國立臺灣大學 National Taiwan University

Distributed Representation of Word

Jau-Chi Huang, Wei-Chen Cheng,
Cheng-Yuan Liou*

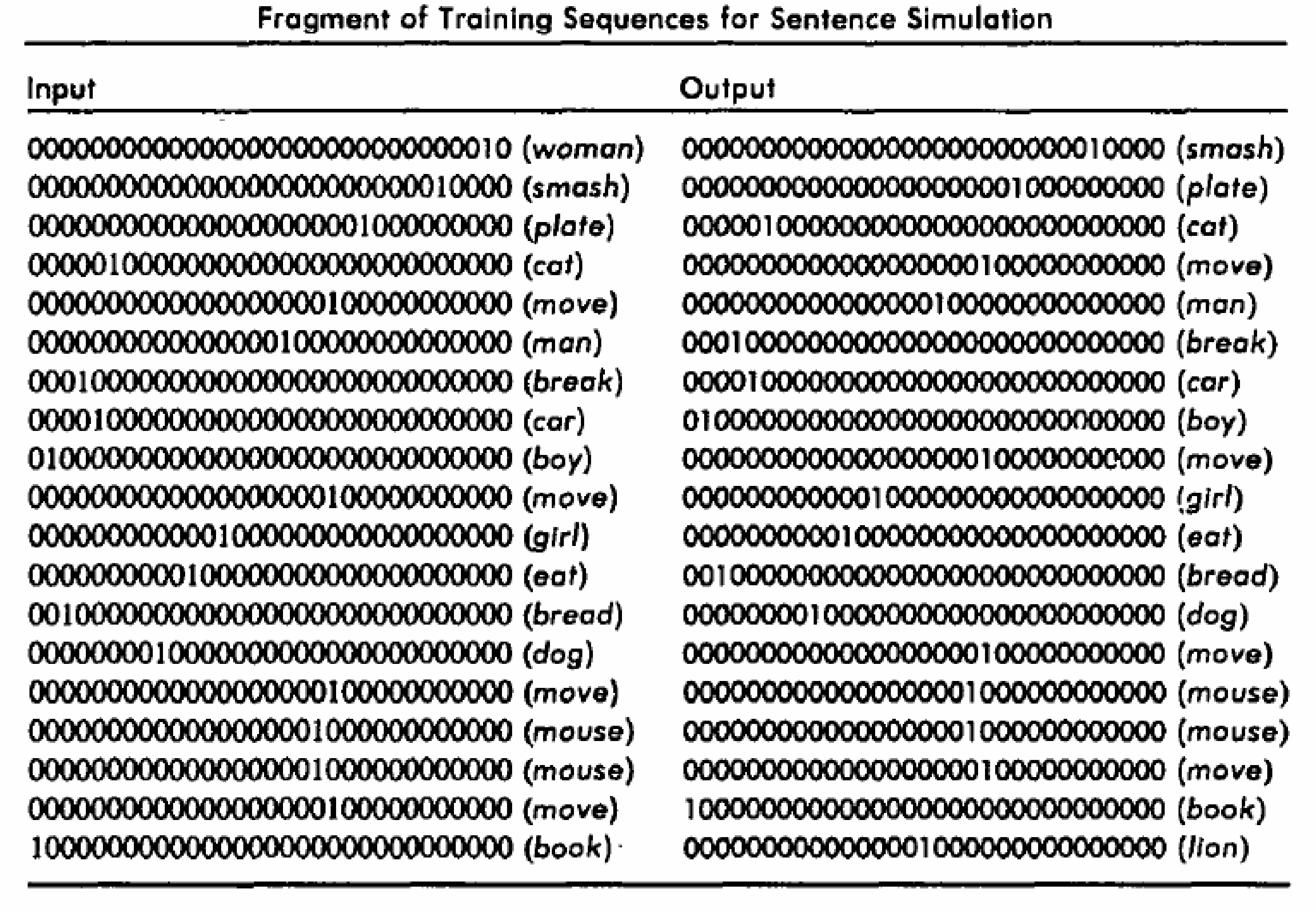
Department of Computer Science and
Information Engineering

Localist codes vs. distributed codes

- Localist codes:
  - word$_1$: [1,0,0,…,0]
  - word$_2$: [0,1,0,…,0]
  - word$_3$: [0,0,1,…,0]
  - ...
  - word$_N$: [0,0,0,…,1]
- Distributed codes:
  - word$_1$: [0.4447, 0.9218, 0.4057,…, 0.4103]
  - word$_2$: [0.6154, 0.7382, 0.9355,…, 0.8936]
  - word$_3$: [0.7919, 0.1763, 0.9169,…, 0.0579]
  - …

Discovering lexical classes from word order by Jeff. L. Elman(1990)

- Artificial simple sentences
- A small number of vocabularies
  - Only 31 different words used
- Two simple grammars
  - S+V+O
  - S+V

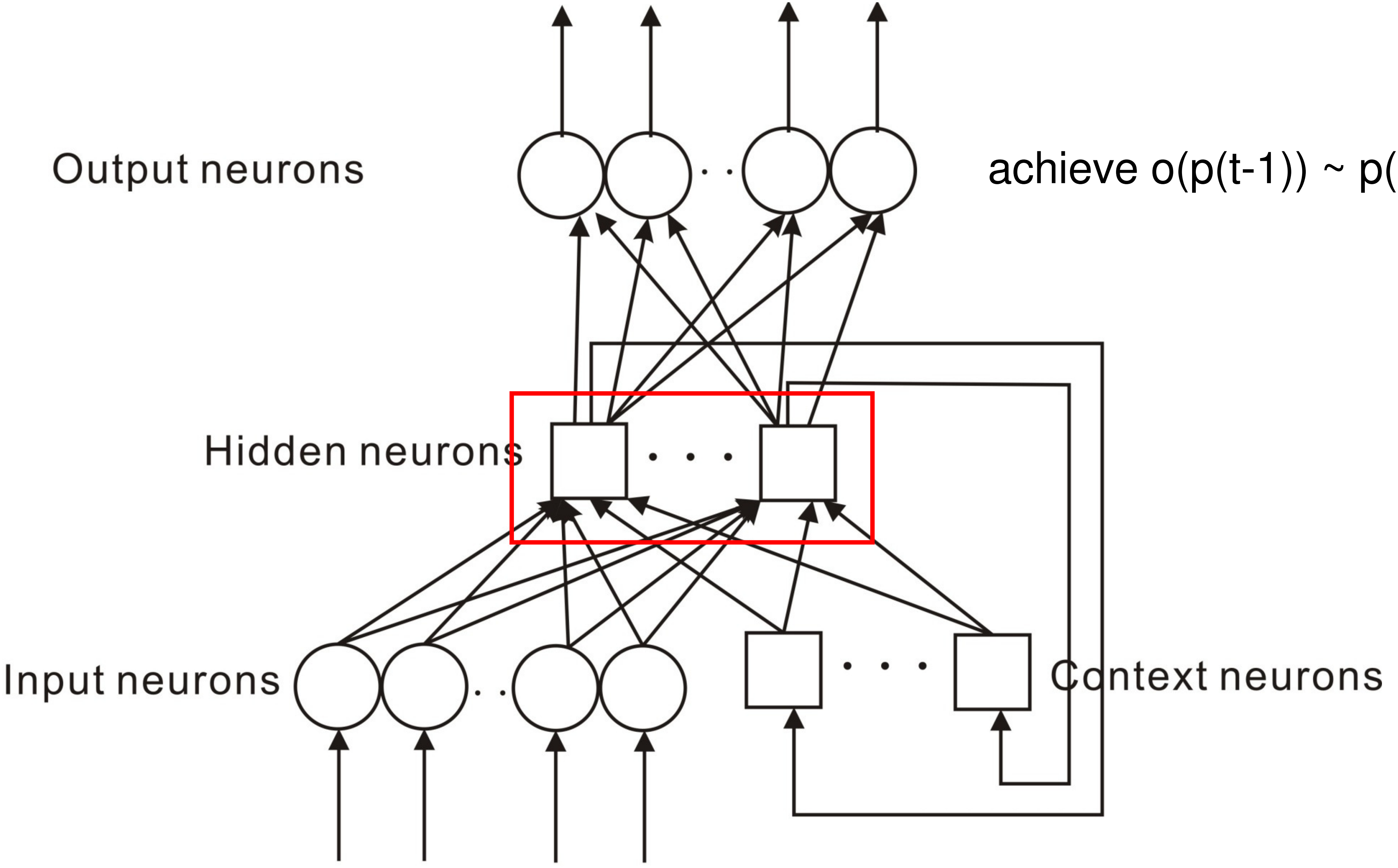


Fragment of Training Sequences for Sentence Simulation

Input	Output
0000000000000000000000000000010 (*woman*)	0000000000000000000000000010000 (*smash*)
0000000000000000000000000010000 (*smash*)	0000000000000000000001000000000 (*plate*)
0000000000000000000001000000000 (*plate*)	0000010000000000000000000000000 (*cat*)
0000010000000000000000000000000 (*cat*)	0000000000000000000100000000000 (*move*)
0000000000000000000100000000000 (*move*)	0000000000000000100000000000000 (*man*)
0000000000000000100000000000000 (*man*)	0001000000000000000000000000000 (*break*)
0001000000000000000000000000000 (*break*)	0000100000000000000000000000000 (*car*)
0000100000000000000000000000000 (*car*)	0100000000000000000000000000000 (*boy*)
0100000000000000000000000000000 (*boy*)	0000000000000000000100000000000 (*move*)
0000000000000000000100000000000 (*move*)	0000000000001000000000000000000 (*girl*)
0000000000001000000000000000000 (*girl*)	0000000000100000000000000000000 (*eat*)
0000000000100000000000000000000 (*eat*)	0010000000000000000000000000000 (*bread*)
0010000000000000000000000000000 (*bread*)	0000000010000000000000000000000 (*dog*)
0000000010000000000000000000000 (*dog*)	0000000000000000000100000000000 (*move*)
0000000000000000000100000000000 (*move*)	0000000000000000001000000000000 (*mouse*)
0000000000000000001000000000000 (*mouse*)	0000000000000000001000000000000 (*mouse*)
0000000000000000001000000000000 (*mouse*)	0000000000000000000100000000000 (*move*)
0000000000000000000100000000000 (*move*)	1000000000000000000000000000000 (*book*)
1000000000000000000000000000000 (*book*)	0000000000000001000000000000000 (*lion*)

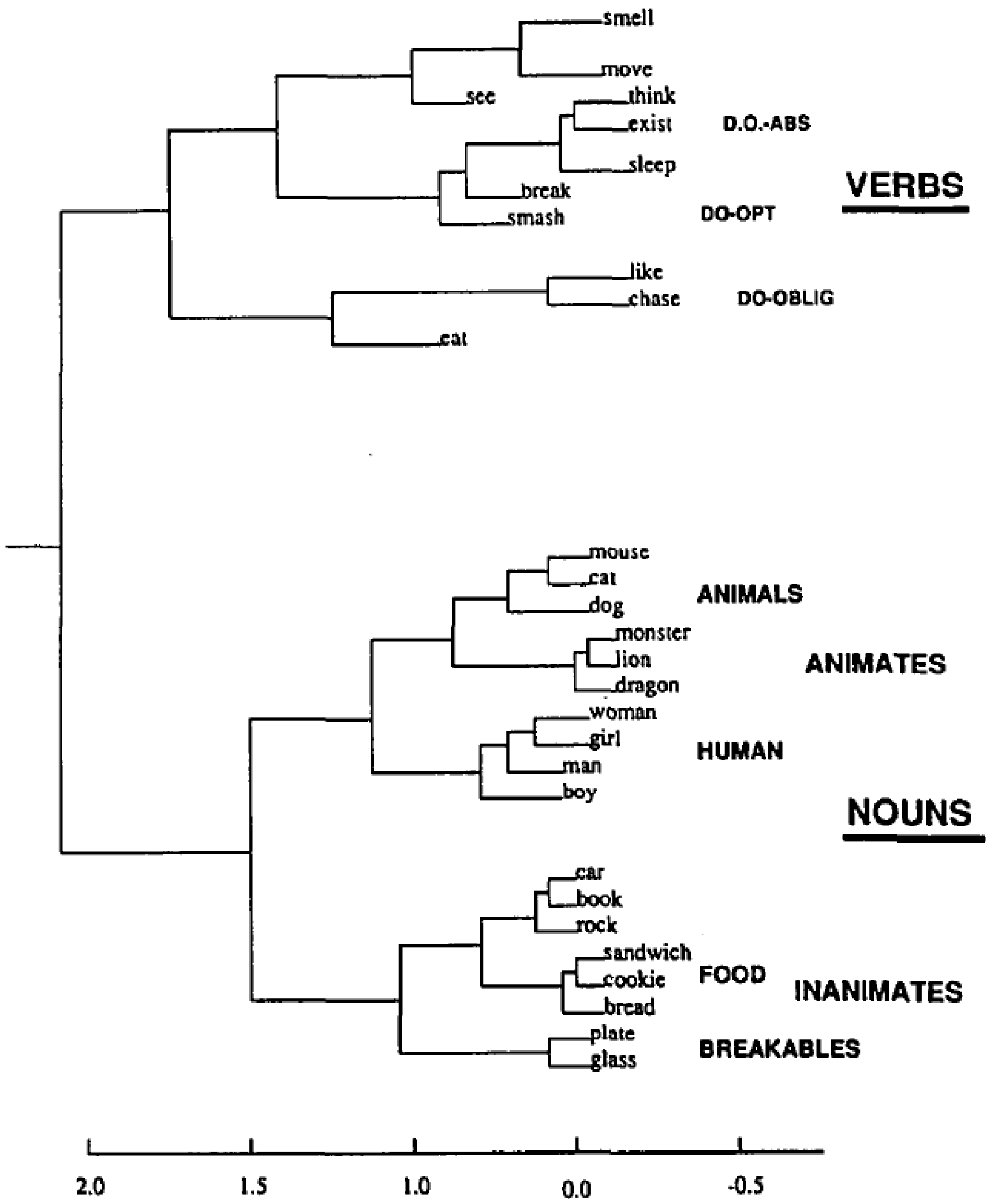
Elman, J.L., Finding structure in time. Cognitive Science 14, 179–211 (1990)

Desired output: d(t-1)=smash, d(t)=plate, d(t+1)=cat, d(t+2)=move …



achieve o(p(t-1)) ~ p(t)

Input: p(t-1)=woman, p(t)=smash, p(t+1)=plate, p(t+2)=cat, p(t+3)=move …

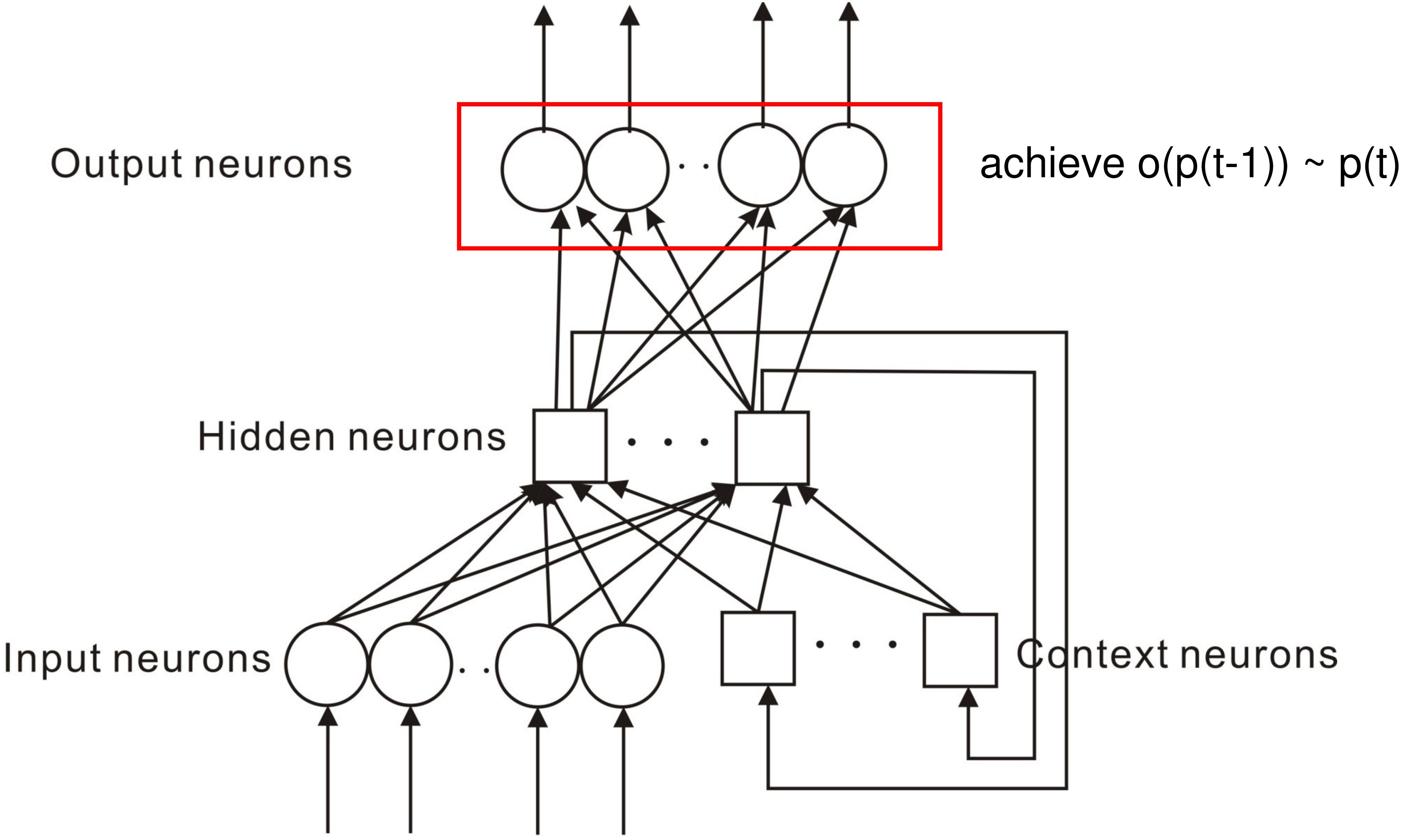


Elman, J.L., Finding structure in time. Cognitive Science 14, 179–211 (1990)

Redesign training method

- Seek better performance
- Process real complex sentences
- Replace Elman's fixed codes by iteratively improved codes.

Desired output: d(t-1)=smash, d(t)=plate, d(t+1)=cat, d(t+2)=move …



achieve o(p(t-1)) ~ p(t)

Input: p(t-1)=woman, p(t)=smash, p(t+1)=plate, p(t+2)=cat, p(t+3)=move …

Iterative re-encoding

- We modify his method. Each word has a random lexical code initially

$$c_n^{j=0} = [c_{n1} \quad c_{n2} \quad \dots \quad c_{nR}]^T$$

- After the jth training epoch, an improved code is calculated by

$$c_n^{raw} = \frac{1}{freq_n} \sum_{p(t)=p_n} o(p(t-1)), n = 1 \dots N.$$

N = total number of words

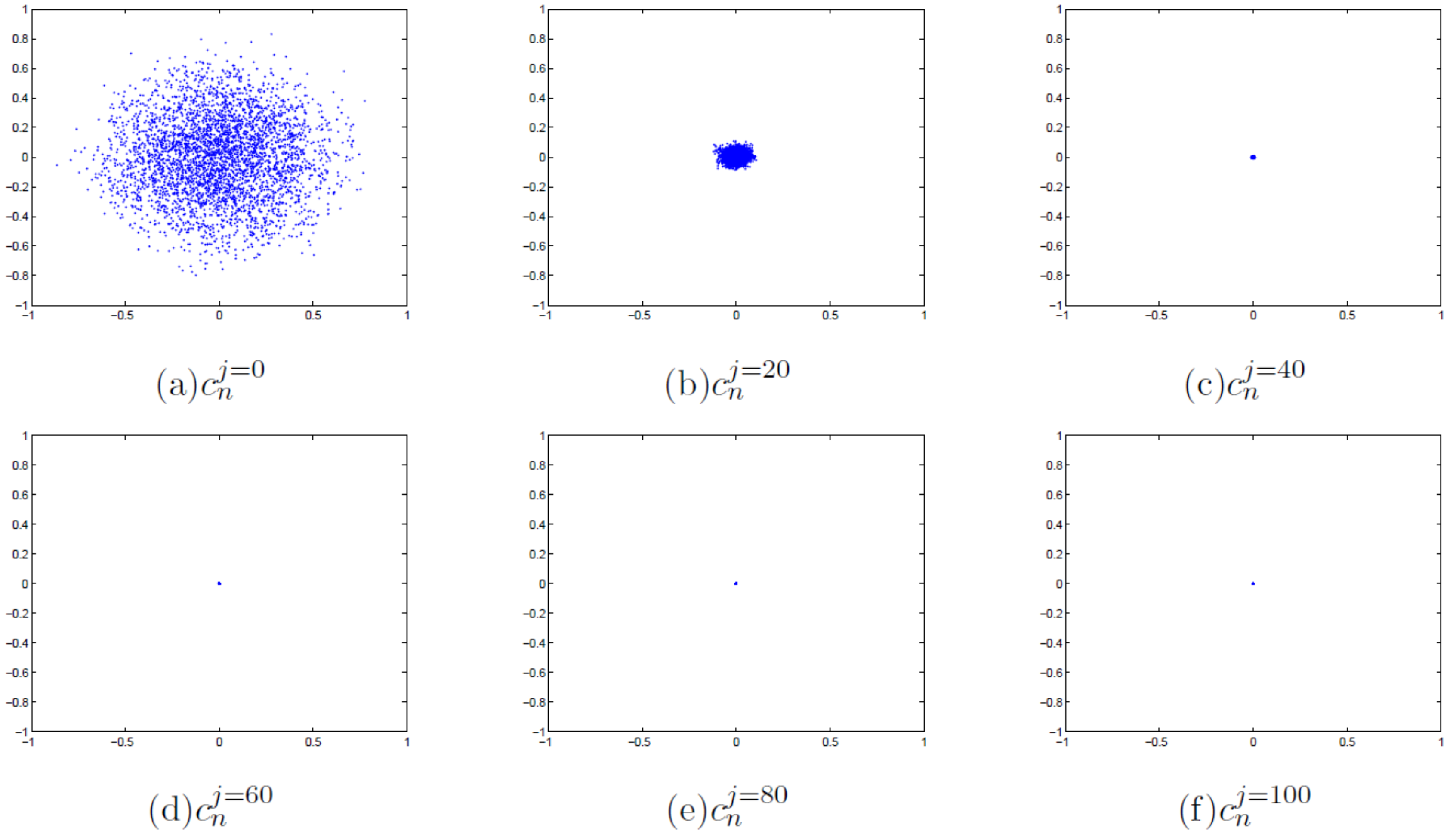
Normalization

- After re-encoding iteration, all the codes are normalized by the following two equations.

$$C^{ave}_{R\times N} = C^{raw}_{R\times N} - \frac{1}{N} C^{raw}_{R\times N} \begin{bmatrix} 1 & \cdots & \cdots & 1 \\ \vdots & 1 & & \vdots \\ \vdots & & \ddots & \vdots \\ 1 & \cdots & \cdots & 1 \end{bmatrix}_{N\times N}$$

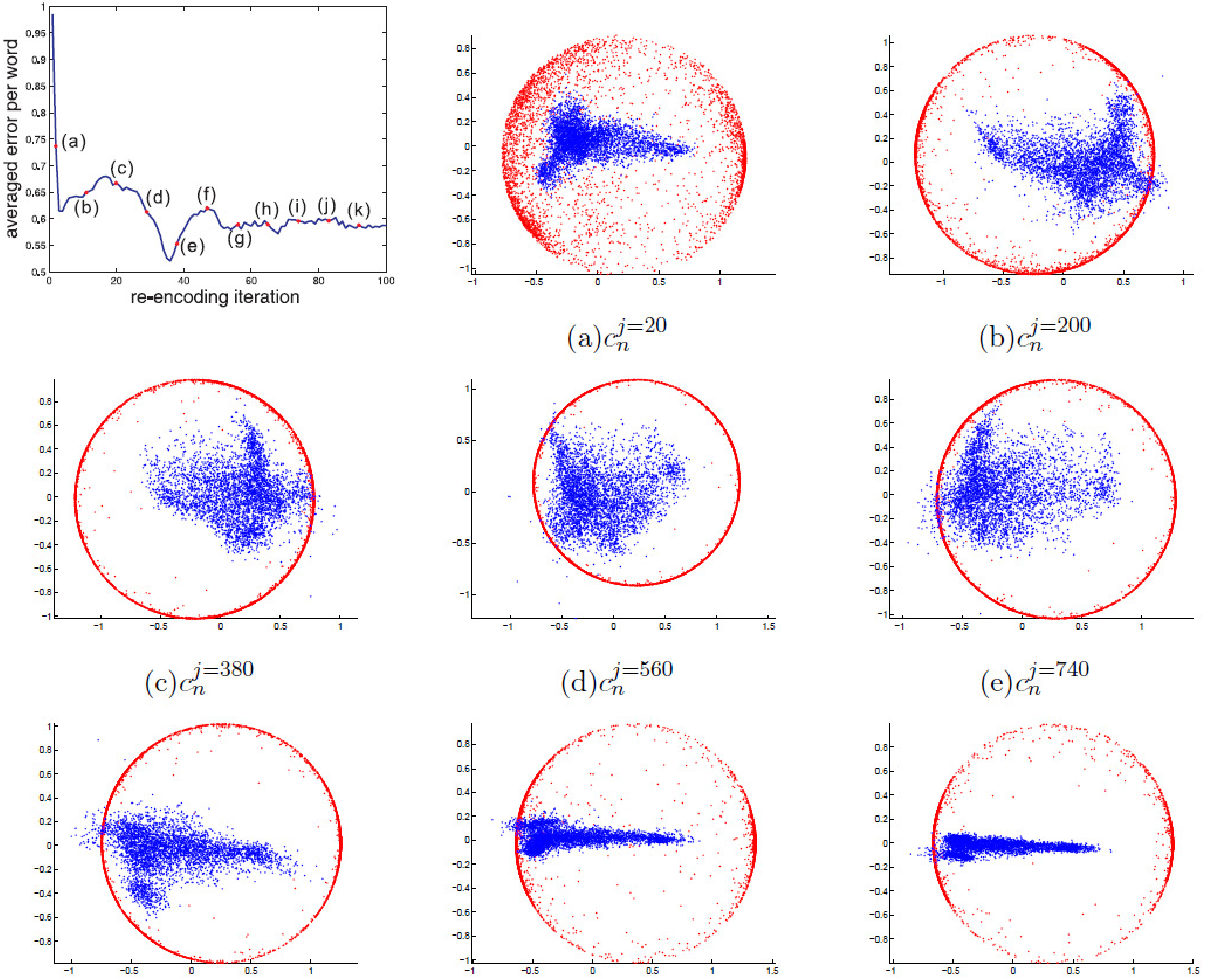
$$c^{j}_{n} = c^{nom}_{n} = \left\| c^{ave}_{n} \right\|^{-1} c^{ave}_{n}, \text{ where } \left\| c_{n} \right\| = (c^{T}_{n} c_{n})^{0.5}, n = 1 \ldots N$$

Without normalization



(a)$c_n^{j=0}$ (b)$c_n^{j=20}$ (c)$c_n^{j=40}$

(d)$c_n^{j=60}$ (e)$c_n^{j=80}$ (f)$c_n^{j=100}$



(a)$c_n^{j=20}$

(b)$c_n^{j=200}$

(c)$c_n^{j=380}$

(d)$c_n^{j=560}$

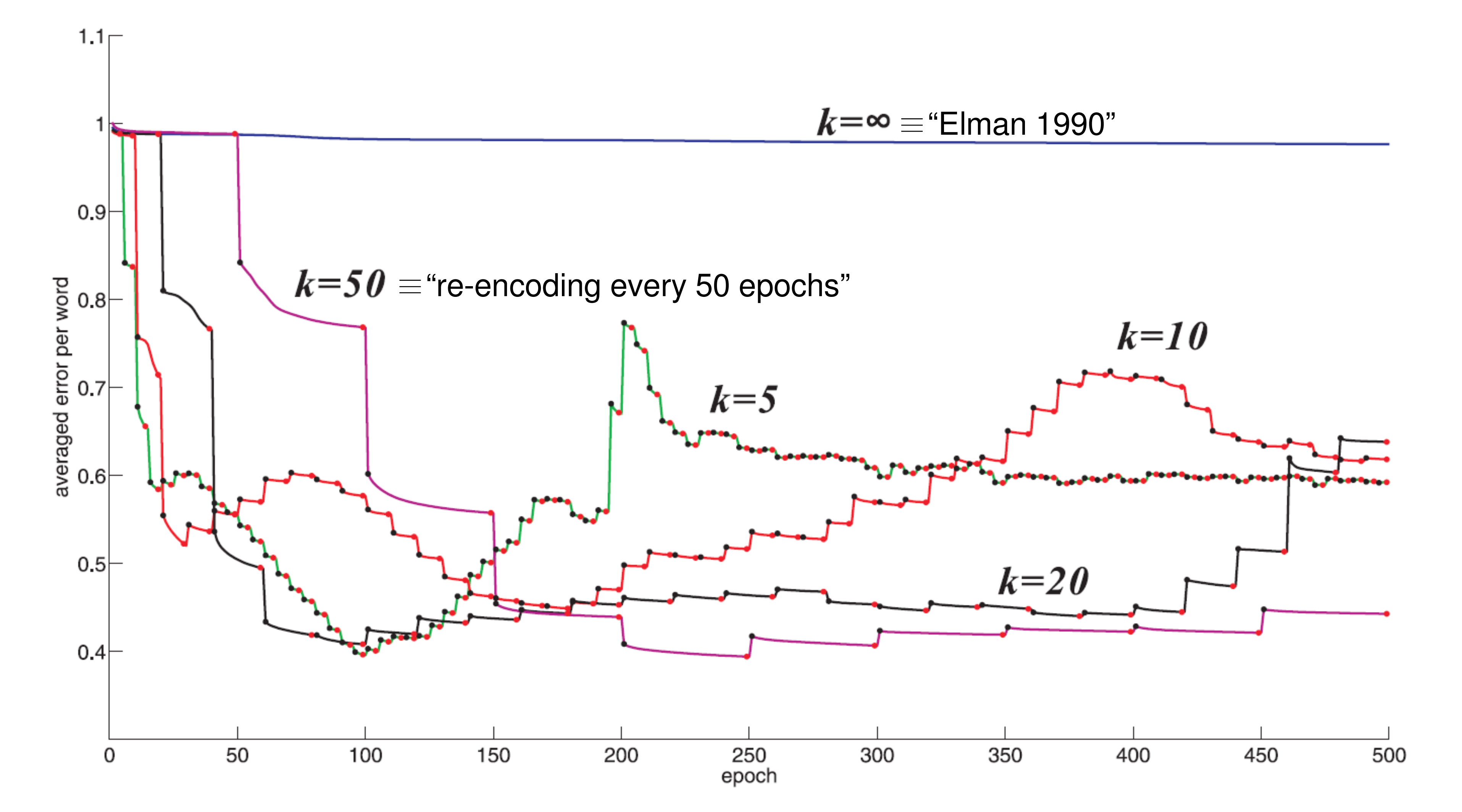
(e)$c_n^{j=740}$

(f)$c_n^{j=920}$

(g)$c_n^{j=1100}$

(h)$c_n^{j=1280}$

Error curves



Example in Peter Pan

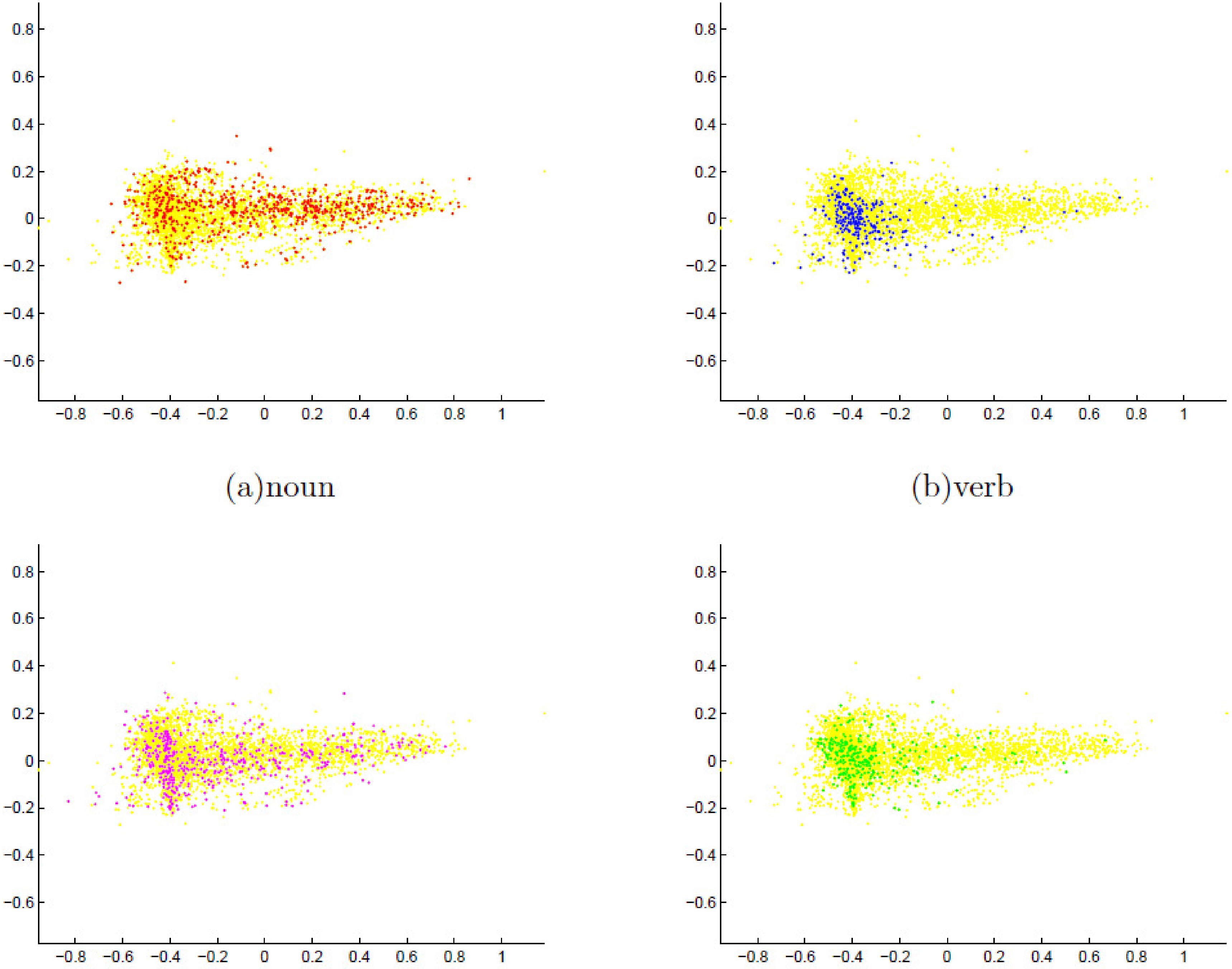
- **Initial:** R=15; N=3805
- ***Boy***: [0.1867, -0.3411, 0.2665, 0.3037, 0.3157, 0.2574, 0.2387, -0.2287, -0.3550, 0.3220, -0.2163, 0.2809, -0.1270, -0.0870, 0.1770]^T
- ***Man***: [-0.0571, 0.2584, 0.0934, -0.2320, 0.4250, -0.2483, -0.3830, 0.0888, 0.0509, -0.1402, 0.3559, -0.2303, 0.3278, -0.2766, 0.2910]^T
- **distance**: 1.6473

- **After training:**
- ***Boy***: [-0.4363, -0.1845, -0.1174, 0.0072, -0.1722, -0.2460, 0.3524, -0.2572, -0.0608, 0.3965, 0.3854, 0.1936, -0.2149, 0.1318, 0.2662]^T
- ***Man***: [-0.4777, -0.1726, -0.0979, 0.0818, -0.1747, -0.2602, 0.3276, -0.2490, -0.1086, 0.3986, 0.3443, 0.1696, -0.2775, 0.0910, 0.2343]^T
- **distance**: 0.1409

Example in Peter Pan

- **Initial:**
- ***Time***: [-0.0693, -0.1763, -0.2876, -0.0366, 0.4460, -0.2576, -0.4200, -0.2537, 0.0567, 0.4300, -0.2080, -0.0808, -0.1117, -0.0879, 0.3404]^T
- ***Long***: [-0.1176, -0.2166, -0.4828, -0.1114, -0.1213, 0.1773, -0.1128, 0.1564, -0.0203, -0.3818, 0.2546, -0.0276, -0.5095, 0.3430, 0.1460]^T
- **distance**: 1.4439

- **After training:**
- ***Time***: [-0.4195, -0.1801, -0.0135, 0.1475, -0.1684, -0.1515, 0.3831, -0.3056, -0.0390, 0.3809, 0.3861, 0.2289, -0.2062, 0.1365, 0.2625]^T
- ***Long***: [0.3273, 0.1145, 0.4445, 0.4733, 0.1177, 0.5038, -0.0499, -0.0432, 0.1556, -0.2693, -0.1592, 0.0405, 0.2087, -0.0263, -0.1349]^T
- **distance**: 1.6965



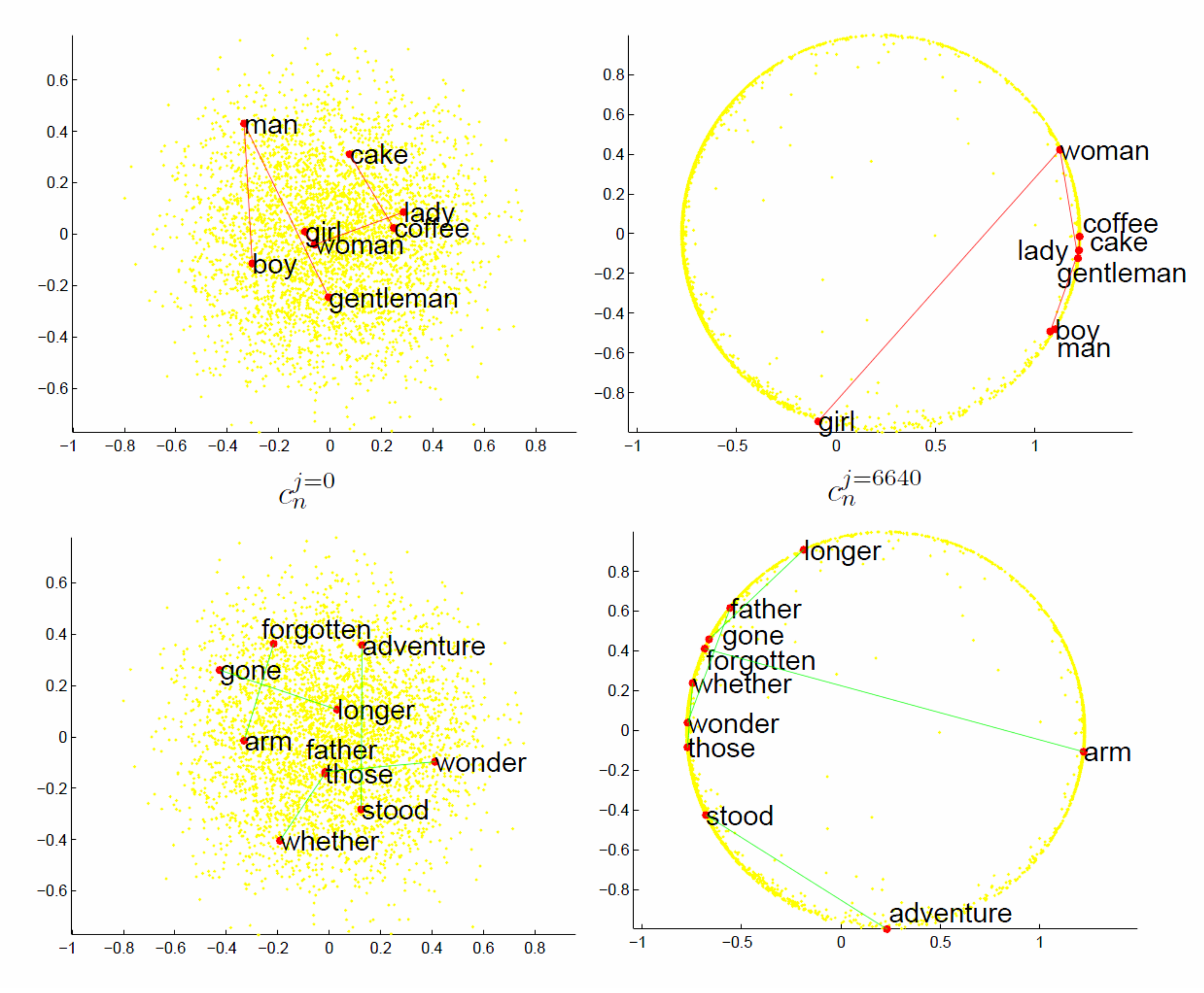
(a)noun

(b)verb

(c)adjective

(d)adverb

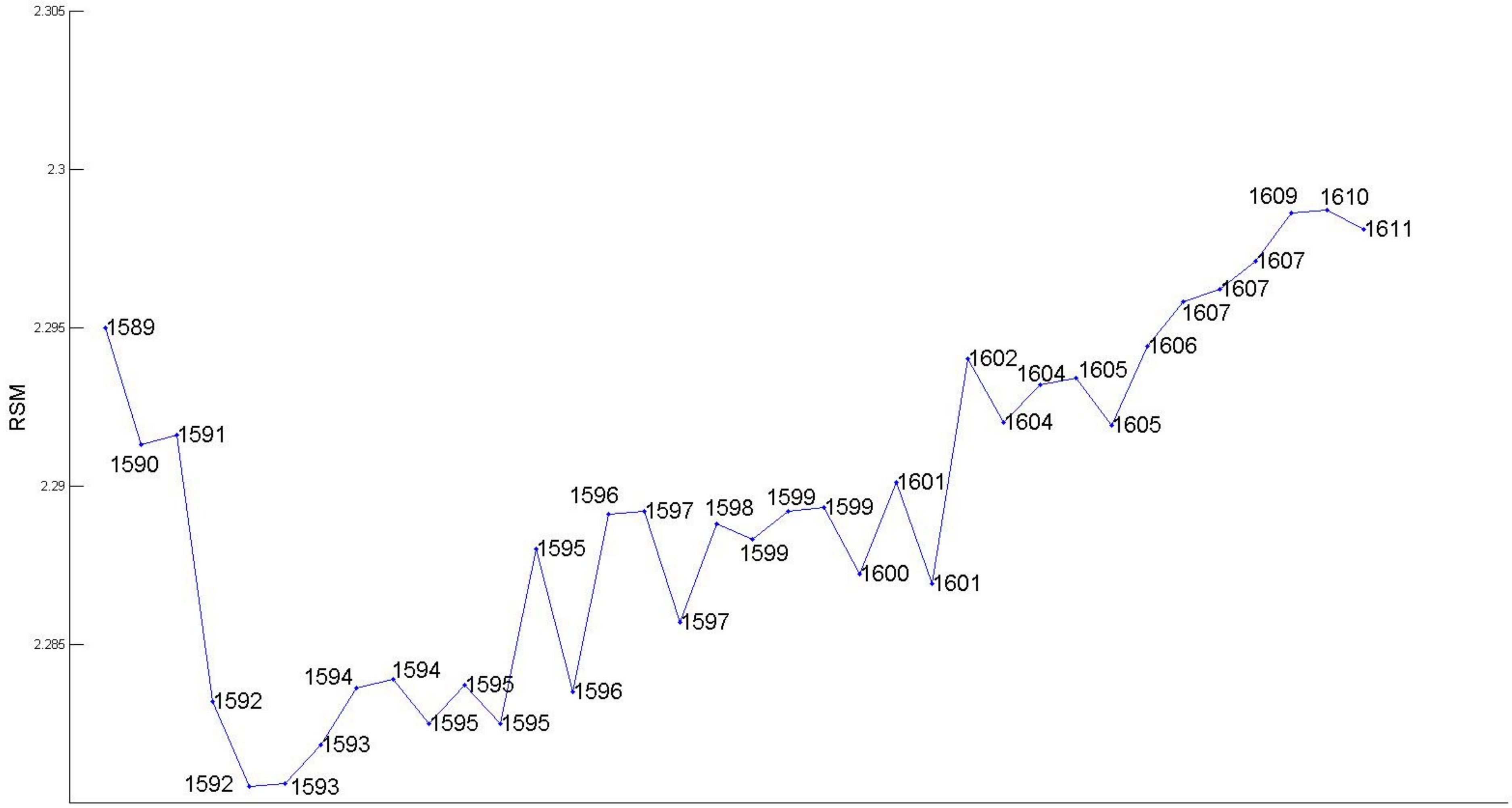
Semantic indexing



Conclusion

- A new method to get distributed representation of word automatically from sequence of words without outside knowledge.
- Better performance during learning
  - Adjusting not only weights but also codes to achieve lower training error.
- Process real complex sentences
  - Semantic indexing, semantic search, text classification, data mining, …ect.

Richness semantic meaning of Shakerspeare's works


RSM