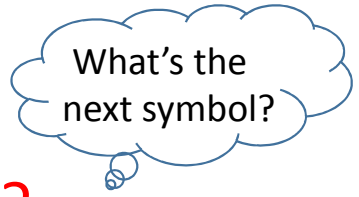
Sequence Prediction Using Neural Network Classifiers

Yanpeng Zhao

ShanghaiTech University

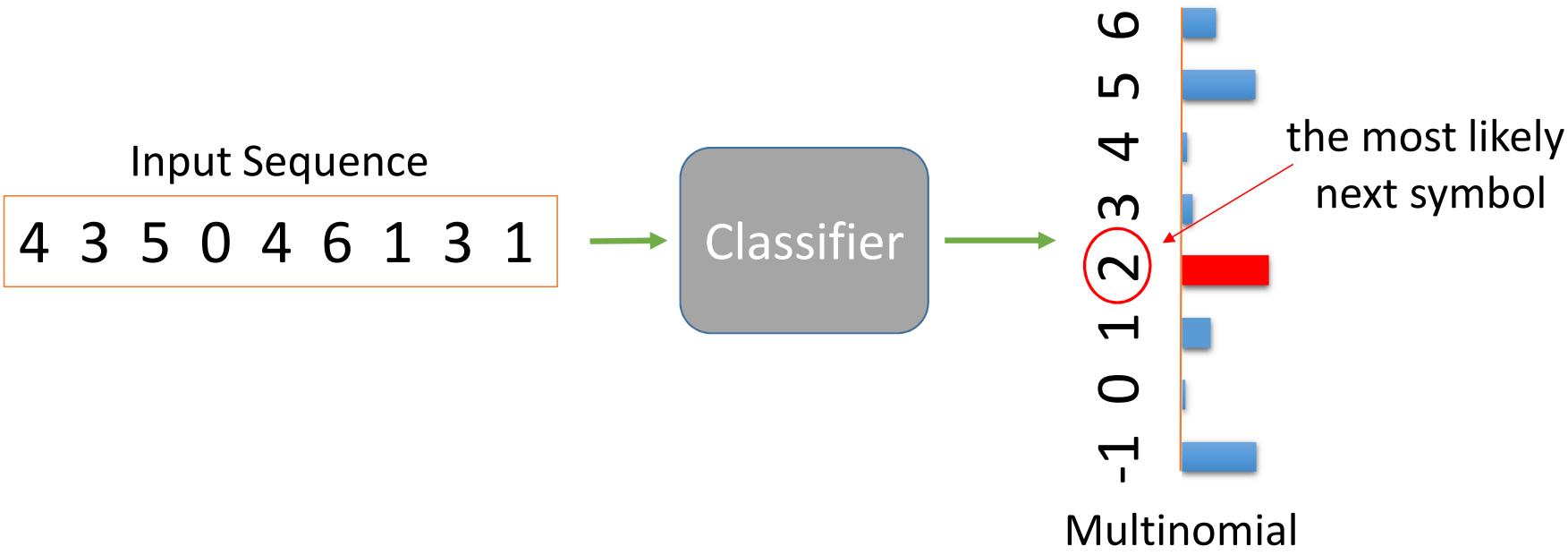
ICGI, Oct 7th, 2016, Delft, the Netherlands

Sequence Prediction



4 3 5 0 4 6 1 3 1 ?

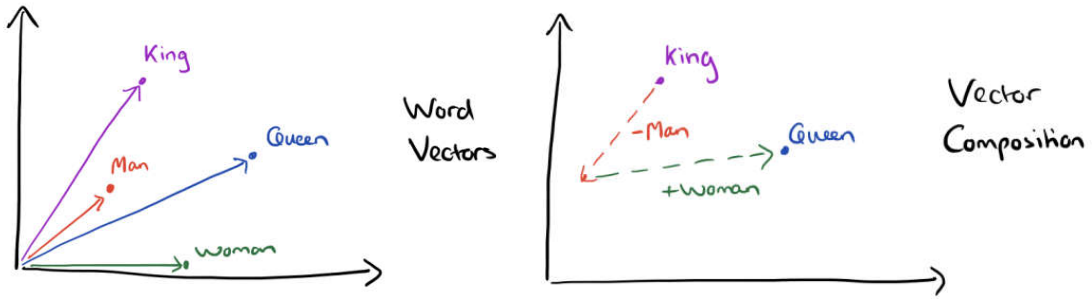
Classification Perspective



Representation of Inputs

Continuous vector representation of the discrete symbols

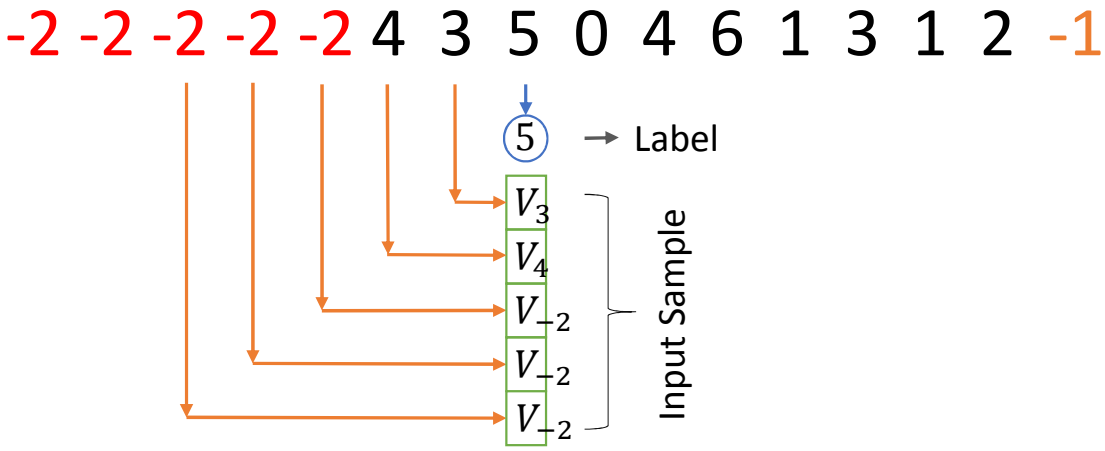
King – Man + Woman ≈ Queen



Images are from: https://blog.acolyer.org/2016/04/21/the-amazing-power-of-word-vectors/

Representation of Inputs

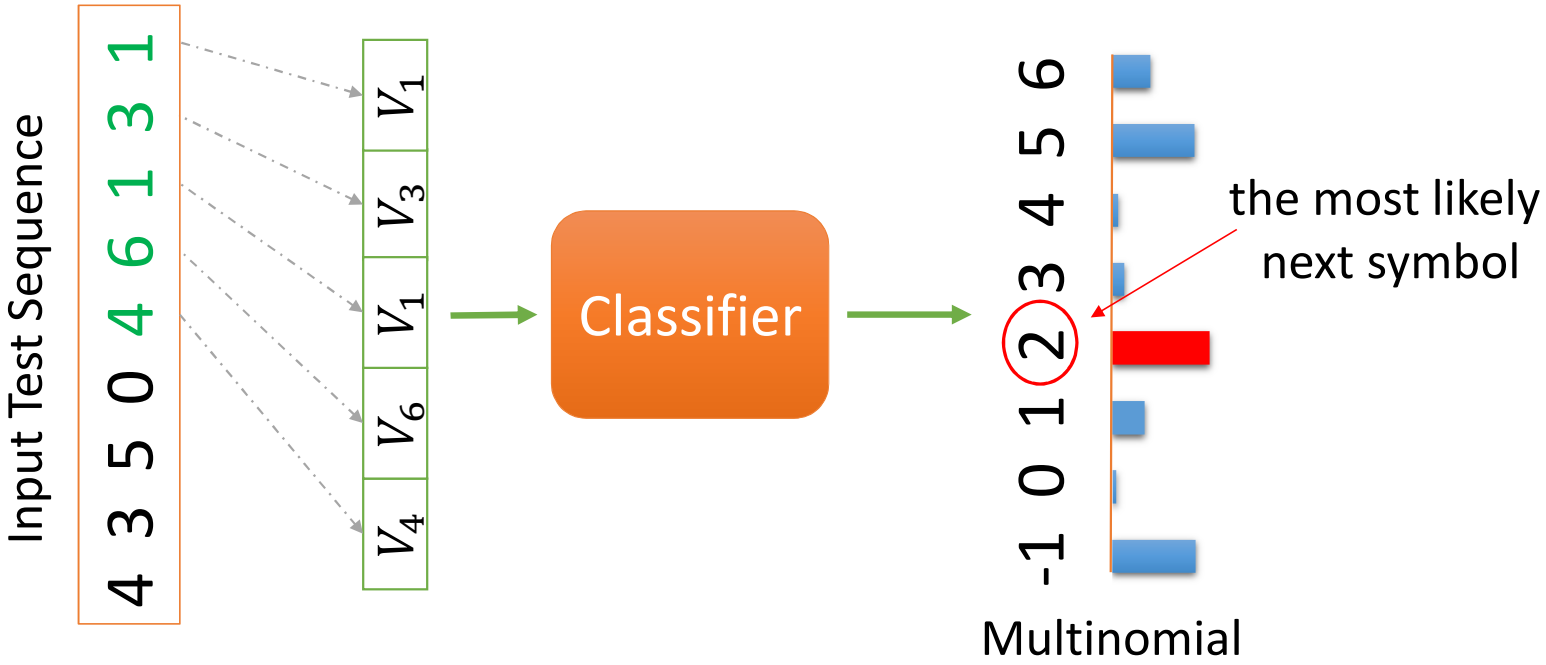
Construct inputs for classifiers using learned word vectors



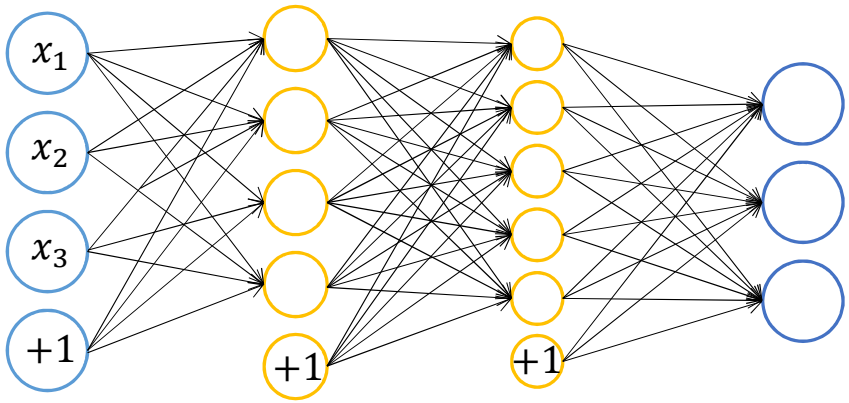
Word vectors are concatenated or stacked

Predict the next symbol from the previous $k = 15$ symbols, each represented by a 30-dimension vector

Neural Network Classifiers

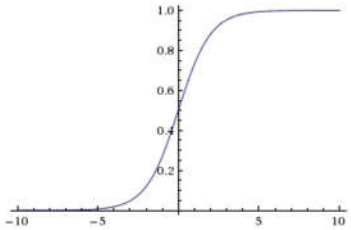


Multilayer Perceptrons (MLPs)

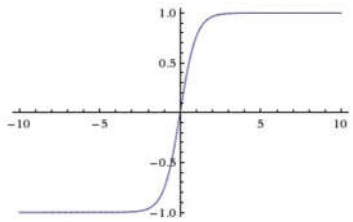


Input X Hidden Layer 1 Hidden Layer 2 Softmax Output

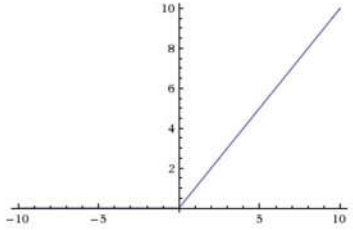
$$z_1 = [W_1\ b_1] \cdot \begin{bmatrix} X \\ 1 \end{bmatrix}$$
$$\mathrm{a}_1 = \sigma(z_1)$$
$$z_2 = [W_2\ b_2] \cdot \begin{bmatrix} a_1 \\ 1 \end{bmatrix}$$
$$\mathrm{a}_2 = \sigma(z_2)$$
$$z_3 = [W_3\ b_3] \cdot \begin{bmatrix} a_2 \\ 1 \end{bmatrix}$$
$$o = softmax(z_3)$$



$\sigma(x) = 1/(1+e^{-x})$

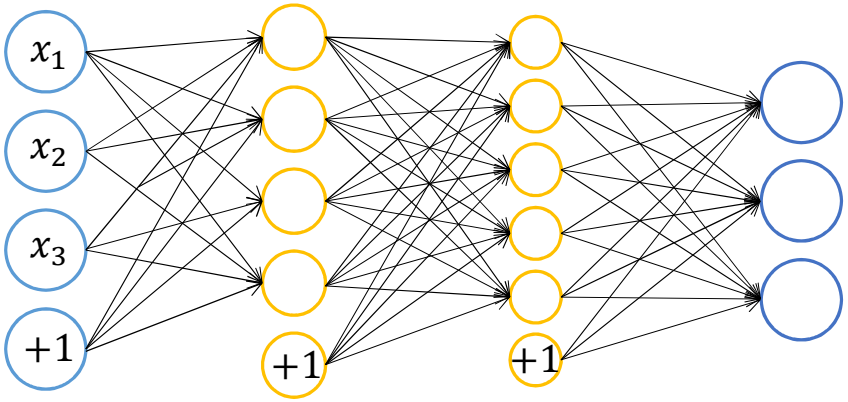


$tanh(x) = (e^{x}-e^{-x})/(e^{x}+e^{-x})$



$ReLU(x) = \max(0, x)$

Multilayer Perceptrons (MLPs)

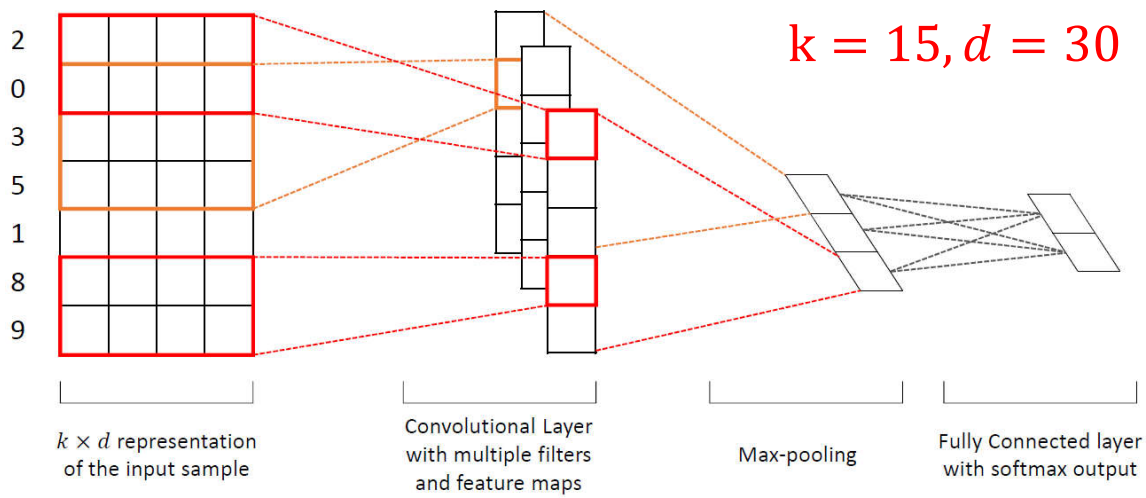


$$z_1 = [W_1\ b_1] \cdot \begin{bmatrix} X \\ 1 \end{bmatrix}$$
$$\mathrm{a}_1 = \sigma(z_1)$$
$$z_2 = [W_2\ b_2] \cdot \begin{bmatrix} a_1 \\ 1 \end{bmatrix}$$
$$\mathrm{a}_2 = \sigma(z_2)$$
$$z_3 = [W_3\ b_3] \cdot \begin{bmatrix} a_2 \\ 1 \end{bmatrix}$$
$$o = softmax(z_3)$$

$|X| = 450$: 15 symbols with a 30-dimension vector for each symbol

$|z_1| = 750$ and $|z_2| = 1000$

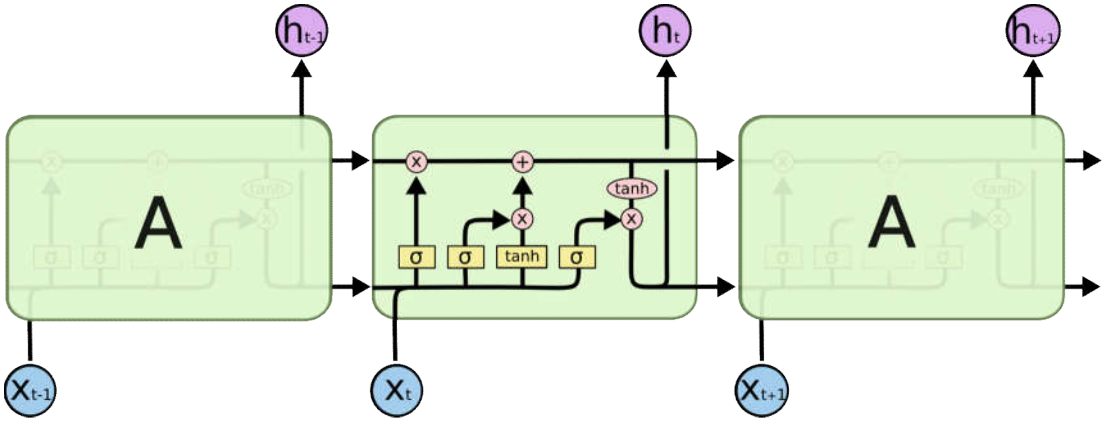
Convolutional Neural Networks (CNNs)



CNN model architecture adapted from Yoon Kim. *Convolutional neural networks for sentence classication. arXiv preprint arXiv:1408.5882*, 2014

Filter windows (height) of $10, 11, 12, 13, 14, 15$; 200 feature maps for each window

Long Short Term Memory Networks (LSTMs)



$$\begin{bmatrix} i_t \\ f_t \\ o_t \\ \widehat{C}_t \end{bmatrix} = \begin{bmatrix} \sigma \\ \sigma \\ \sigma \\ tanh \end{bmatrix} [U\ W\ b] \cdot \begin{bmatrix} h_{t-1} \\ x_t \\ 1 \end{bmatrix} \qquad \begin{aligned} C_t &= f_t \otimes C_{t-1} + i_t \otimes \widehat{C}_t \\ h_t &= o_t \otimes tanh(C_t) \end{aligned}$$

Time step is 15, and h_t of dim 32 is fed to a logistic regression classifier

Images are from: http://colah.github.io/posts/2015-08-Understanding-LSTMs/

Weighted n-Gram Model (WnGM)

$$\begin{matrix} 3 \\ 2 \\ 1 \\ 0 \\ -1 \end{matrix} \quad w_1 \cdot \begin{bmatrix} 0.11 \\ 0.24 \\ 0.21 \\ 0.26 \\ 0.18 \end{bmatrix} + w_2 \cdot \begin{bmatrix} 0.01 \\ 0.26 \\ 0.29 \\ 0.21 \\ 0.23 \end{bmatrix} + w_3 \cdot \begin{bmatrix} 0.28 \\ 0.15 \\ 0.25 \\ 0.06 \\ 0.26 \end{bmatrix} = output$$

Label $\quad n_1$-Gram $\quad n_2$-Gram $\quad n_3$-Gram

We set n to $2, 3, 4, 5, 6$ with weights $0.3, 0.2, 0.2, 0.15, 0.15$ respectively

Overview of Experiments

- Implementation
  - MLP & CNN were implemented in MxNet
  - LSTM was implemented in Tensorflow
  - https://bitbucket.org/thinkzhou/spice
- System & Hardware
  - CentOS 7.2 (64Bit) server
  - Intel Xeon Processor E5-2697 v2 @ 2.70GHz & Four Tesla K40Ms
- Time cost
  - Run all the models on all datasets in less than 16h

Detail Scores on Public Test Sets

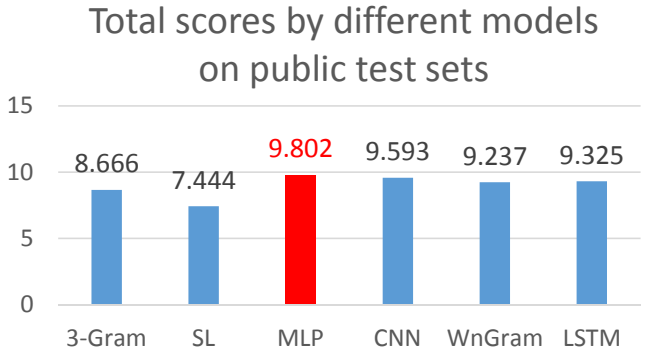
Problem ID	3-Gram	SL	MLP	CNN	WnGM	LSTM
1	0.842825	0.874877	0.894517	0.877351	0.841534	**0.903276**
2	0.817825	0.87404	0.904835	0.897068	0.817638	**0.910337**
3	0.775519	0.828139	0.86131	0.850179	0.779835	**0.874903**
4	0.537655	0.431704	0.528351	0.531538	**0.588389**	0.422016
5	0.598158	0.292452	**0.761046**	0.743044	0.690147	0.685619
6	0.674949	0.429366	0.705684	0.667631	0.759378	**0.769689**
7	0.442091	0.324983	**0.685551**	0.6506	0.503653	0.5767
8	0.590873	0.491134	**0.615426**	0.603933	0.61359	0.574858
9	0.858806	0.741451	0.921942	0.904356	0.896695	**0.952874**
10	0.413004	0.256069	**0.556514**	0.553421	0.44162	0.528451
11	0.38855	0.329707	**0.51323**	0.499009	0.441006	0.443673
12	0.700295	0.55699	0.715182	0.667296	**0.747883**	0.664733
13	0.435876	0.328572	**0.524219**	0.521	0.492081	0.385525
14	0.338599	**0.406484**	0.350904	0.357211	0.362929	0.366087
15	0.251157	**0.277731**	0.263684	0.269752	0.260136	0.266703
Total Score	8.666182	7.443699	9.802395	9.593389	9.236514	9.325444

numbers in bold indicate the best scores achieved on public test sets.

Total score on private test sets is **10.160324**

Discussion & Future Work

- MLPs
  - make the best use of the symbol order information
- CNNs
  - should use the problem-specific model architecture
  - update vectors while training
  - train a deep averaging network (DAN) [Mohit et al., 2015]
- LSTMs
- Future work
  - integrate neural networks into probabilistic grammatical models in sequence prediction



Thanks