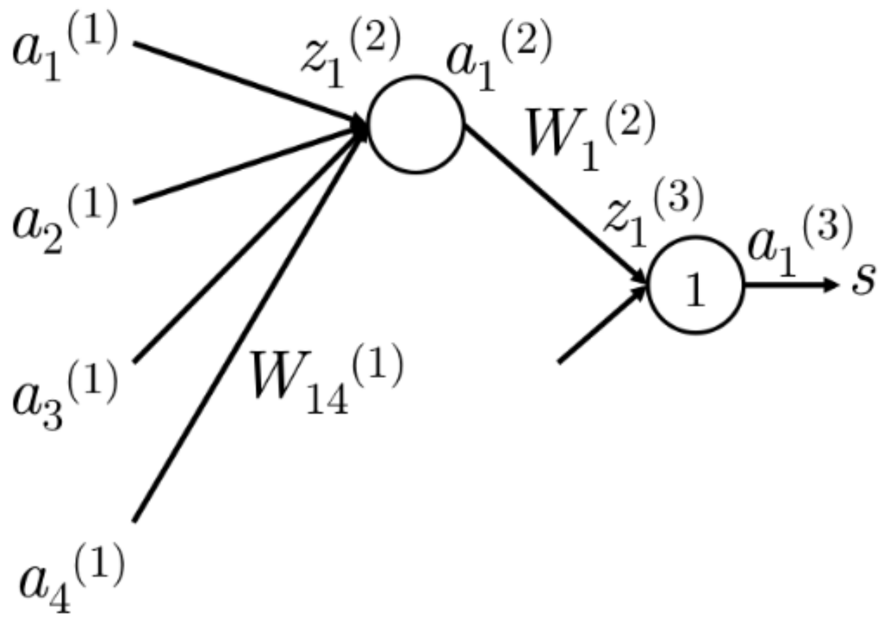


Neural Network and BP Algorithm

Back Propagation Algorithm

"error sharing/distribution" interpretation

$$\theta^{(t+1)} = \theta^{(t)} - \alpha \nabla_{\theta^{(t)}} J$$

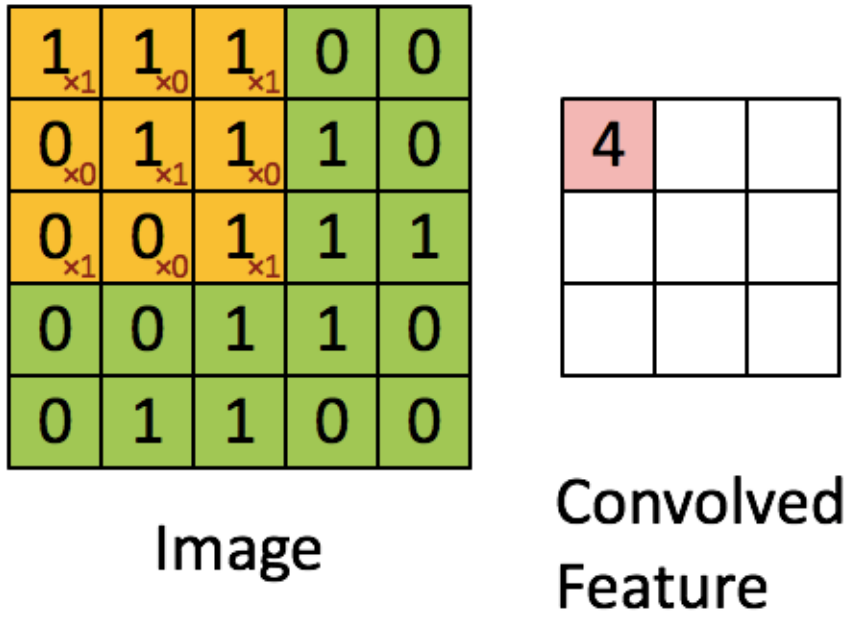


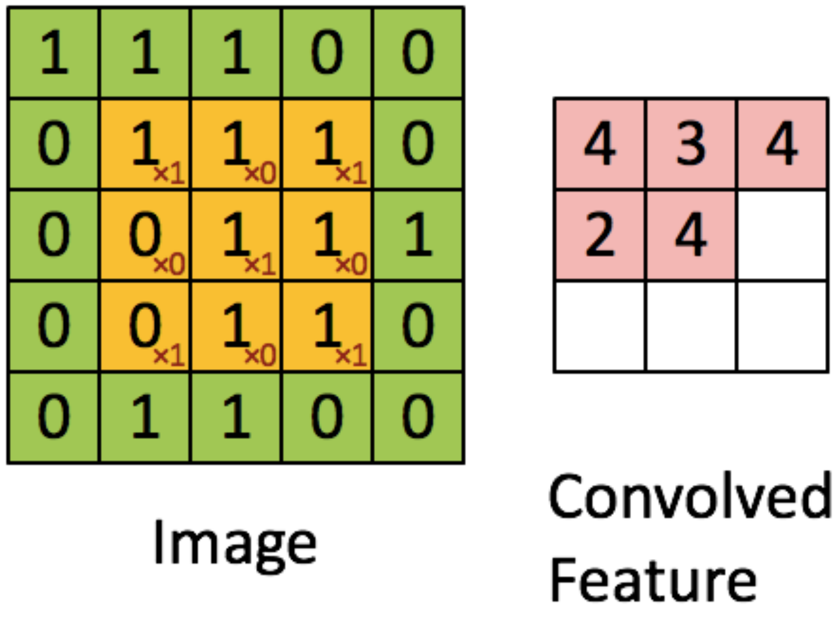
1. Start with error signal 1.
2. Multiple error by local gradient of the neuron. (still 1)
3. Distribute error signal according to the parameter weight $(W_1^{(2)})$
4. Move the error across the neuron. Also multiply by local gradient. $f'(z_1^{(2)}) * W_1^{(2)}$
5. Distribute the error. $W_1^{(2)} * f'(z_1^{(2)}) * a_4^{(1)}$

02 Convolutional Neural Network

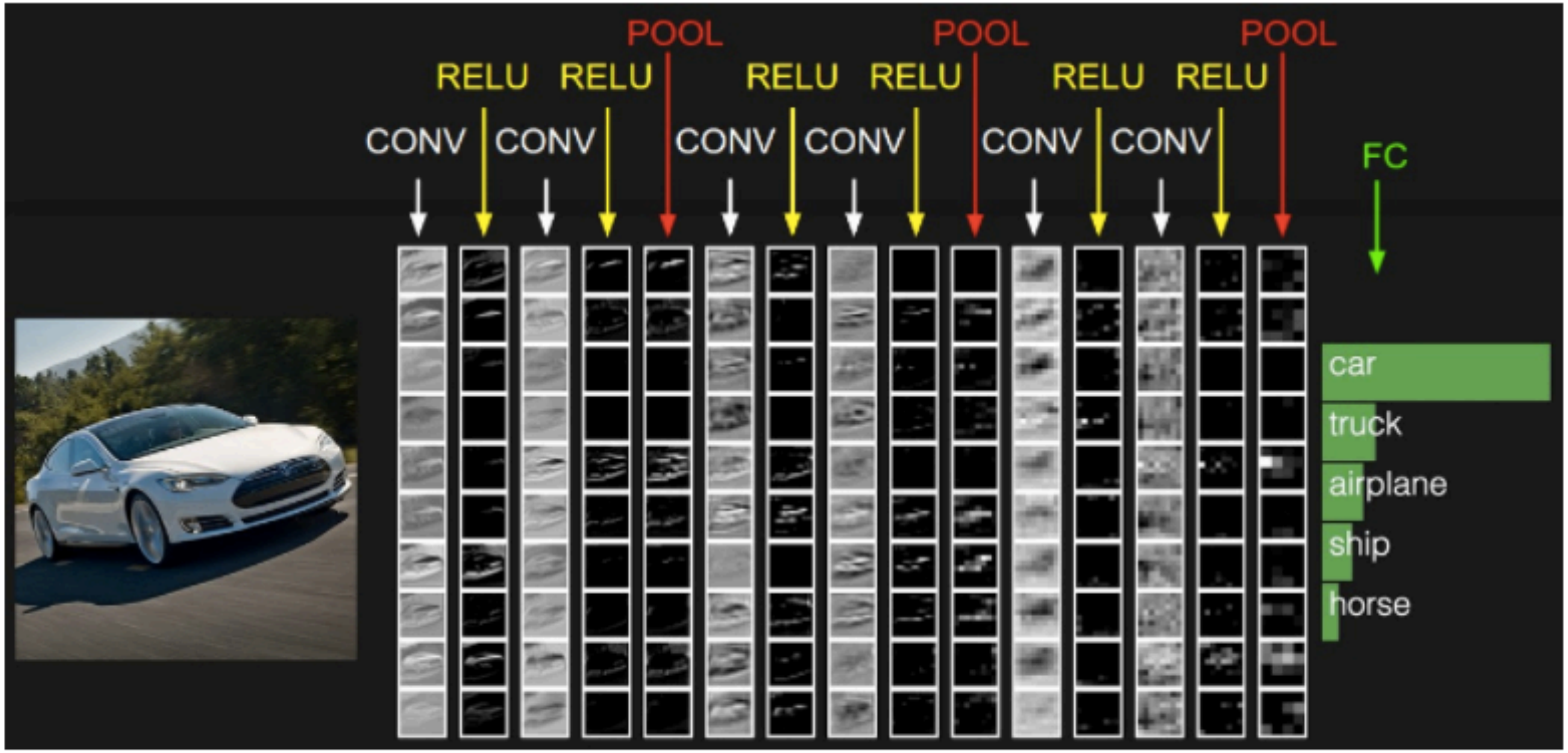
What is convolution?

What is the effect of convolution?
https://docs.gimp.org/en/plug-in-convmatrix.html





Convolutional Neural Network



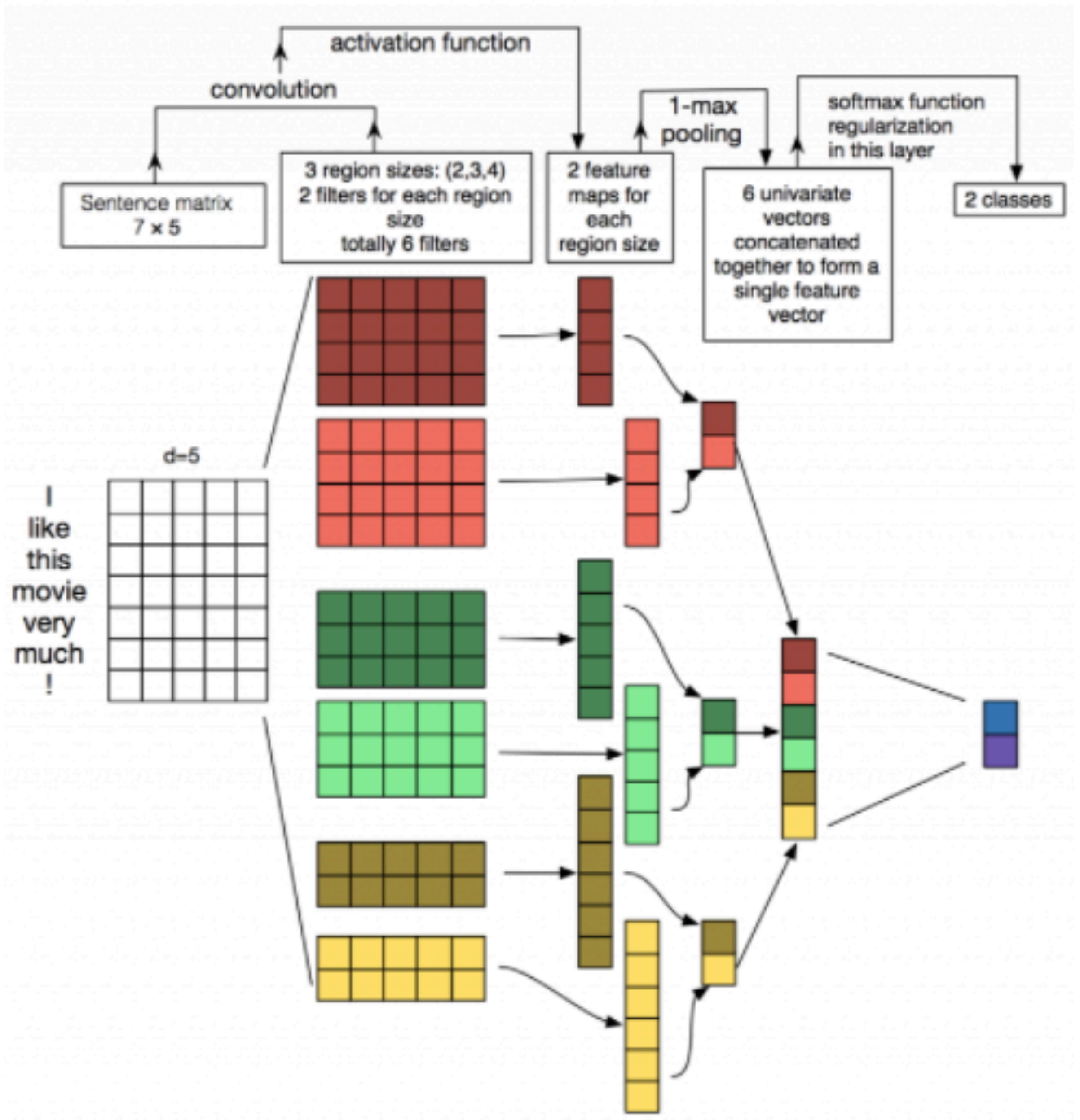
02 Convolutional Neural Network

In NLP we typically use filters that slide over full rows of the matrix. The "width" of our filters is usually the same as the width of the input matrix. (height 2-5)

Each filter *gathers some local information*. *Mapping* lower-level features into higher-level representation.

CV: Edges -> Shape -> object (face shape)

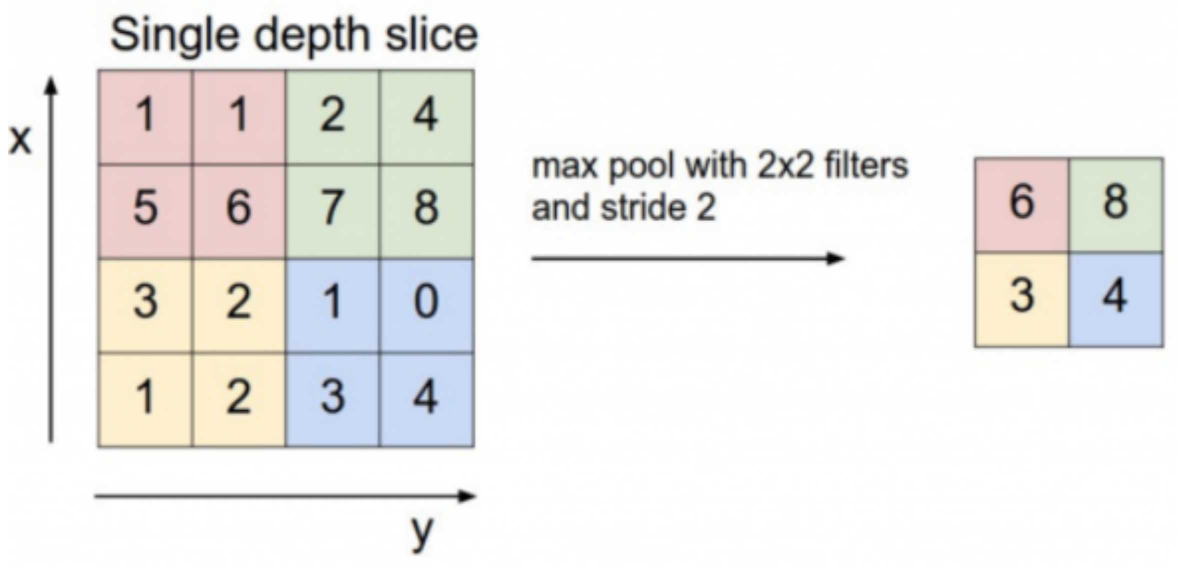
NLP: represent N-grams


activation function
convolution
1-max pooling
softmax function regularization in this layer
Sentence matrix 7 × 5
3 region sizes: (2,3,4) 2 filters for each region size totally 6 filters
2 feature maps for each region size
6 univariate vectors concatenated together to form a single feature vector
2 classes
d=5
I
like
this
movie
very
much
!

Convolutional Neural Network

Convolution neural network?

Convolution layer + Pooling layer



Max pooling in CNN. Source: http://cs231n.github.io/convolutional-networks/#pool

The meaning of Pooling:

Fix output size, allow us to use variable input sentences.

In computer vision, also provide invariance to shifting and rotation. Can recognize the object even it changes the position in an image.

03 Some Practical Tricks

Some Practical Tricks

$$\theta_{t,i} = \theta_{t-1,i} - \frac{\alpha}{\sqrt{\sum_{\tau=1}^{t} g_{\tau,i}^2}} g_{t,i} \text{ where } g_{t,i} = \frac{\partial}{\partial \theta_i^t} J_t(\theta)$$

Neuron Unit

Sigmoid, Relu, Leaky Relu

Learning Rate

$$\alpha(t) = \frac{\alpha_0 \tau}{\max(t, \tau)}$$

Small->Slow   Large->Diverge

1. Decrease with time
2. AdaGrad, parameters updated fewer in the past have larger Alpha
3. Momentum

Parameter Initialization

The empirical findings suggest that for sigmoid units.
Lower error, faster convergence.

$$W \sim U\left[-\sqrt{\frac{6}{n^{(l)} + n^{(l+1)}}}, \sqrt{\frac{6}{n^{(l)} + n^{(l+1)}}}\right]$$

Where $n^{(l)}$ is the number of input units to W (fan-in)

and $n^{(l+1)}$ is the number of output units from W (fan-out)

Regularization

L1, L2 parameter regularization
Early Stopping
Dropout
Noise Injection (Regularization and Bayesian Uncertainty)
Multi-task learning
Ensamble

04 Apply CNN in News Classification

04 Apply CNN in News Classification

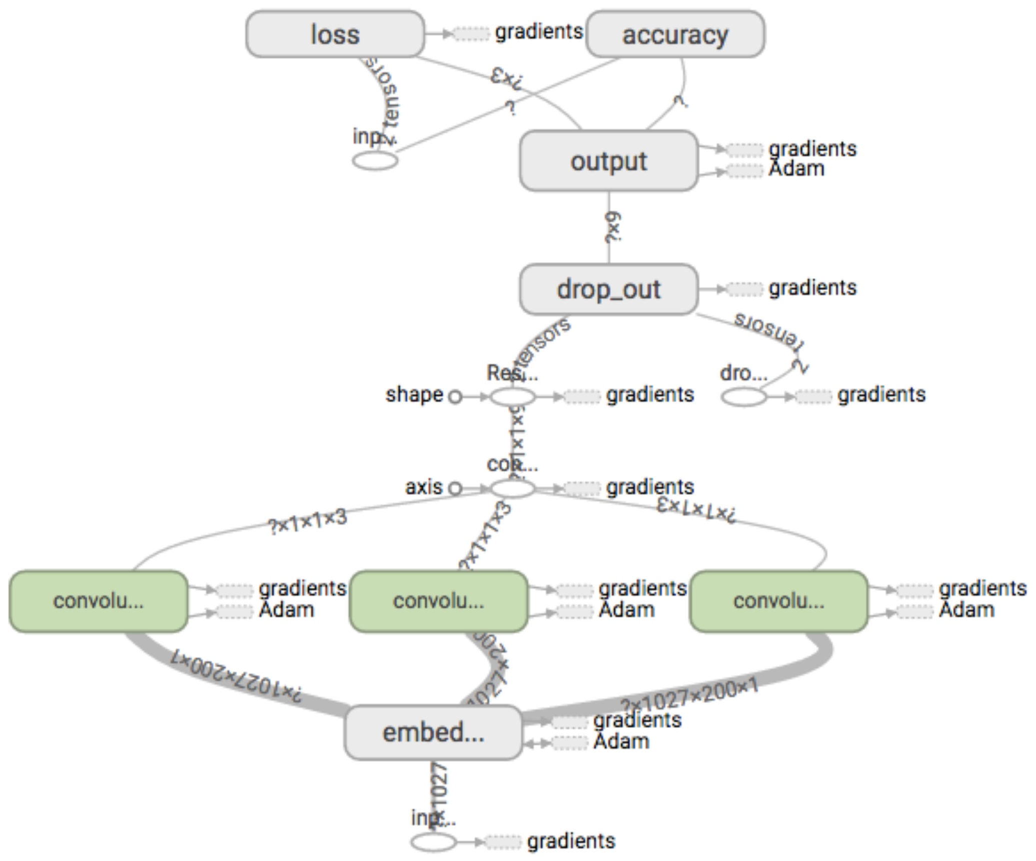
Data Format

```
<doc>
<url>Page_URL</url>
<docno>Page_ID</docno>
<contenttitle>Title</contenttitle>
<content>Content</content>
</doc>
```

Health: 10,000 items

Auto: 10,000 items

Business: 10,000 items



Parameter Setting

embedding_size:	200
keep_prob:	0.7
batch_size:	500
evaluate_step:	100
filter_size:	2,3,4
train_dev_split_ratio:	0.99
topic_num:	12
epoch_size:	10
filter_num:	3
shuffle_input:	False

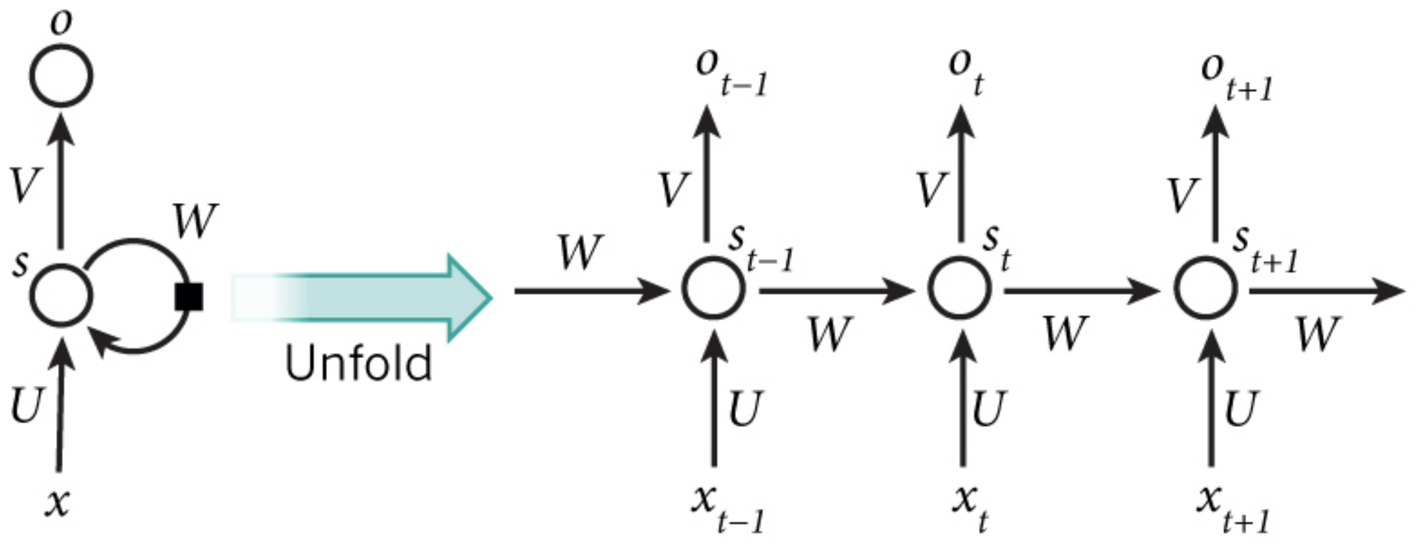
05 Experiment Result

05 Experiment Result

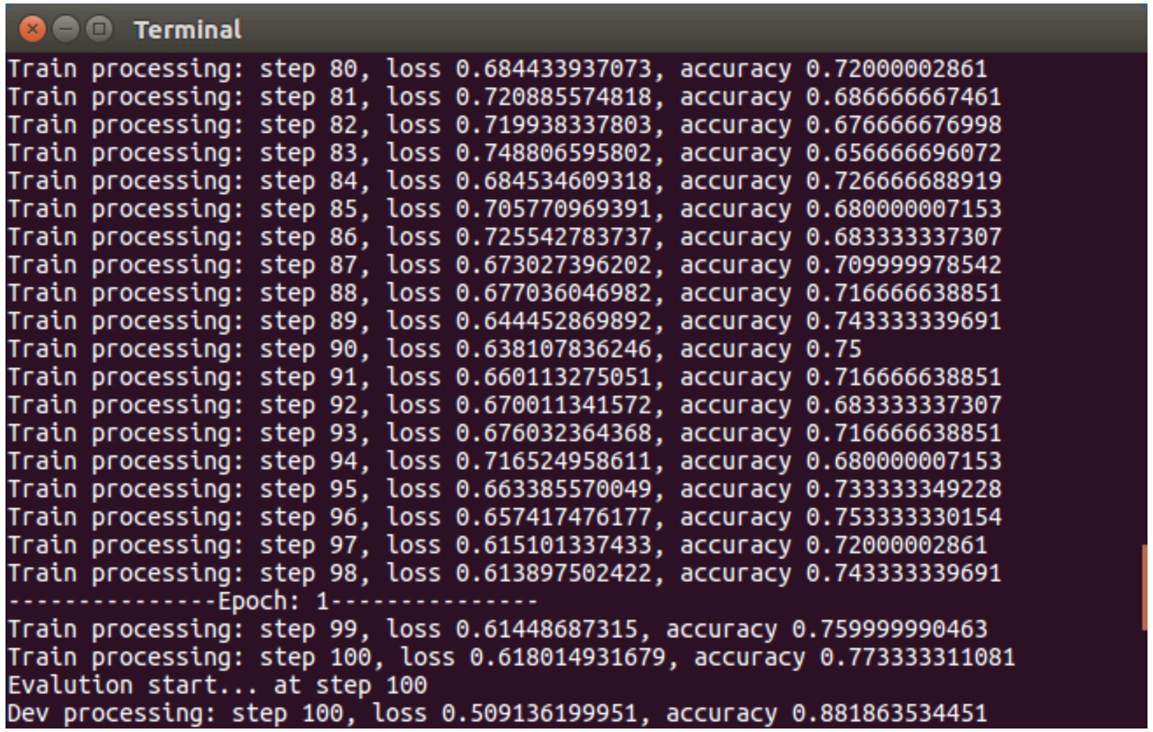
CNN performs very good in computer vision because it ignore object positions and extract local feature. However, In natural language processing, word position in one sentence is important.

RNN simulates human's from-left-to-right reading habit but CNN smooth and ignore word order.

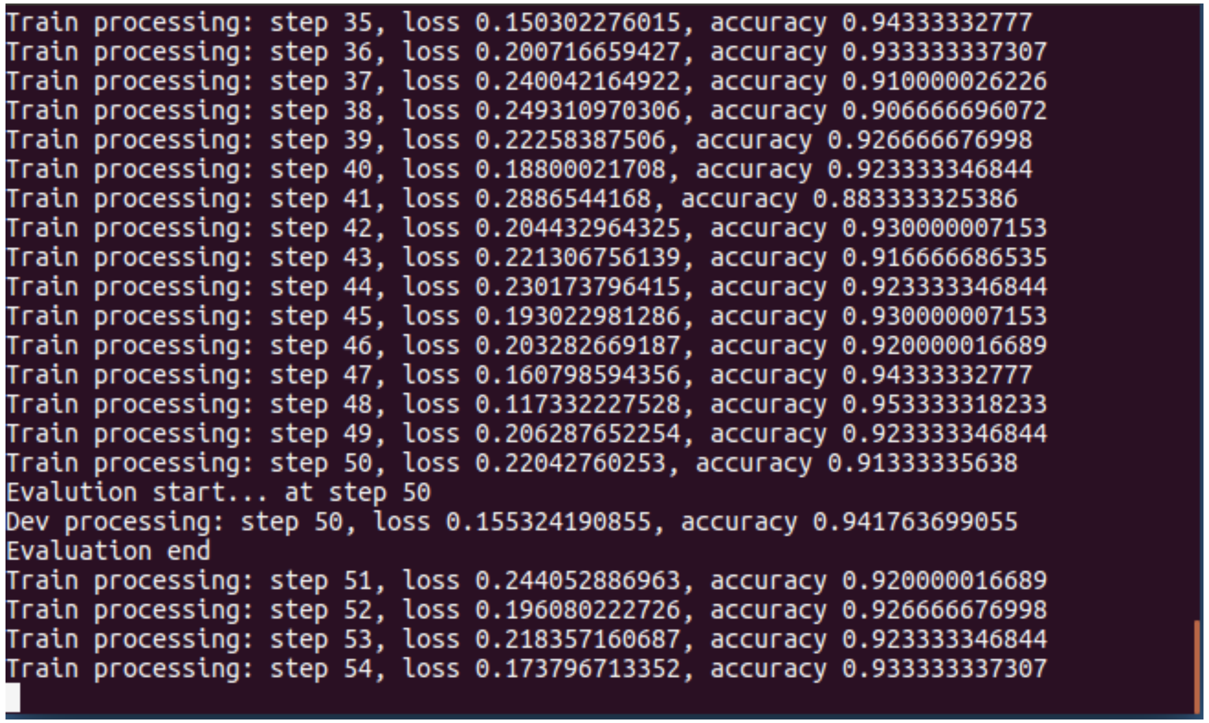
Will RNN perform better?



05 Experiment Result



```
Train processing: step 80, loss 0.684433937073, accuracy 0.72000002861
Train processing: step 81, loss 0.720885574818, accuracy 0.686666667461
Train processing: step 82, loss 0.719938337803, accuracy 0.676666676998
Train processing: step 83, loss 0.748806595802, accuracy 0.656666696072
Train processing: step 84, loss 0.684534609318, accuracy 0.726666688919
Train processing: step 85, loss 0.705770969391, accuracy 0.680000007153
Train processing: step 86, loss 0.725542783737, accuracy 0.683333337307
Train processing: step 87, loss 0.673027396202, accuracy 0.709999978542
Train processing: step 88, loss 0.677036046982, accuracy 0.716666638851
Train processing: step 89, loss 0.644452869892, accuracy 0.743333339691
Train processing: step 90, loss 0.638107836246, accuracy 0.75
Train processing: step 91, loss 0.660113275051, accuracy 0.716666638851
Train processing: step 92, loss 0.670011341572, accuracy 0.683333337307
Train processing: step 93, loss 0.676032364368, accuracy 0.716666638851
Train processing: step 94, loss 0.716524958611, accuracy 0.680000007153
Train processing: step 95, loss 0.663385570049, accuracy 0.733333349228
Train processing: step 96, loss 0.657417476177, accuracy 0.753333330154
Train processing: step 97, loss 0.615101337433, accuracy 0.72000002861
Train processing: step 98, loss 0.613897502422, accuracy 0.743333339691
---------------Epoch: 1---------------
Train processing: step 99, loss 0.61448687315, accuracy 0.759999990463
Train processing: step 100, loss 0.618014931679, accuracy 0.773333311081
Evalution start... at step 100
Dev processing: step 100, loss 0.509136199951, accuracy 0.881863534451
```



```
Train processing: step 35, loss 0.150302276015, accuracy 0.94333332777
Train processing: step 36, loss 0.200716659427, accuracy 0.933333337307
Train processing: step 37, loss 0.240042164922, accuracy 0.910000026226
Train processing: step 38, loss 0.249310970306, accuracy 0.906666696072
Train processing: step 39, loss 0.22258387506, accuracy 0.926666676998
Train processing: step 40, loss 0.18800021708, accuracy 0.923333346844
Train processing: step 41, loss 0.2886544168, accuracy 0.883333325386
Train processing: step 42, loss 0.204432964325, accuracy 0.930000007153
Train processing: step 43, loss 0.221306756139, accuracy 0.916666686535
Train processing: step 44, loss 0.230173796415, accuracy 0.923333346844
Train processing: step 45, loss 0.193022981286, accuracy 0.930000007153
Train processing: step 46, loss 0.203282669187, accuracy 0.920000016689
Train processing: step 47, loss 0.160798594356, accuracy 0.94333332777
Train processing: step 48, loss 0.117332227528, accuracy 0.953333318233
Train processing: step 49, loss 0.206287652254, accuracy 0.923333346844
Train processing: step 50, loss 0.22042760253, accuracy 0.91333335638
Evalution start... at step 50
Dev processing: step 50, loss 0.155324190855, accuracy 0.941763699055
Evaluation end
Train processing: step 51, loss 0.244052886963, accuracy 0.920000016689
Train processing: step 52, loss 0.196080222726, accuracy 0.926666676998
Train processing: step 53, loss 0.218357160687, accuracy 0.923333346844
Train processing: step 54, loss 0.173796713352, accuracy 0.933333337307
```

CNN is so fast that takes only 2 minutes to finish 50 train steps and get an accuracy of 88% on test data after one epoch.

RNN is relative slower, it takes 4 minutes to finish 50 train steps, but it gets accuracy around 90% just after a few train steps and attains 94.2% accuracy on test data after half epoch.

06 Homework

06 Homework

As mentioned in lecture,

How to derive least squares error and cross entropy using maximum likelihood method?
(Just use simple formulas and descriptions)

Reference

[1] Yoon Kim. " Convolutional Neural Networks for Sentence Classification" EMNLP 2014.

LINK: http://aclweb.org/anthology/D/D14/D14-1181.pdf

[2] Peng Wang, Jiaming Xu. "Semantic Clustering and Convolutional Neural Network for Short Text Categorization" 2015.

LINK: http://www.aclweb.org/anthology/P15-2058

Question Time