# Continuous Space Language Model(NNLM)

Liu Rong
Intern students of CSLT
2013-12-30

# Outline

- N-gram
  - Introduction
  - data sparsity and smooth
- NNLM
  - Introduction
  - Multi NNLMs
  - Toolkit
- Word2vec(Deep learing in NLP)
  - Introduction
  - some methods on train word2vec
  - Toolkit
- References

# N-gram-Introduction

- N-Gram
  - A language model is usually formulated as a probability distribution *p*(*s*) over strings *s* that attempts to reflect how frequently a string *s* occurs as a sentence

$$\mathrm{p}(\mathrm{s}) = \mathrm{p}(\mathrm{w}_1, \mathrm{w}_2, \cdots, \mathrm{w}_k) = \mathrm{p}(\mathrm{w}_1)\mathrm{p}(\mathrm{w}_2|\mathrm{w}_1)\cdots\mathrm{p}(\mathrm{w}_k|\mathrm{w}_1, \mathrm{w}_2, \cdots, \mathrm{w}_{k-1})$$

  - n-gram

$$p(s) = \prod_{i=1}^{l+1} p(w_i|w_{i-n+1}^{i-1}) \qquad p(w_i|w_{i-n+1}^{i-1}) = \frac{c(w_{i-n+1}^{i})}{\sum_{w_i} c(w_{i-n+1}^{i})}$$

  - example(3-gram)

eg: “This is a example”

p(This is a example) ≈p(This|<s>)p(is|This <s>)p(a| <s> This is)

p(example|This is a)p(</s>| is a example)

**=>** to calculate the p(example|This is a)

N-Gram: p(example|This is a) = $\frac{C(This\ is\ a\ example)}{C(This\ is\ a)}$

# N-gram-data sparsity

- Data sparsity

Example:

the data: John read moby dick(白鲸记). Mary read a different book. She read a book by Cher.(15 words+18 phrase)

p(John read a book)=p(John|<s>)p(read|John)p(a | read)p(book|a)p(</s>|book)

=1/3 * 1/1 * 2/3 * 1/2 * 1/2

=0.06

P(Cher read a book)=p(Cher|<s>)p(read|a)p(a|read)p(book|a)p(</s>|book)

=0/3 0/1 2/3 1/2 ½

=0

\data\

ngram 1= 13

ngram 2= 16

\1-grams:

| | |
|---|---|
| 2/15 | a |
| 2/15 | book |
| 1/15 | by |
| 1/15 | dick |
| 1/15 | differernt |
| 1/15 | John |
| 1/15 | Mary |
| 1/15 | moby |
| 3/15 | read |
| 1/15 | cher |
| 1/15 | she |
| 3/15 | <s> |
| 3/15 | </s> |

\2-grams:

| | |
|---|---|
| 1/1 | John read |
| 1/3 | read moby |
| 1/1 | moby dick |
| 1/1 | Mary read |
| 2/3 | read a |
| 1/2 | a different |
| 1/1 | different book |
| 1/1 | she read |
| 1/2 | a book |
| 1/2 | book by |
| 1/1 | by cher |
| 1/1 | cher </s> |
| 1/3 | <s> John |
| 1/1 | dick </s> |
| 1/3 | <s> Mary |
| 1/2 | book </s> |

# N-gram-smoothing methods

- Data sparsity
- Smoothing methods

\data\
ngram 1= 13
ngram 2= 16

\1-grams:

| | | |
|---|---|---|
| 2/15 | a | 0.1 |
| 2/15 | book | 0.2 |
| 1/15 | by | 0.1 |
| 1/15 | dick | 0.01 |
| 1/15 | differernt | 0.1 |
| 1/15 | John | 0.1 |
| 1/15 | Mary | 0.1 |
| 1/15 | moby | 0.1 |
| 3/15 | read | 0.1 |
| 1/15 | cher | 0.1 |
| 1/15 | she | 0.1 |
| 3/15 | <s> | 0.1 |
| 3/15 | </s> | 0.1 |

\2-grams:

| | |
|---|---|
| 1/1 | John read |
| 1/3 | read moby |
| 1/1 | moby dick |
| 1/1 | Mary read |
| 2/3 | read a |
| 1/2 | a different |
| 1/1 | different book |
| 1/1 | she read |
| 1/2 | a book |
| 1/2 | book by |
| 1/1 | by cher |
| 1/1 | cher </s> |
| 1/3 | <s> John |
| 1/1 | dick </s> |
| 1/3 | <s> Mary |
| 1/2 | book </s> |

P(Cher read a book)=p(Cher|<s>)p(read|a)p(a|read)p(book|a)p(</s>|book)

=p(Cher)*0.1 p(read)*0.1* p(a|read) *p(book|a)*p(</s>|book)

=0.00002

# N-gram-smoothing methods

- Data sparsity
- Smoothing methods
  - Additive smoothing
  - Good-Turing estimate
  - Jelinek-Mercer smoothing (interpolation)
  - Katz smoothing (backoff)
  - Witten-Bell smoothing
  - Absolute discounting
  - Kneser-Ney smoothing

$$p_{\text{add}}(w_i|w_{i-n+1}^{i-1}) = \frac{\delta + c(w_{i-n+1}^{i})}{\delta|V| + \sum_{w_i} c(w_{i-n+1}^{i})}$$

$$P_{\text{katz}}(w_n|w_{n-N+1}^{n-1}) = \begin{cases} P^*(w_n|w_{n-N+1}^{n-1}), & \text{if } C(w_{n-N+1}^{n}) > 0 \\ \alpha(w_{n-N+1}^{n-1}) P_{\text{katz}}(w_n|w_{n-N+2}^{n-1}), & \text{otherwise.} \end{cases}$$

# NNLM-Introduction

A Neural Probabilistic Language Model(Bengio et al, NIPS'2000 and JMLR 2003)

- Motivation:
  - LM does not take into account contexts farther than 2 words.
  - LM does not take into account the "similarity" between words.
- Idea:
  - A word $w$ is associated with a distributed feature vector (a real-valued vector in $\mathbb{R}^n$ $n$ is much smaller than size of the vocabulary)
  - Express joint probability function $f$ of words sequence in terms of feature vectors
  - Learn simultaneously the word feature vector and the parameters of $f$

# NNLM-Intoduction

- Target:

  Neural architecture

  $P(w_j | w_{j-n+1,\ldots,} w_{j-2,} w_{j-1})=$

  $$f(i, w_{t-1}, \ldots, w_{t-n+1}) = g(i, C(w_{t-1}), \ldots, C(w_{t-n+1}))$$

- Projection:

  word2vector: word→[0.1,0.2,…..,0.3]→$C_k$

  =>feature vector for each word

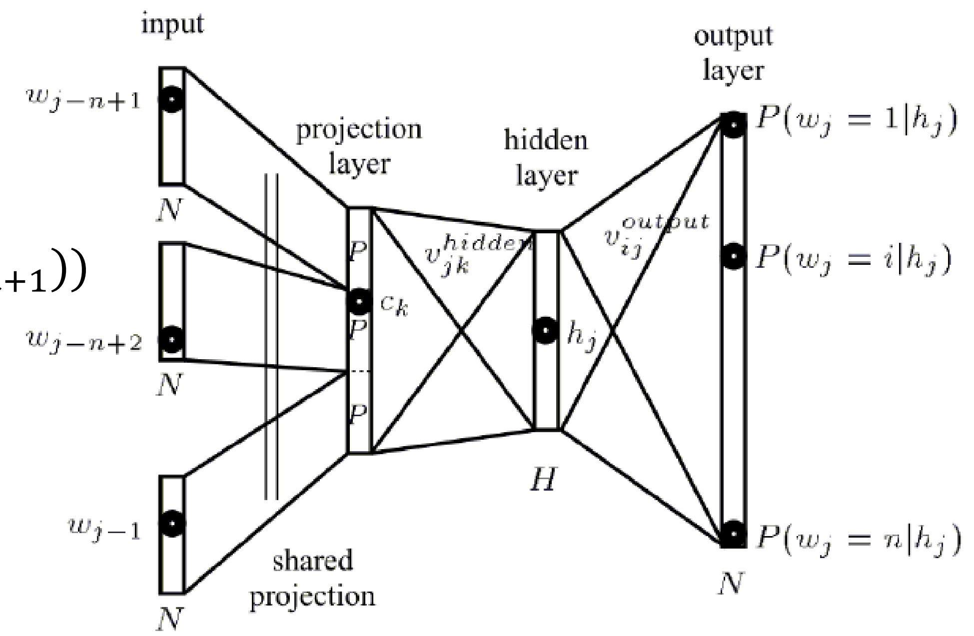



Figure: *Feedforward neural network based LM used by Y. Bengio and H. Schwenk*


[1] Y. Bengio, and R. Ducharme. A neural probabilistic language model. In *Neural Information Processing Systems,* volume 13, pages 932-938. 2001.

# NNLM-Introduction

- softmax output layer:

$$P(w_t|w_{t-1},\cdots,w_{t-n+1}) = \frac{e^{y_{w_t}}}{\sum_i e^{y_i}}$$

- $y_i$ unnormalized log-probabilities for each output word $i$

$$y = b + U\tanh(d + Hx)$$

- $x$ is the word features layer activation vector

$$x = \big(C(w_{t-1}), \ldots, C(w_{t-n+1})\big)$$

- The free parameters of the model are:

$$\theta = (b, d, W, U, H, C)$$

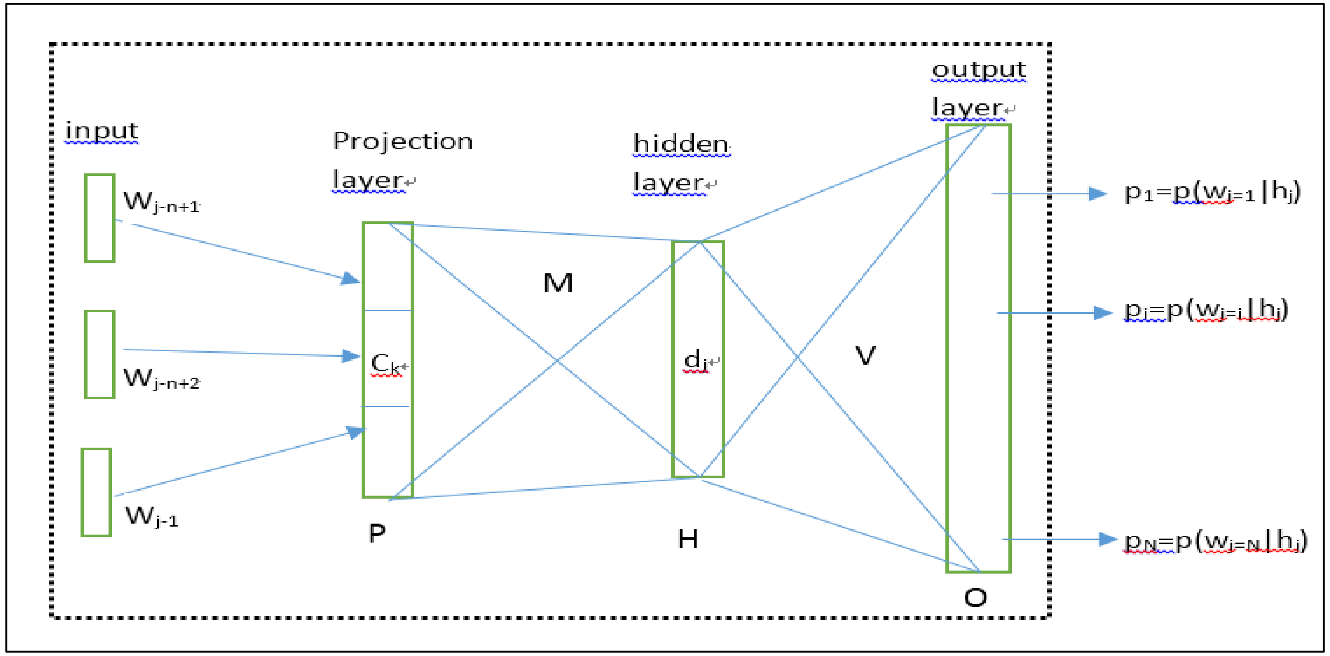
  - $b$ output biases ($|V|$)
  - $d$ the hidden layer biases ($h$)
  - $U$ the hidden-to-output weights ($|V| \times h$)
  - $H$ the hidden layer weights ($h \times (n-1)m$)
  - $C$ word features ($|V| \times m$)

# NNLM-Forward Phase

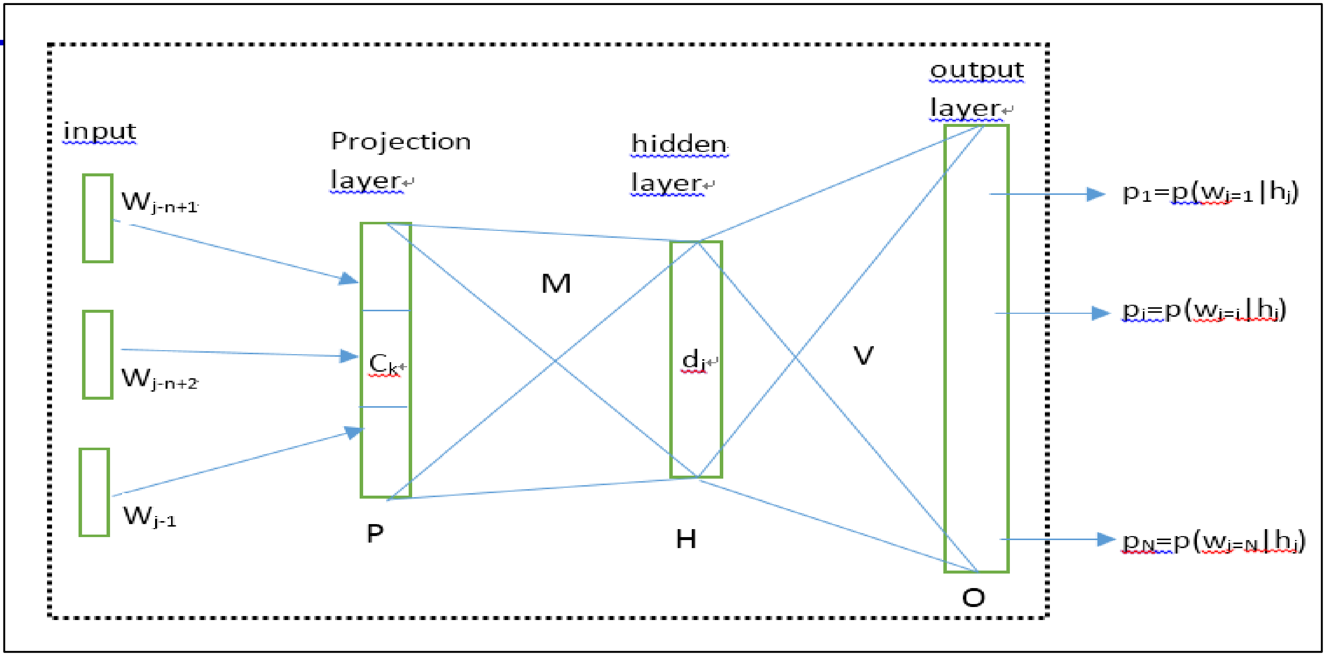
- Steps forward:

$$h_j = \sum_l m_{jl}\, c_l + b_j$$
$$\mathrm{d}_j = \tanh(h_j)$$

$$\mathrm{O}_i = \sum_j v_{ij} d_j + k_i$$
$$\mathrm{p}_i = e^{O_i} / \sum_{r=1}^{N} e^{O_r}$$



# NNLM-Backward/Update Phase




- Steps BP

$$E = \sum_{i=1}^{N} t_i ln p_i + \beta(\sum_{jl} m_{jl}^2 + \sum_{ij} v_{ij}^2)$$

where $t_i$=1 if the next word is i or $t_i$=0

note:

the first part: cross-entropy

the second part: is regularization term to prevent the neural network from overfitting the training data

$$\frac{\partial E}{\partial o_i} = \frac{\partial E}{\partial p_i}\frac{\partial p_i}{\partial o_i} = (1 - p_i)$$

$$\frac{\partial E}{\partial k_i} = \frac{\partial E}{\partial o_i}\frac{\partial o_i}{\partial k_i} = \frac{\partial E}{\partial o_i} \Rightarrow k_i = k_i + \varepsilon\frac{\partial E}{\partial k_i}$$

$$\frac{\partial E}{\partial v_{ij}} = \frac{\partial E}{\partial o_i}\frac{\partial o_i}{\partial v_{ij}} = \frac{\partial E}{\partial o_i} d_j \Rightarrow v_{ij} = v_{ij} + \varepsilon\frac{\partial E}{\partial v_{ij}}$$

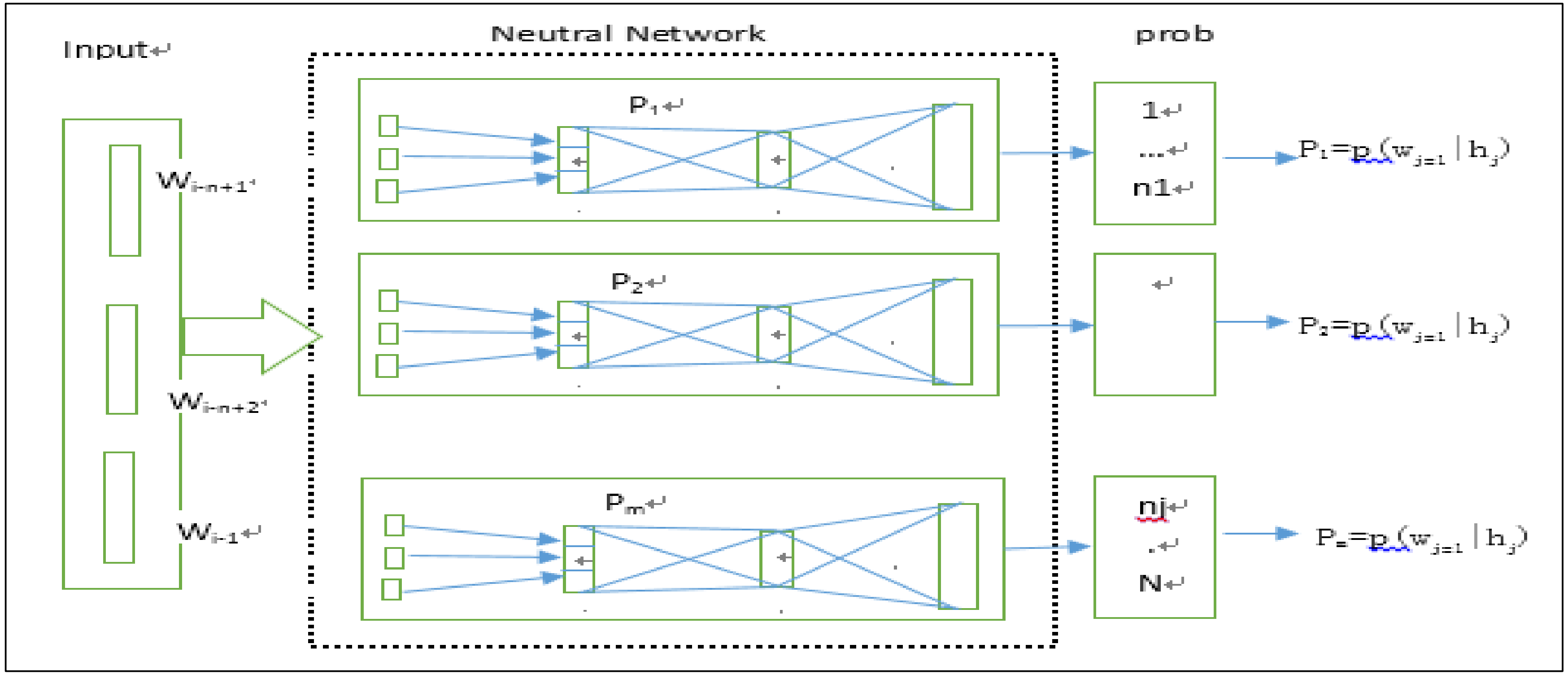
$$\frac{\partial E}{\partial d_j} = \frac{\partial E}{\partial o}\frac{\partial o}{\partial d_j} = \sum_i \frac{\partial E}{\partial o_i} v_{ij}$$

$$\frac{\partial E}{\partial h_j} = \frac{\partial E}{\partial d_j}\frac{\partial d_j}{\partial h_j} = \frac{\partial E}{\partial d_j}(1 - \tanh(h_j)^2)$$

$$\frac{\partial E}{\partial b_j} = \frac{\partial E}{\partial h_j}\frac{\partial h_j}{\partial b_j} = \frac{\partial E}{\partial h_j} \Rightarrow b_j = b_j + \varepsilon\frac{\partial E}{\partial b_j}$$

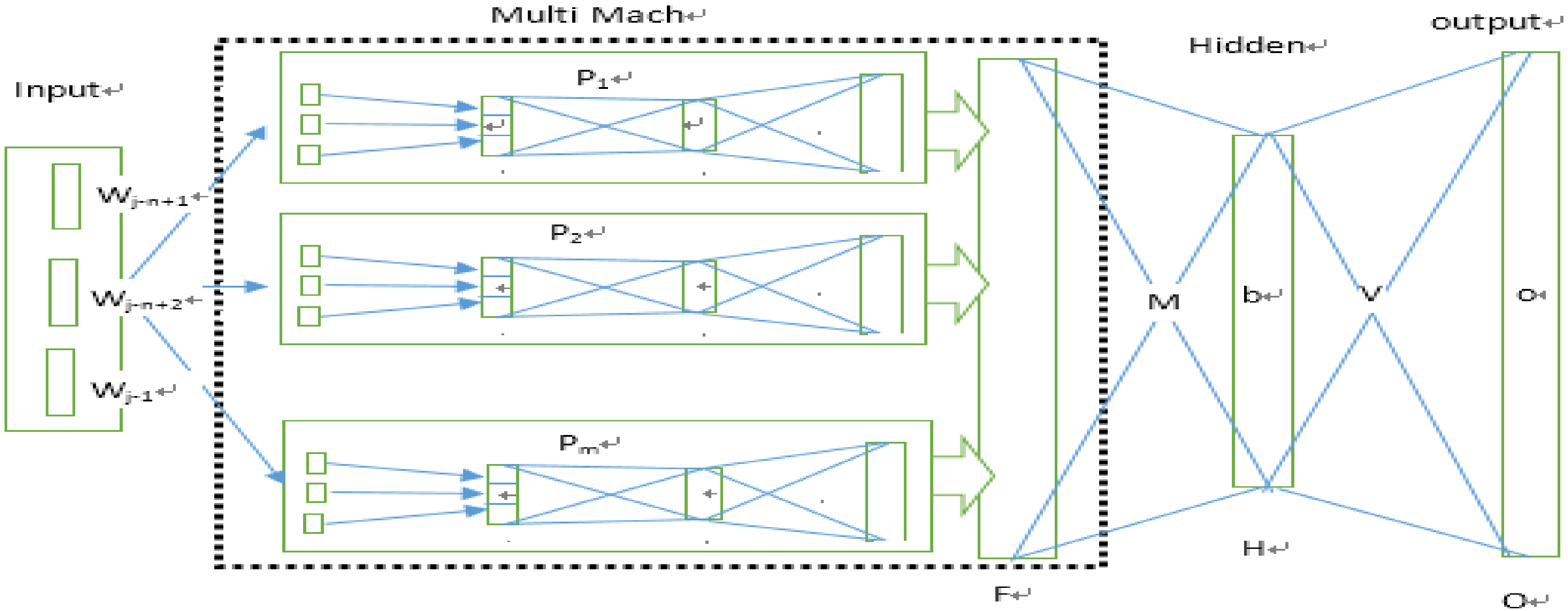
$$\frac{\partial E}{\partial m_{jl}} = \frac{\partial E}{\partial h_j}\frac{\partial h_j}{\partial m_{jl}} = \frac{\partial E}{\partial h_j} c_l \Rightarrow m_{jl} = m_{jl} + \varepsilon\frac{\partial E}{\partial m_{jl}}$$

# NNLM--Multi NNLMs




- $$\left(P\left(\mathrm{w}_j = \mathrm{i}\middle|\mathrm{h}_j\right) = \begin{cases} p_1\left(\mathrm{w}_j = \mathrm{i}\middle|\mathrm{h}_j\right) & 0 < i < N_1 \\ p_2\left(\mathrm{w}_j = \mathrm{i}\middle|\mathrm{h}_j\right) & N_1 < i < N_2 \\ \vdots & \\ p_m\left(\mathrm{w}_j = \mathrm{i}\middle|\mathrm{h}_j\right) & N_{m-1} < i < N_m \end{cases}\right.$$

# NNLM--Merge NNLMs




The steps:

$$F = (P_1(o)^T, P_2(o)^T, \cdots, P_m(o)^T)^T$$

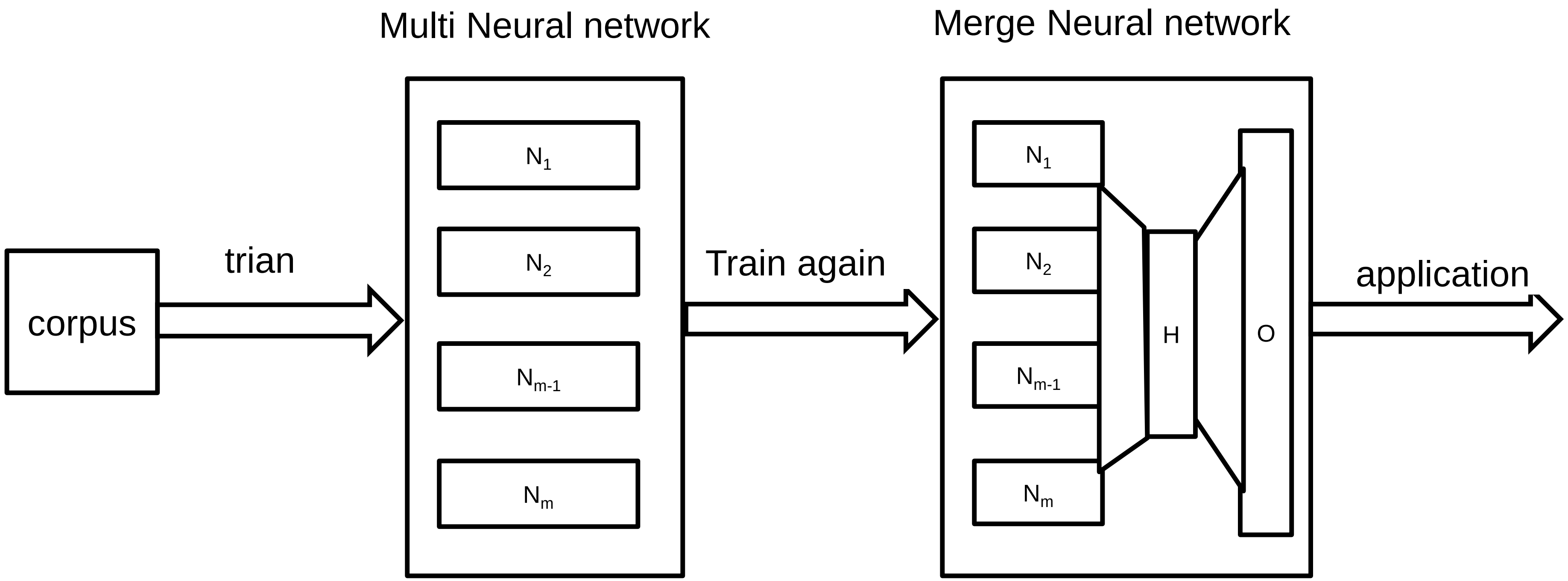
$$H = F \times M + b$$

$$O = H \times V + o$$

$$p_i = e^{O_i} / \sum_{r=1}^{N} e^{O_r}$$

where: $P_m(o)^T$ is the output of single NNLM

# NNLM--Merge NNLMs



# NNLM-results

| model | map | 2044 | notep3 | record1900 | general | online1 | online2 | speedup | average |
|---|---|---|---|---|---|---|---|---|---|
| ngram | 34.4 | 27.18 | 18.78 | 12.89 | 43.23 | 39.49 | 32.15 | 31.47 | 29.94875 |
| Ngram(replace) | 34.93 | 27.57 | 18.78 | 12.92 | 43.45 | 39.48 | 32.15 | 31.51 | 30.09875 |
| NNLM(0-10240) | | | | | | | | | |
| P256-h192 | 34.11 | 27.6 | 18.94 | 13.85 | 44.6 | 39.58 | 32.13 | 31.7 | |
| P256-h384 | 33.63 | 27.57 | 20.02 | 13.53 | 44.3 | 39.31 | 32.03 | 31.08 | |
| P256-h576 | 33.81 | 27.18 | 19.27 | 13.59 | 44.32 | 39.3 | 32.03 | 31.17 | |
| P384-h384 | 33.55 | 27.09 | 19.86 | 13.46 | 44.28 | 39.45 | 32.06 | 31.32 | |
| P256-h384-h256 | 34.21 | 27.52 | 18.73 | 13.69 | 44.72 | 39.88 | 32.34 | 31.04 | |

# NNLM-results

| model | map | 2044 | notep3 | record1900 | general | online1 | online2 | speedup | |
|---|---|---|---|---|---|---|---|---|---|
| ngram | 34.4 | 27.18 | 18.78 | 12.89 | 43.23 | 39.49 | 32.15 | 31.47 | |
| Ngram(replace) | 34.93 | 27.57 | 18.78 | 12.92 | 43.45 | 39.48 | 32.15 | 31.51 | |
| NNLM(0-10240) | | | | | | | | | |
| P256-h192 | 34.01 | 27.11 | 18.19 | 12.75 | 43.4 | 39.17 | 31.91 | 30.9 | |
| P256-h384 | 33.9 | 27.05 | 18.35 | 12.71 | 43.3 | 39.2 | 31.86 | 31.04 | |
| P256-h576 | 33.86 | 27 | 18.62 | 12.79 | 43.29 | 39.17 | 31.96 | 30.66 | |
| P384-h384 | 34.14 | 27.08 | 18.35 | 12.68 | 43.29 | 39.2 | 31.85 | 30.83 | |
| P256-h384-h256 | 34.14 | 27.08 | 18.29 | 12.83 | 43.47 | 39.28 | 31.86 | 31.17 | |

Note: weight0.1:new*0.1+0.9old

# NNLM(multi)--Results

| model | map | 2044 | notep3 | record1900 | general | online1 | online2 | speedup |
|---|---|---|---|---|---|---|---|---|
| ngram | 34.4 | 27.18 | 18.78 | 12.89 | 43.23 | 39.49 | 32.15 | 31.47 |
| ngram(replace) | 34.93 | 27.57 | 18.78 | 12.92 | 43.45 | 39.48 | 32.15 | 31.51 |
| mach1 | 34.35 | 27.8 | 20.45 | 13.27 | 44.22 | 39.78 | 32.28 | 32.03 |
| mach2 | 34.41 | 27.94 | 19.91 | 13.4 | 44.49 | 39.76 | 32.32 | 31.91 |
| mach3 | 34.32 | 27.99 | 19.86 | 13.43 | 44.68 | 39.76 | 32.39 | 31.78 |
| mach4 | 34.19 | 28.11 | 19.05 | 13.4 | 44.63 | 39.75 | 32.4 | 31.65 |
| mach5 | 34.16 | 28.04 | 18.94 | 13.43 | 44.72 | 39.73 | 32.41 | 31.63 |
| mach6 | 34.1 | 28.09 | 18.94 | 13.45 | 44.79 | 39.76 | 32.41 | 31.57 |
| mach7 | 34.11 | 28.06 | 18.83 | 13.45 | 44.91 | 39.74 | 32.43 | 31.67 |
| mach8 | 34.11 | 28.06 | 18.83 | 13.45 | 44.81 | 39.79 | 32.46 | 31.67 |
| mach9 | 34.14 | 28.02 | 18.94 | 13.42 | 44.87 | 39.8 | 32.47 | 31.87 |
| mach10 | 34.24 | 28.04 | 19.05 | 13.46 | 44.88 | 39.81 | 32.49 | 31.89 |
| weight0.1 | 34.16 | 27.14 | 18.46 | 12.77 | 43.46 | 39.22 | 31.91 | 30.94 |

Note: weight0.1:new*0.1+0.9old

# NNLM(merge)-results

| model | map | 2044 | notep3 | record1900 | general | online1 | online2 | speedup |
|---|---|---|---|---|---|---|---|---|
| ngram | 34.4 | 27.18 | 18.78 | 12.89 | 43.23 | 39.49 | 32.15 | 31.47 |
| Ngram(replace) | 34.93 | 27.57 | 18.78 | 12.92 | 43.45 | 39.48 | 32.15 | 31.51 |
| Mach10(no merge) | 34.24 | 28.04 | 19.05 | 13.46 | 44.88 | 39.81 | 32.49 | 31.89 |
| Mach10(merge) | | | | | | | | |
| h100 | 35.42 | 28.4 | 19.64 | 14.11 | 45.53 | 40.25 | 32.89 | 32.9 |
| h200 | 35.92 | 28.46 | 19.75 | 14.25 | 45.51 | 40.42 | 32.85 | 32.71 |

# NNLM--toolkit

- CSLM Toolkit http://www-lium.univ-lemans.fr/cslm/

  Holger Schwenk; *CSLM - A modular Open-Source Continuous Space Language Modeling Toolkit*, in Interspeech, August 2013.

# Word Representation

- Introduction
- C&W
- M&H
- RNNLM
- Huang

# Word2vec--Introduction

- One-hot Representation

  dog => [0 0 0 0 1 0 0 0 0 0 0]

  cat => [1 0 0 0 0 0 0 0 0 0 0]

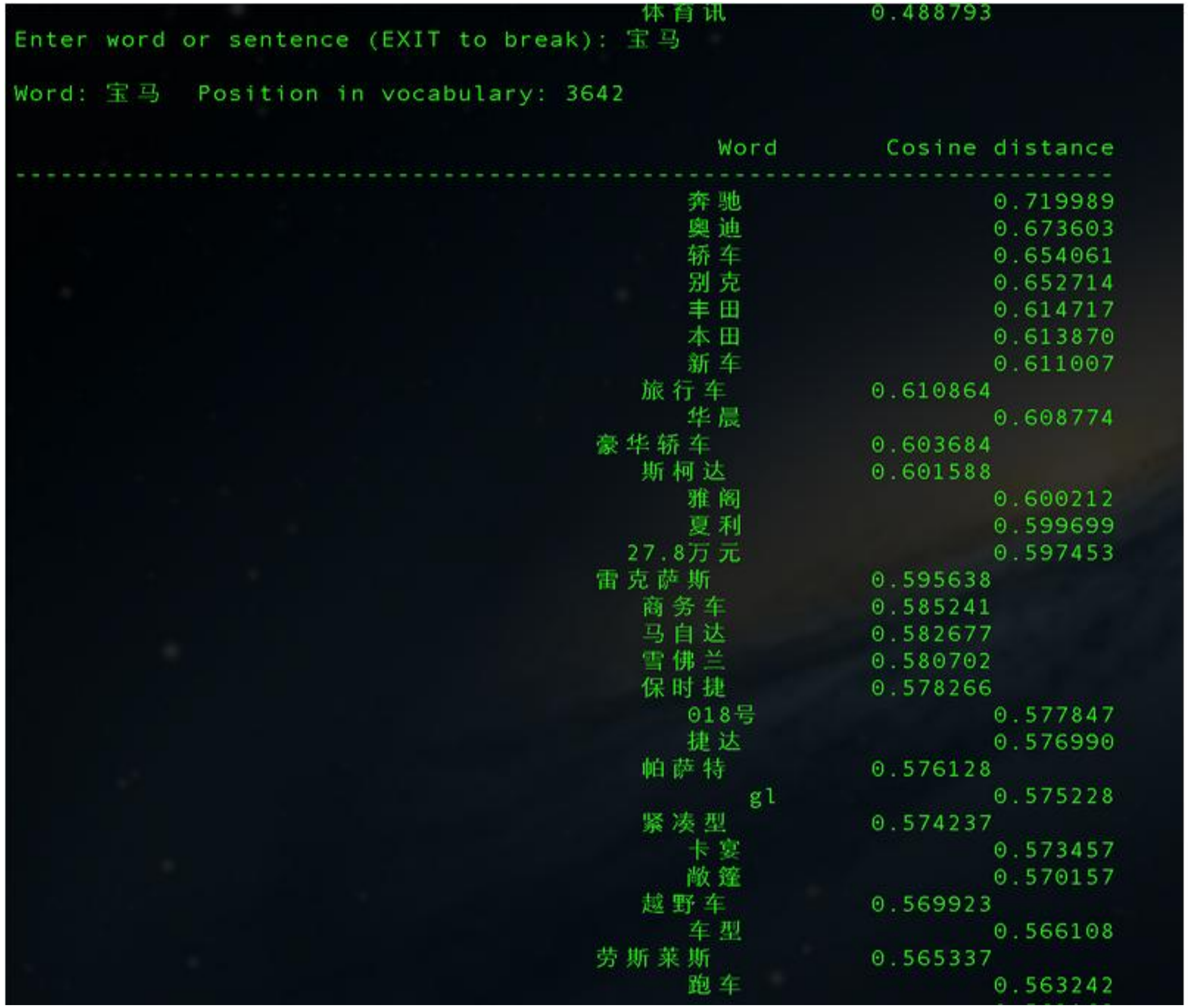
- Distributed Representation

  dog => [0.792 -0.177 0.98 -0.9 …..]

  cat => [0.76 0.12 -0.54 0.9 0.65 ….]

# Word2vec--Introduction

- Language Modeling
  - Speech Recognition
  - Machine Translation
- Part-Of-Speech Tagging
- Chunking
- Named Entity Recognition
- Semantic Role Labeling
- Sentiment Analysis
- Paraphrasing
- Question-Answering
- Word-Sense Disambiguation



# Word2vec—c&w

- A neural network for learning word vectors (Collobert et al. JMLR 2011)

  Natural Language Processing (almost) from Scratch Journal of Machine Learning Research 1 (2000) 1-48

  It focus on how to use word vectors on Natural Language Processing

- Main idea
- A word and its context is a positive training sample; a random word in that same context gives a negative training sample:
  - [+] positive = Score(Cat chills [on] a mat) ---$f(x)$
  - [-] negative = Score(Cat chills [god] a mat)----- $f(x^w)$
- What to feed in the NN
  - each word is an n-dimensional vector, a look up table:

    $$L \in \mathbb{R}^{n\times|V|}$$

  - Training objective:

Positive data set

The negative data set

$$\theta \rightarrow \sum\sum Max\{0, 1 - S_{pos} + S_{neg}\} = \sum_{x\epsilon X}\sum_{w\epsilon D} \max\{0,1 - f(x) + f(x^w)\}$$

Where X is data set(n-windows),D is the dictionary,w is middle word of n-windows

- 3-layer NN:

$$s = U^T f_\theta(Wx + b) \Rightarrow f(w_t, w_{t-1}, \dots, w_{t-n+1})$$

Where $f_\theta(\cdot)$ is a NN function. S is a score for the n-window sentence, x is vector of $(w_t, w_{t-1}, \dots, w_{t-n+1})$

Window size n = 11
|V| = 1300000
7 weeks

- **SENNA:** http://ml.nec-labs.com/senna/

# Word2vec—M&H

Three new graphical models for statistical language modelling   Mnih A, Hinton G.

- Log-Bilinear model

$$h = \sum_{i=1}^{t-1} H_i C(w_i)$$

$$y_j = C(w_j)^T h$$ →Inner product to represent cos distance

Where $C(w_i)$ is a word-vector of $w_i$, $H_i$ is m*m matrix.

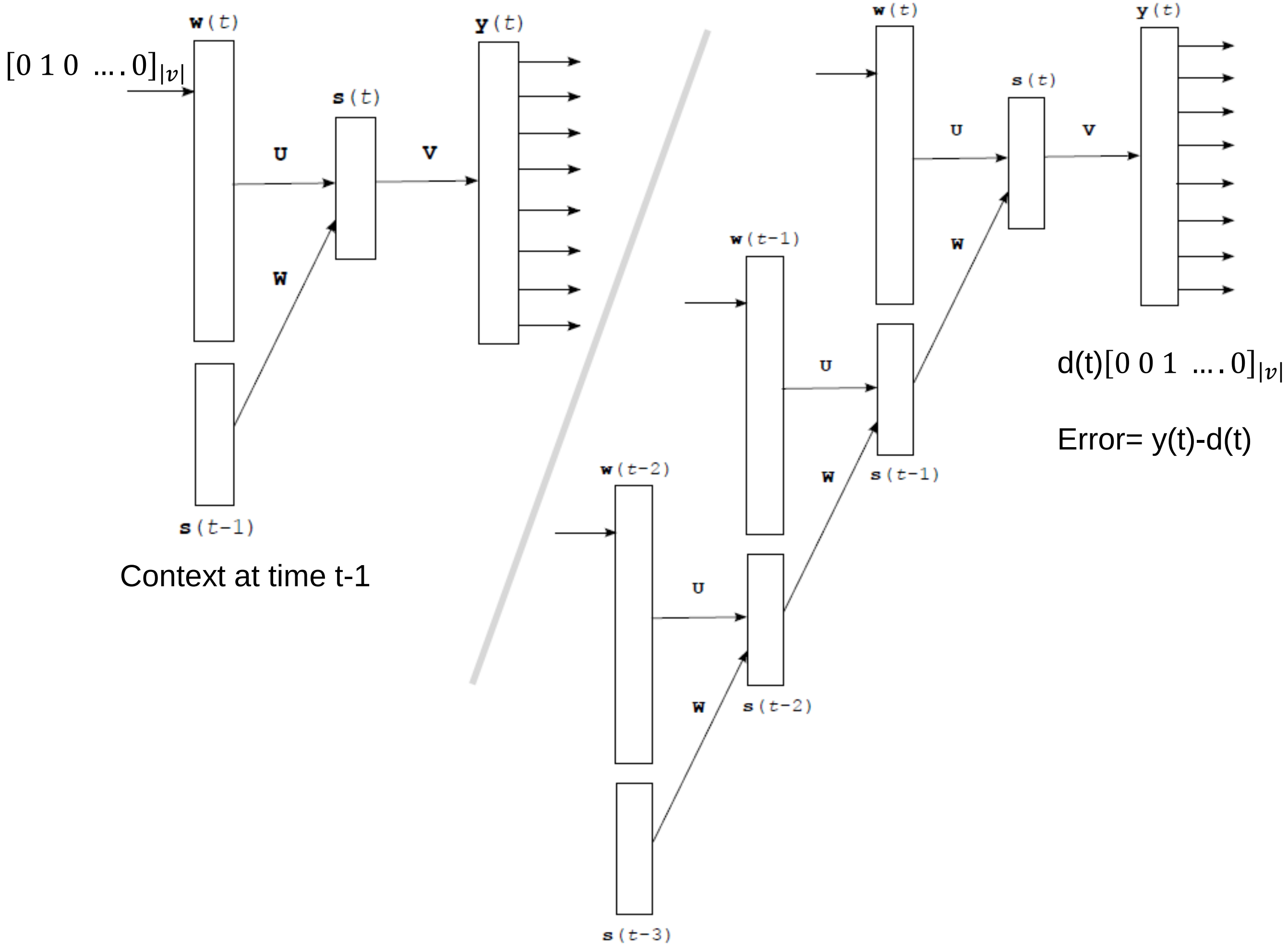
- Hierarchical Log-Bilinear Model

To speed up the calculation

# Word2vec—RNNLM

Linguistic Regularities in Continuous Space Word Representations (Mikolov, et al. 2013)

- Recurrent Neural Network Model



# Word2vec--RNNLM

## Linguistic Regularities in Continuous Space Word Representations (Mikolov, et al. 2013)

- Recurrent Neural Network Model
  - The input vector $w(t)$ represents input word at time $t$ encoded using One hit coding.
  - The output layer $y(t)$ produces a probability distribution over words.
  - The hidden layer $s(t)$ maintains a representation of the sentence history.
  - $w(t)$ and $y(t)$ are of same dimension as vocabulary
- Model:

$$s(t) = f\big(Uw(t) + Ws(t-1)\big)$$
$$y(t) = g(Vs(t))$$

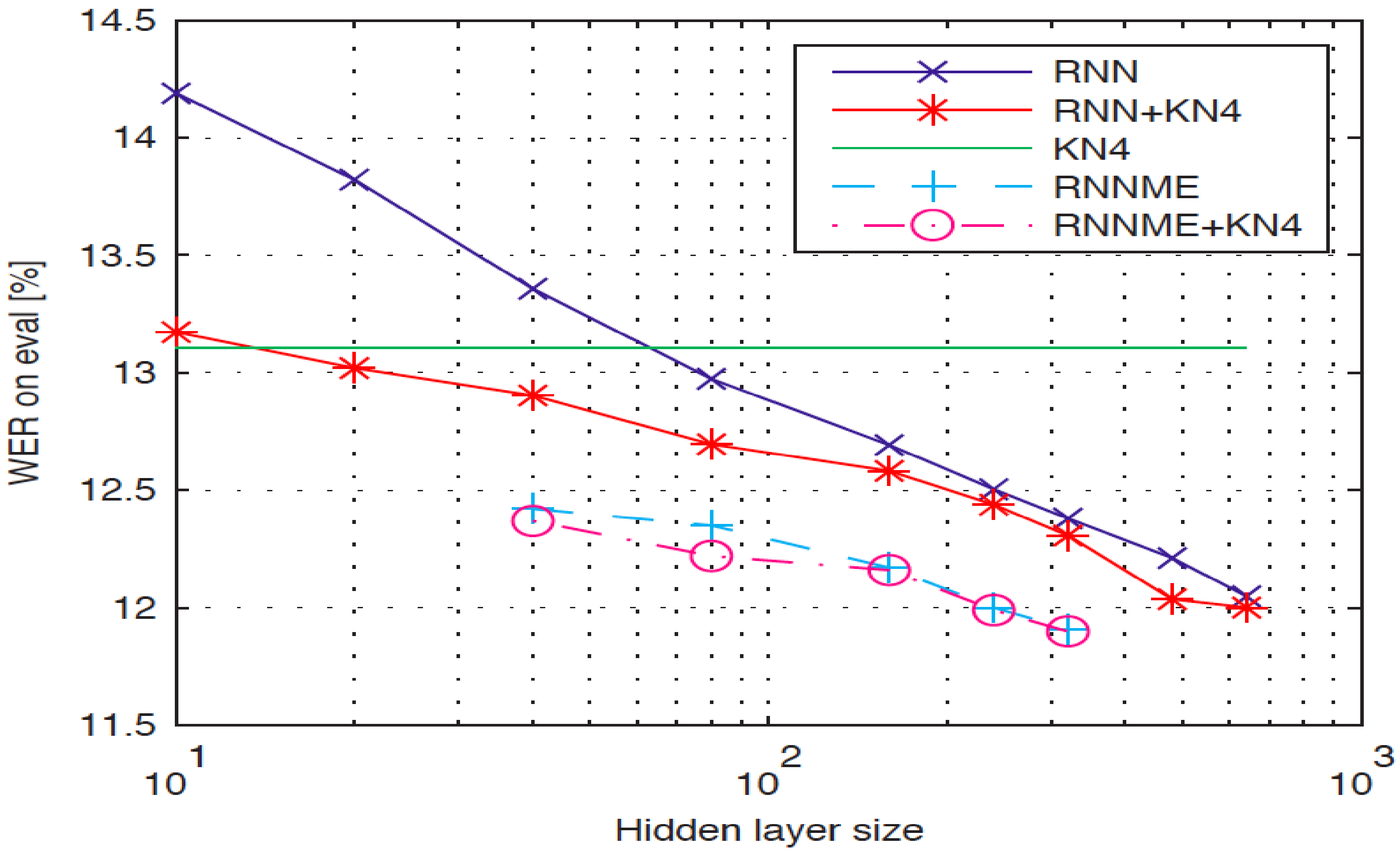
Where $f$ is the sigmod function and $g$ is the softmax funciton

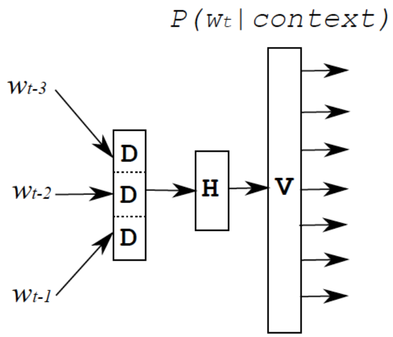
# Word2vec--RNNLM

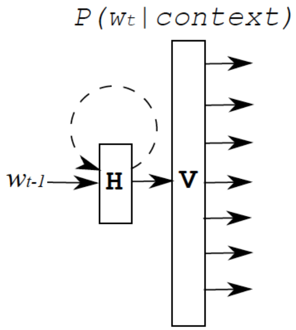
- Training:
  - Stochastic Gradient Descent (SGD)
    Objective(Error) function:
    $$error(t) = d(t) - y(t)$$
    where d(t) is the desired vector, i.e w(t)
  - Go through all the training data iteratively, and update the weight matrices U, V and W online (after processing every word)
  - Training is performed in several epochs (usually 5-10)

- Where is the word representation?
  - $U$, with each column

# Word2vec--RNNLM



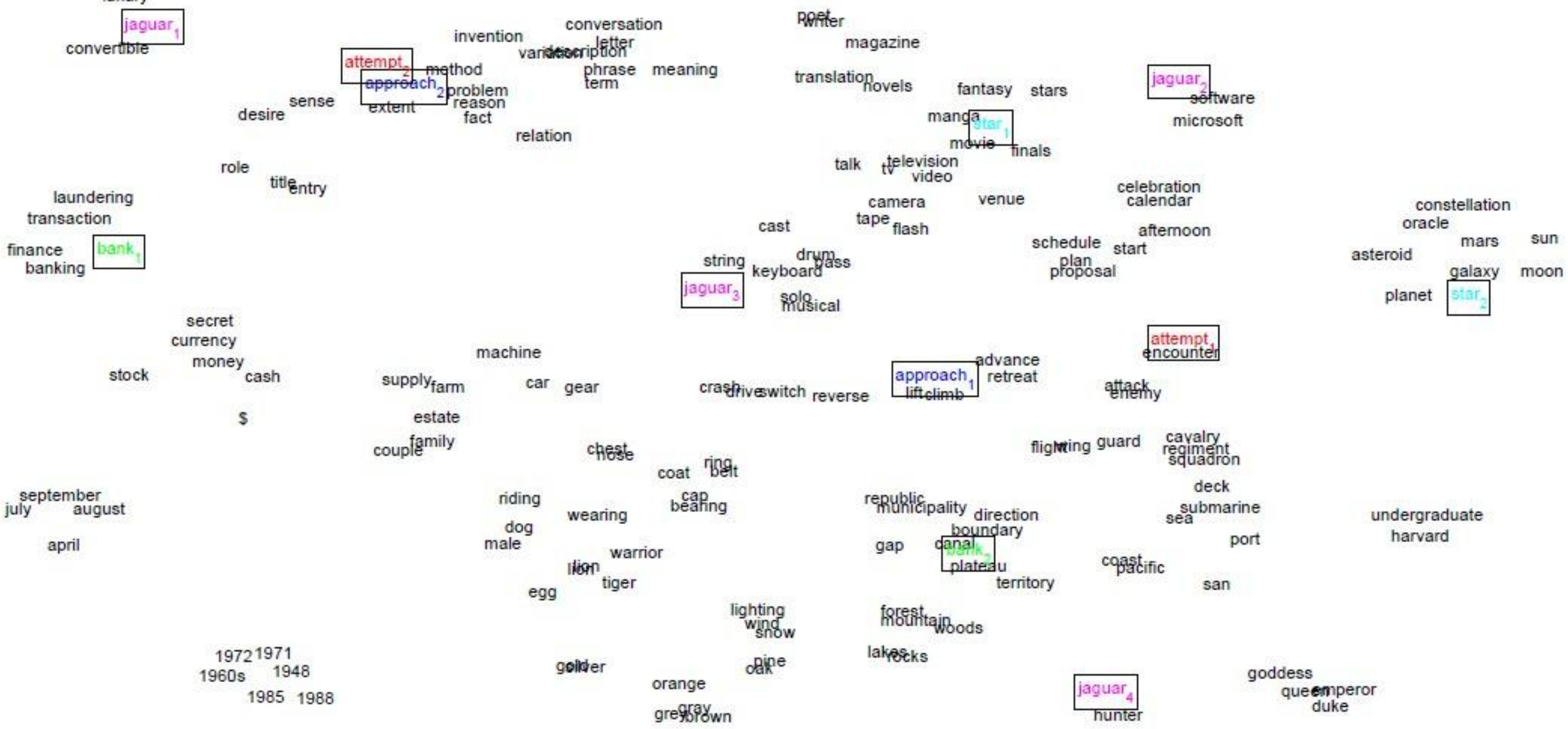




Experiments on Broadcast News NIST-RT04

# Word2vec--Huang

Improving Word Representations via Global Context and Multiple Word Prototypes (Huang, et al. ACL 2013)



# Word2vec--Huang

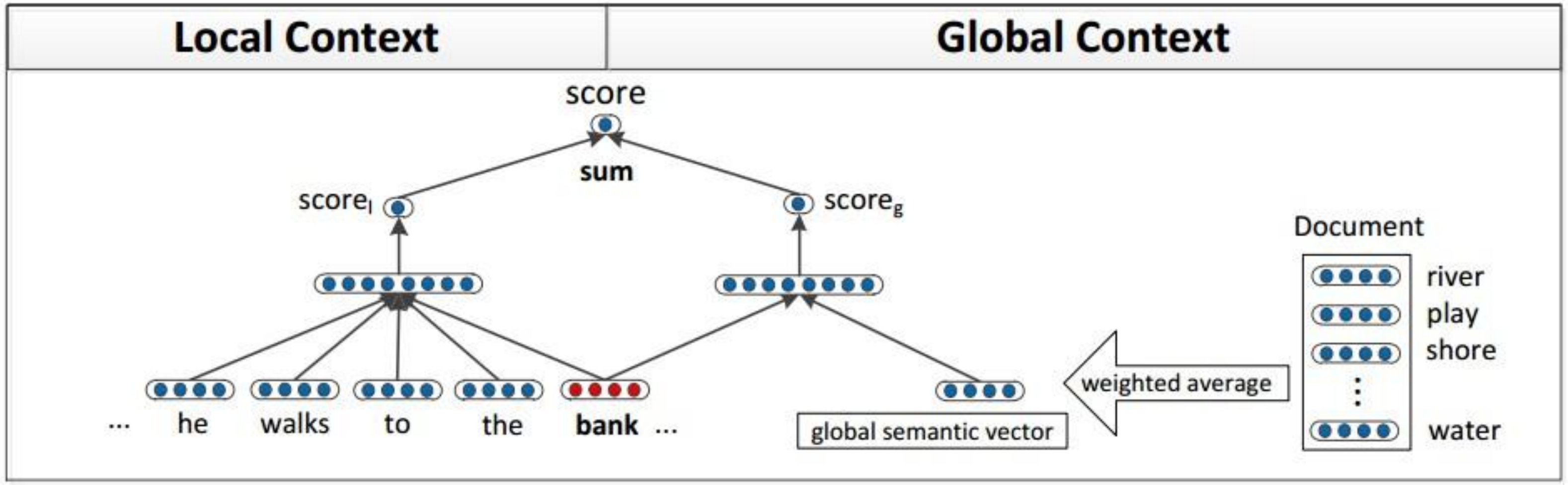
- Improve Collobert & Weston's model
  - Training objective:

$$\theta \rightarrow \sum \sum Max\{0, 1 - S_{pos} + S_{neg}\}$$

$$\downarrow$$

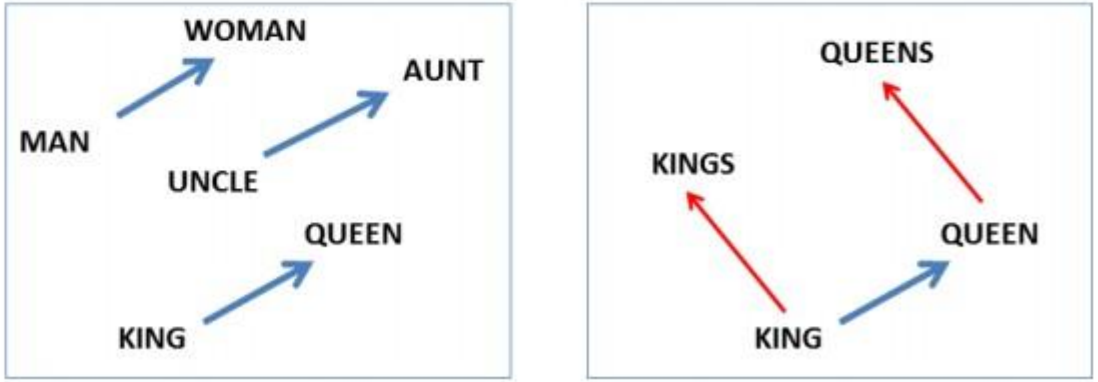
$$\theta \rightarrow \sum \sum Max\{0, 1 - S_{pos,d} + S_{neg,d}\}$$

where $d$ is the document (weighted sum of words in $d$)



# Word2vec—Interesing finding

- Measuring Linguistic Regularity
  - Syntactic/Semetic Test




$C(king)-C(queen) \approx C(man)-C(woman)$

$C(king)-C(man)+C(woman) \approx C(queen)$

These representations are surprisingly good at capturing syntactic and semantic regularities in language, and that each relationship is characterized by a relation-specific vector offset.

Exploiting Similarities among Languages for Machine Translation (Mikolov, et al. 2013 http://arxiv.org/pdf/1309.4168.pdf)

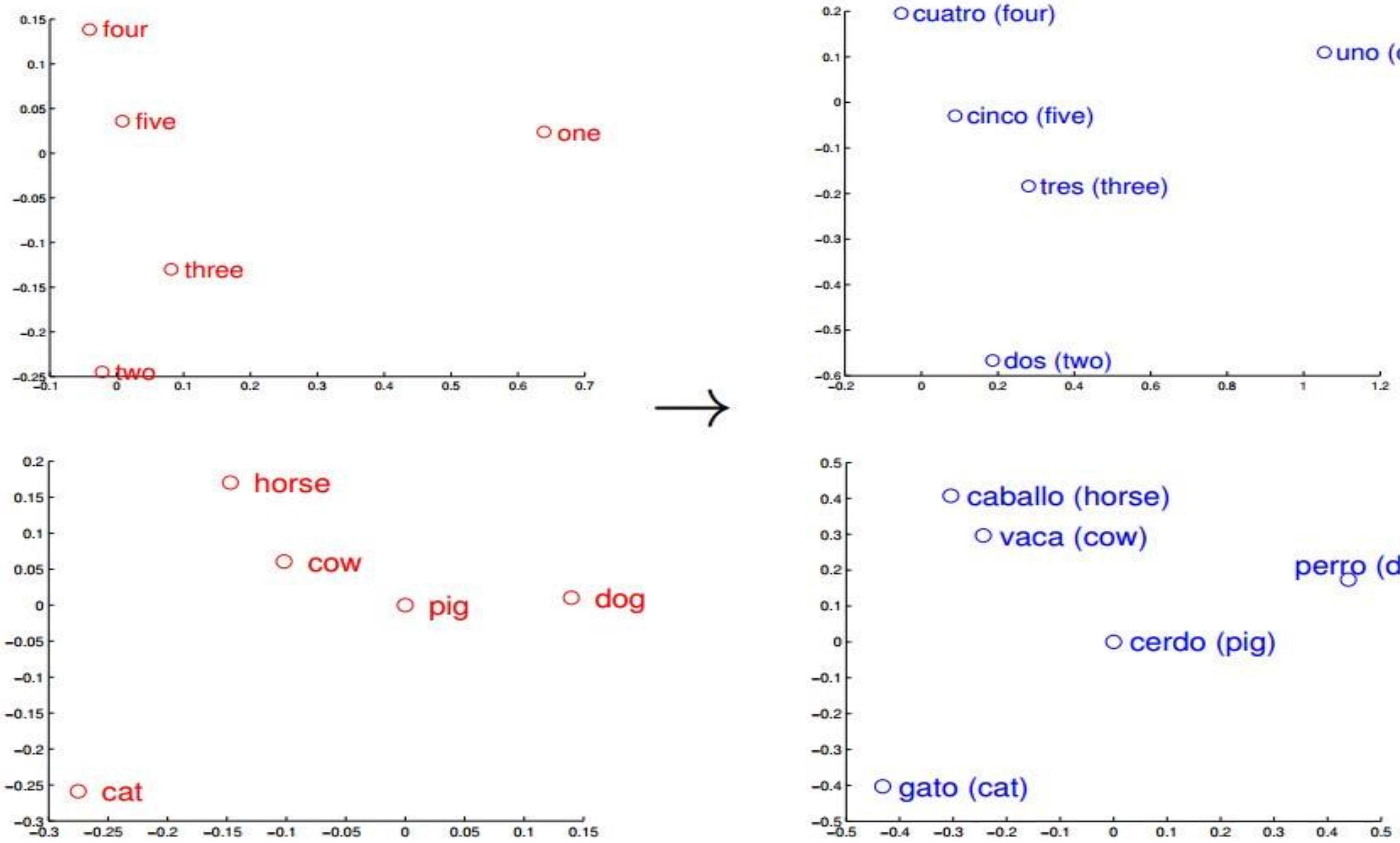



Figure 1: Distributed word vector representations of numbers and animals in English (left) and Spanish (right). The five vectors in each language were projected down to two dimensions using PCA, and then manually rotated to accentuate their similarity. It can be seen that these concepts have similar geometric arrangements in both spaces, suggesting that it is possible to learn an accurate linear mapping from one space to another.

# Word2vec--summary

- Useful tools

1. google https://code.google.com/p/word2vec/

   train word vector

2. SENNA http://ml.nec-labs.com/senna/

   Part of Speech (POS)
   Chunking (CHK)
   Name Entity Recognition (NER)
   Semantic Role Labeling (SRL)
   Syntactic Parsing (PSG)

3. Word Representations for NLP http://metaoptimize.com/projects/wordreprs/

   Neural language model (Collobert + Weston)
   HLBL language model
   Brown clusters
   CRF Chunking with word representations
   Perceptron NER with word representations
   Random indexing word representations

4 . Huang http://www.socher.org/index.php/Main/ImprovingWordRepresentationsViaGlobalContextAndMultipleWordPrototypes

5. RNNLM Toolkit http://www.fit.vutbr.cz/~imikolov/rnnlm/

# References


- [1] Holger Schwenk; *CSLM - A modular Open-Source Continuous Space Language Modeling Toolkit*, in Interspeech, August 2013.
- [2] Y. Bengio, and R. Ducharme. A neural probabilistic language model. In *Neural Information Processing Systems,* volume 13, pages 932-938. 2001
- [3] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient Estimation of Word Representations in Vector Space. In Proceedings of Workshop at ICLR, 2013.
- [4]Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg Corrado, and Jeffrey Dean. Distributed Representations of Words and Phrases and their Compositionality. In Proceedings of NIPS, 2013.
- [5]Tomas Mikolov, Wen-tau Yih, and Geoffrey Zweig. Linguistic Regularities in Continuous Space Word Representations. In Proceedings of NAACL HLT, 2013.
- [6] ngram smoothing http://nlp.stanford.edu/~wcmac/papers/20050421-smoothing-tutorial.pdf
- Mikolov, Tomas, Wen-tau Yih, and Geoffrey Zweig. **Linguistic regularities in continuous space word representations**. Proceedings of NAACL-HLT. 2013.

- [7] Ronan Collobert, Jason Weston, Léon Bottou, Michael Karlen, Koray Kavukcuoglu and Pavel Kuksa.**Natural Language Processing (Almost) from Scratch**. Journal of Machine Learning Research (JMLR), 12:2493-2537, 2011.
- [8] Andriy Mnih & Geoffrey Hinton. **Three new graphical models for statistical language modelling**. International Conference on Machine Learning (ICML). 2007.
- [9] Andriy Mnih & Geoffrey Hinton. **A scalable hierarchical distributed language model**. The Conference on Neural Information Processing Systems (NIPS) (pp. 1081–1088). 2008.
- [10] Mikolov Tomáš. **Statistical Language Models based on Neural Networks**. PhD thesis, Brno University of Technology. 2012.
- [11] Turian, Joseph, Lev Ratinov, and Yoshua Bengio. **Word representations: a simple and general method for semi-supervised learning**. Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics (ACL). 2010.

- Thanks
- Q & A