SENTENCE REORDERING FOR QA TASK

Maochen Guan

2016-07

TASK DEF.

Sentence reordering is the first step of QA.
Need to be efficient for vast amount of docs.
Conventional semantic parsing and logic inference won't work.

Example:
Q: What is the price of IBM share?

C1: IBM is located in the U.S.
C2: Linkedin's stock price was increased by 40% during the day.
C3: IBM did win Jeopardy prize.
C4: Currently, IBM's price is about 150.79 per share.

After reordering:
Sentence - Prob
C4 - 0.89
C2 - 0.40
C3 - 0.33
C1 - 0.28

Only consider single answer with 1 knowledge each case.
Not valid: Tom went to the school and John is happy.

SENTENCE REORDERING STRATEGY

Query - Q
Candidate - C

For every candidate Ci:
(Q, Ci) -> Feature Extractors -> ML Models -> Predict Output (Oi)

Sort Ci based on Oi.

Evaluation:
Take the highest prob candidate as actual output.
10-fold CV, report F1 score.

DATA AND FEATURES

Training Data:
WikiQA — Yi Yang, Wen-tau Yih, and Christopher Meek (2015) MSFT Research
3,047 questions and 29,258 sentences where 1,473 sentences were labeled as positive answers.

Factorial Question only.

QuestionID	Q11
Question	how big is bmc software in houston, tx
DocumentID	D11
DocumentTitle	BMC Software
SentenceID	D11-3
Sentence	Employing over 6,000, BMC is often credited …
Label	1

Training Data Tuple: (context, question, sentence, isAnswer)

Features:
Question Type
Question Document ID
Candidate Document ID
Question Vector (100 dimension)
Candidate Vector (100 dimension)

Features end up with 203 dimensions.

QUESTION TYPE CLASSIFICATION

Question Type Distribution:

Syntactic form
Wh-type Question 87.7% (544)
Yes-no Question 9.5% (59)
Imperative (Information request) 2.6% (16)
Declarative (Answer to clarification) 0.2% (1)

For the Factorial Questions:

Dataset:
Xin Li, Dan Roth, Learning Question Classifiers. COLING'02, Aug., 2002.

Size: 5500 labeled question.

Example of training data:
NUM:date What is the date of Boxing Day ?

Features:
N-Gram word features
Brown Cluster features
POS feats
Dependency features from Stanford parser
Named entity features
Wordnet sense

Trained a MaxEnt QT classifier based on UIUC dataset.

Evaluation: (10-fold CV)
Coarse class F1-score: 95.13%
Fine class F1-score: ??? (Missing)

Syntactic classification of user utterances from (Kato et al., 2006)

Class	#	Class	#
ABBREVIATION	18	term	19
abbreviation	2	vehicle	7
expression	16	word	0
DESCRIPTION	153	**HUMAN**	171
definition	126	group	24
description	13	individual	140
manner	7	title	4
reason	7	description	3
ENTITY	174	**LOCATION**	195
animal	27	city	44
body	5	country	21
color	12	mountain	5
creative	14	other	114
currency	8	state	11
disease/medicine	3	**NUMERIC**	289
event	6	code	1
food	7	count	22
instrument	1	date	146
lang	3	distance	38
letter	0	money	9
other	19	order	0
plant	7	other	24
product	9	period	18
religion	1	percent	7
sport	3	speed	9
substance	20	temp	7
symbol	2	vol.size	4
technique	1	weight	4

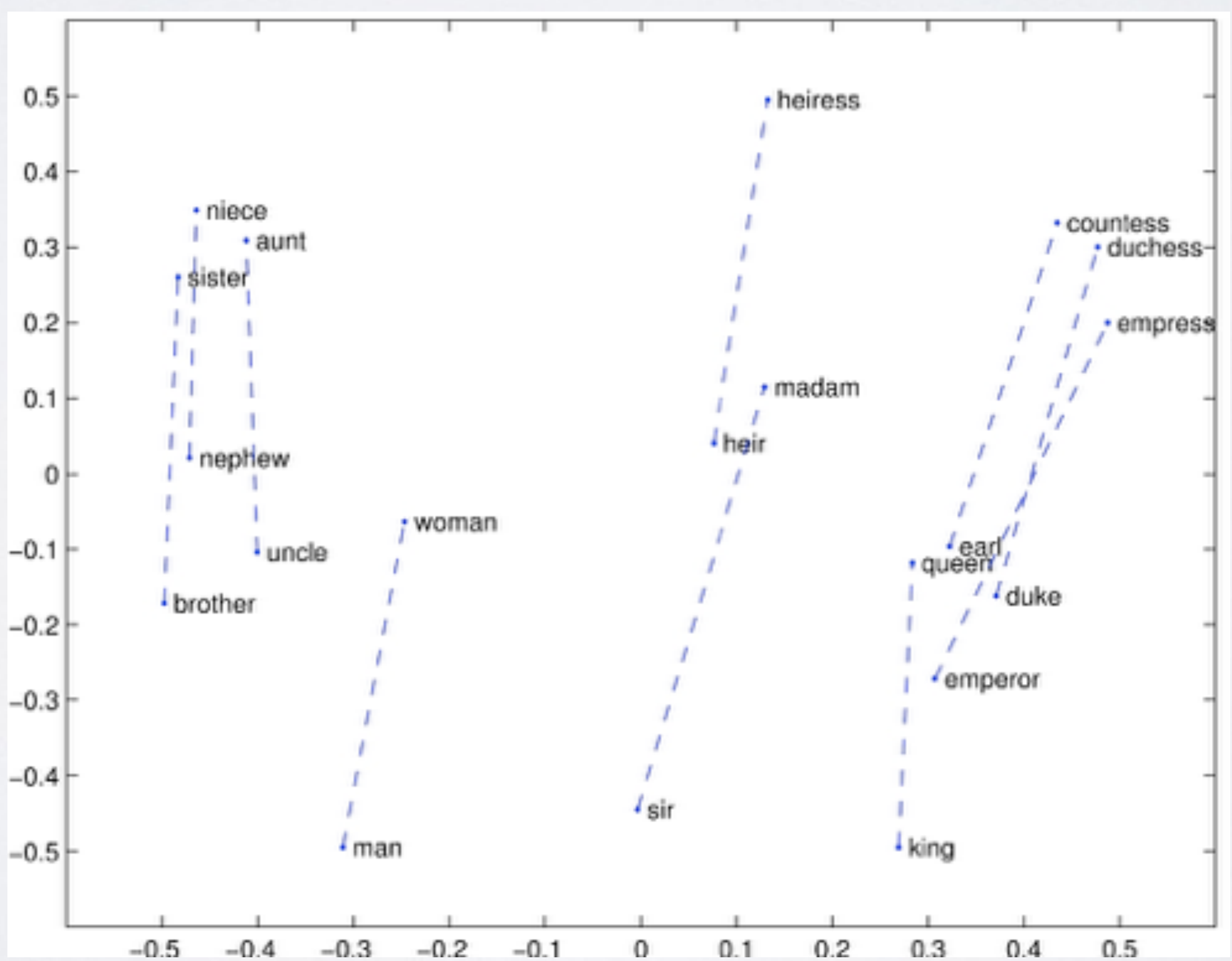
WORD EMBEDDINGS

Use PCA or any Single Value Decomposition based approach?

Problems with SVD:
1. Computational cost scales quadratically for n x m matrix: O(mn^2) flops (when n<m)
2. Linear transformation

— Direct Learn Low Dimensional Word Vectors

<How to Generate a Good Word Embedding?> Cite as: arXiv:1507.05523 [cs.CL]
Compared different Word2vec models, NNLM as well as GloVe.



ONE-HOT WORD REPRESENTATION

- NLP treats words mainly (rule-based/statistical approaches at least) as atomic symbols:

  Love   Candy   Store

- or in vector space:

  [0 0 0 0 0 **1** 0 0 0 0 0 0 0 0 0 0 0 ...]

- also known as "one hot" representation.

- Its problem ?

Candy [0 0 0 0 0 **1** 0 0 0 0 0 0 0 0 0 0 0 ...]   AND

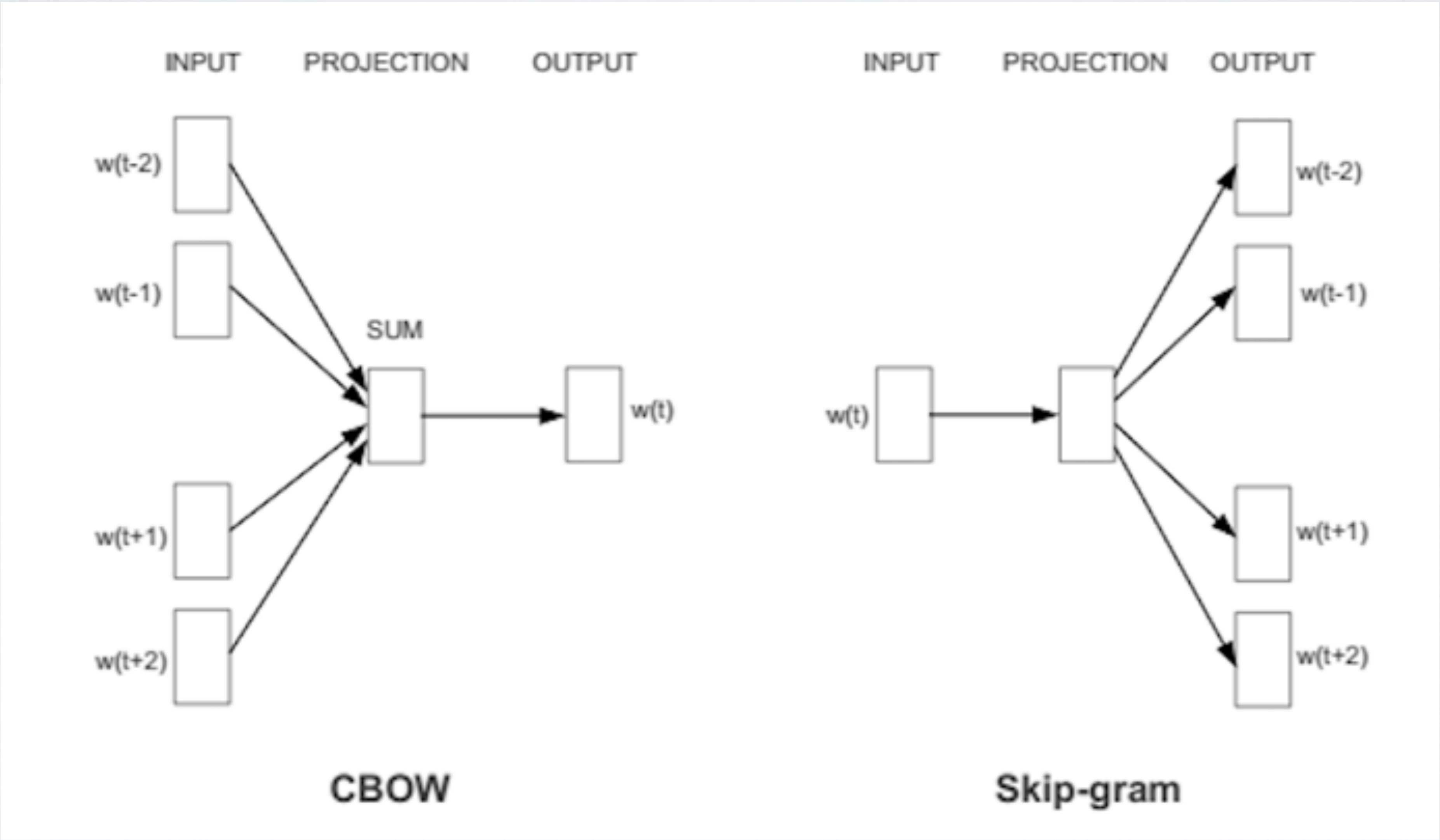
Store [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 **1** 0 ...]   = 0 !

WORD2VEC

Input: "one-hot" word vector

$$J(\theta) = \frac{1}{T}\sum_{t=1}^{T} \sum_{-m\leq j\leq m, j\neq 0} \log p(w_{t+j}|w_t)$$

$$p(o|c) = \frac{\exp\left(u_o^T v_c\right)}{\sum_{w=1}^{W} \exp\left(u_w^T v_c\right)}$$



Loss Function:

$$E = -\log p(w_t|w_{t-c}..w_{t+c})$$

$$E = -\log p(w_{t-c}..w_{t+c}|w_t)$$

WORD2VEC

For example window size c = 1, sentence:
"I like learning ."

First window computes gradients for:
Internal vector v_{like} and external vectors u_I and $u_{learning}$

Compute Grad for all words

$$\theta = \begin{bmatrix} v_{aardvark} \\ v_a \\ \vdots \\ v_{zebra} \\ u_{aardvark} \\ u_a \\ \vdots \\ u_{zebra} \end{bmatrix} \in \mathbb{R}^{2dV}$$

WORD2VEC CONCLUSION

Pros:
1. No annotated data required
2. Fast to train
3. Can do semantic logical deduction

Cons:
1. Lost word order info
2. After projection layer, there is just a logistic function to map to the output.
3. Cannot distinguish negation.

These two sentences will yield to the same set of vectors
Tom likes Mary == Mary likes Tom

Autoencoder - Semantic Hashing

1. Observations from word2vec cons (Last slides)
2. Window size = Avg sentence length (Both question and candidates) ~20 words.
(300 dimension * 20 = 6000)

Training Data:
All WikiQA Q&As, treat all data as sentence, no labeled data involved in here.

Avg. len. of question 9.59
Avg. len. of sentence 28.85

Approach:

Take w2v as initial input layer (6000 nodes), 3 RBM hidden layer (3000, 800, 100), use tanh as activation function.

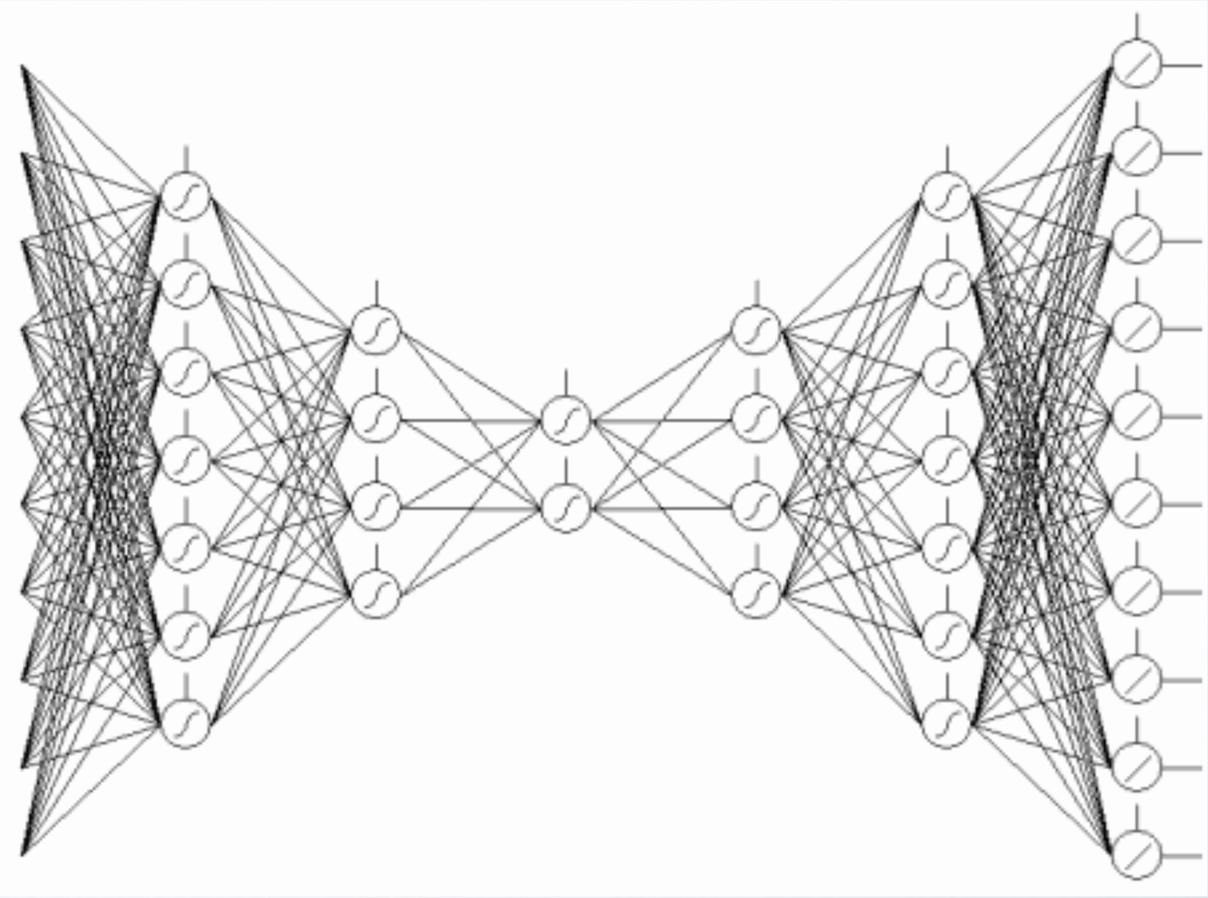
Right side is W_(n) transpose.

Objective Function: $J(A,s) = \|As - x\|_2^2 + \lambda\|s\|_1$

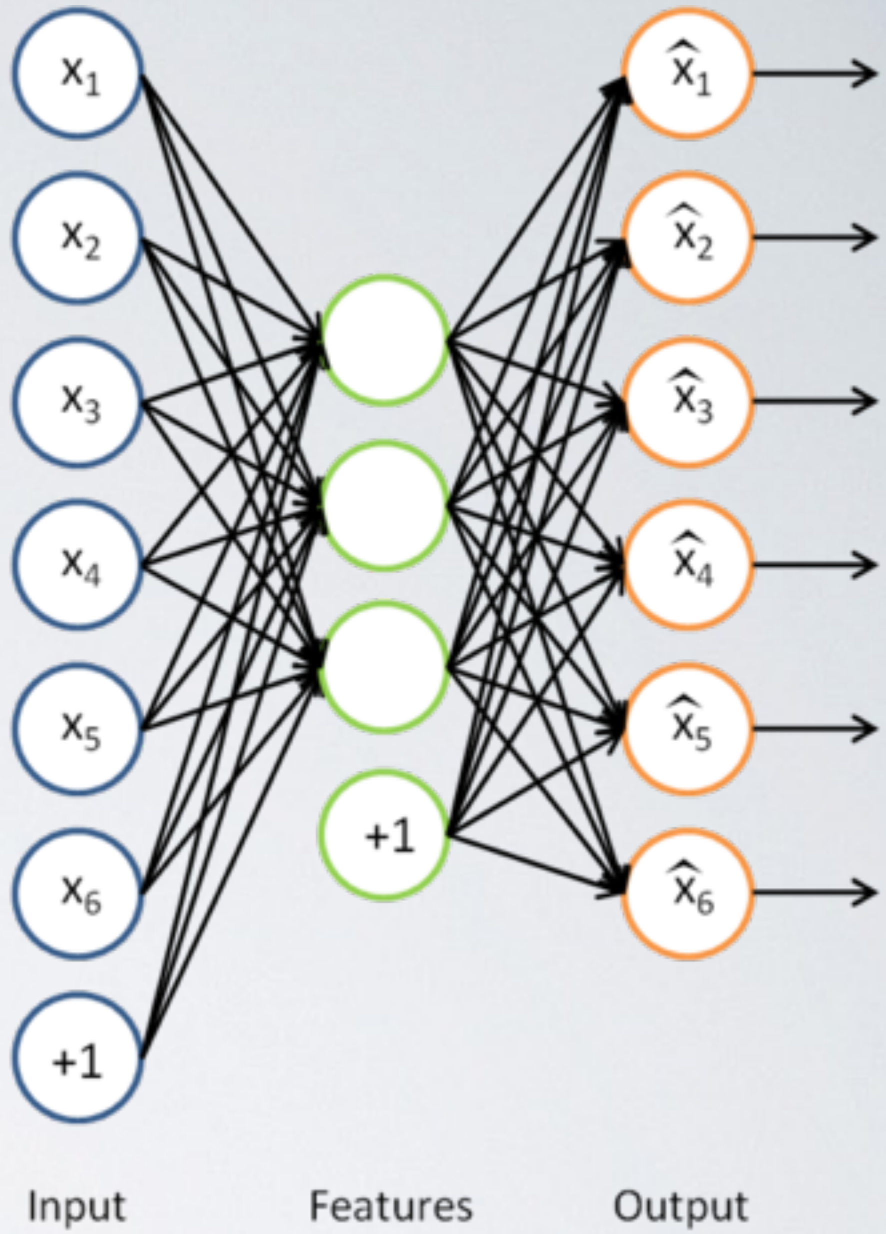
$$A_j^T A_j \leq 1 \ \forall j$$

s - sparse coding
A - weight matrix

$$\|x\|_k = \left(\sum |x_i^k|\right)^{\frac{1}{k}}$$



Autoencoder - Semantic Hashing



Learning Alg:

Objective Function: $J(A,s) = \|As - x\|_2^2 + \lambda \|s\|_1$

$$J(A,s) = \|As - x\|_2^2 + \lambda \sqrt{s^2 + \epsilon} + \gamma \|A\|_2^2$$

$$\sum_k \sqrt{s_k^2 + \epsilon} \qquad \sum_r \sum_c A_{rc}^2$$

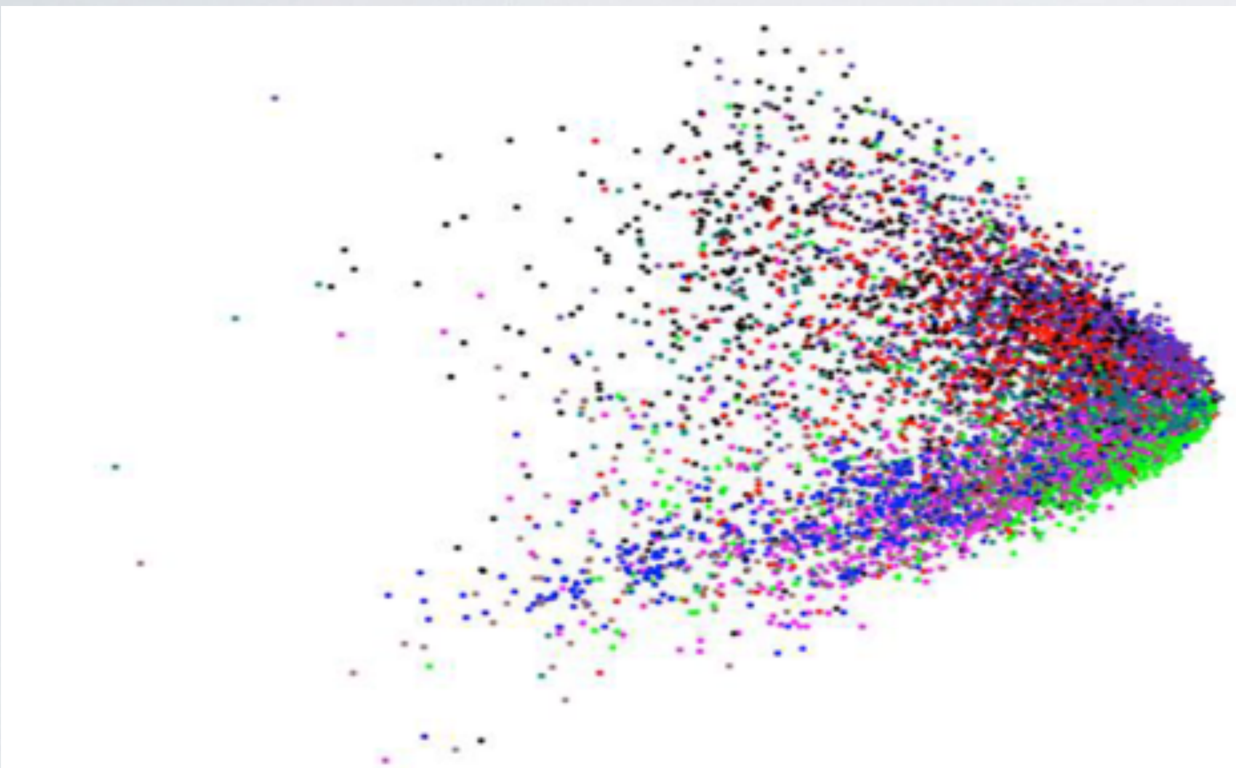
1. Initialize A randomly
2. Repeat until convergence

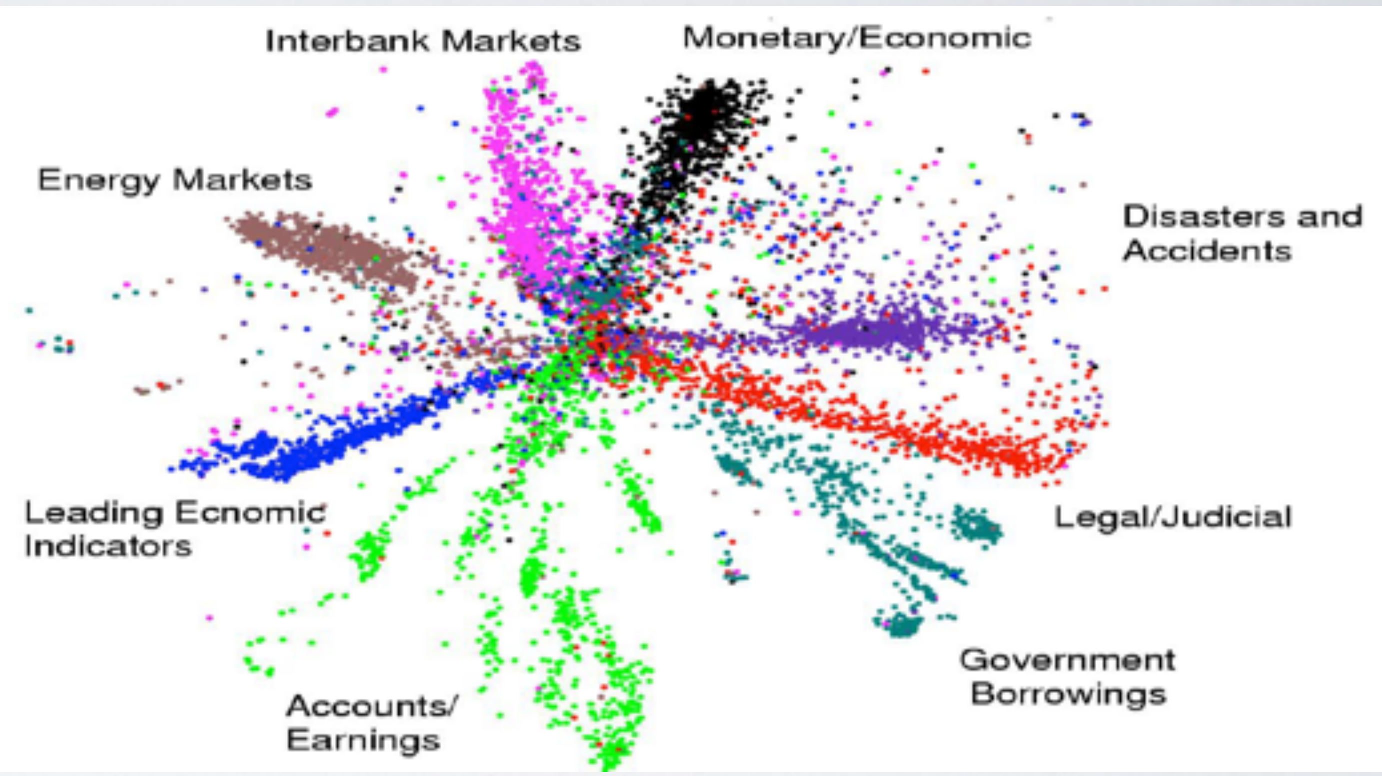
   I. Find the s that minimizes J(A,s) for the A found in the previous step

   II. Solve for the A that minimizes J(A,s) for the s found in the previous step

RESULT OF AUTOENCODER VS PCA

400,000 Reuters business news stories
Different color for different doc categories





PCA 2-dim

Autoencoder 2-dim

From Geoffrey Hinton's NN slides

FEATURES

Question Vector/Candidate Vector: 100 dim each.
Question Type: QT classifier (trained on UIUC data)

ContextID matcher
1. Parse the sentence (Dependency tree)
2. Take the subj + obj as sentence topic
3. Use word2vec embeddings correspondingly to find the closest distance predefined existing context, use that as context id.

Dependency parse tree example (Stanford CoreNLP):

I have a car.

ID	Form	Lemma	CPOS	POS		Head Id	Dep Label			
1	I	I	PRON	PRP	_	2	nsubj	_	_	
2	have	have	VERB	VBP	_	0	root	_	_	_
3	a	a	DET	DT	_	4	det	_	_	_
4	car	car	NOUN	NN	_	2	dobj	_	_	_
5	.	.	PUNCT	.	_	2	punct			

ML MODEL — MAXENTROPY

Derived from LR (softmax version LR)
Discriminative model (Only needs P(y|x))
For experiment, we are using OpenNLP MaxEnt package

$$p^*(y \mid x) = \frac{\exp\left(\sum_i \lambda_i f_i(x,y)\right)}{\sum_y \exp\left(\sum_i \lambda_i f_i(x,y)\right)}$$

$$H(p) = -\sum_{x \in A \times B} p(x) \log p(x)$$

Generalized Iterative Scaling

1. Random init lambda_j
2. Define $$C = \max_{x \in \varepsilon} \sum_{j=1}^{k} f_j(x)$$
3. Add a bias feature $$\forall x \in \varepsilon \quad f_{k+1}(x) = C - \sum_{j=1}^{k} f_j(x)$$
4. Compute $$E_{p^{(n)}} f_j = \sum_{x \in \varepsilon} p^{(n)}(x) f_j(x) \qquad E_p f_j = d_j$$
5. Update rule $$\lambda_j^{(n+1)} = \lambda_j^{(n)} + \frac{1}{C} \left(\log \frac{d_i}{E_{p^{(n)}} f_j}\right)$$
6. Repeat step 3, 4 until converge

TRAINING & EVALUATION

Training:
Balance Pos/Neg samples: Randomly pickup negative samples corresponding to the positive.

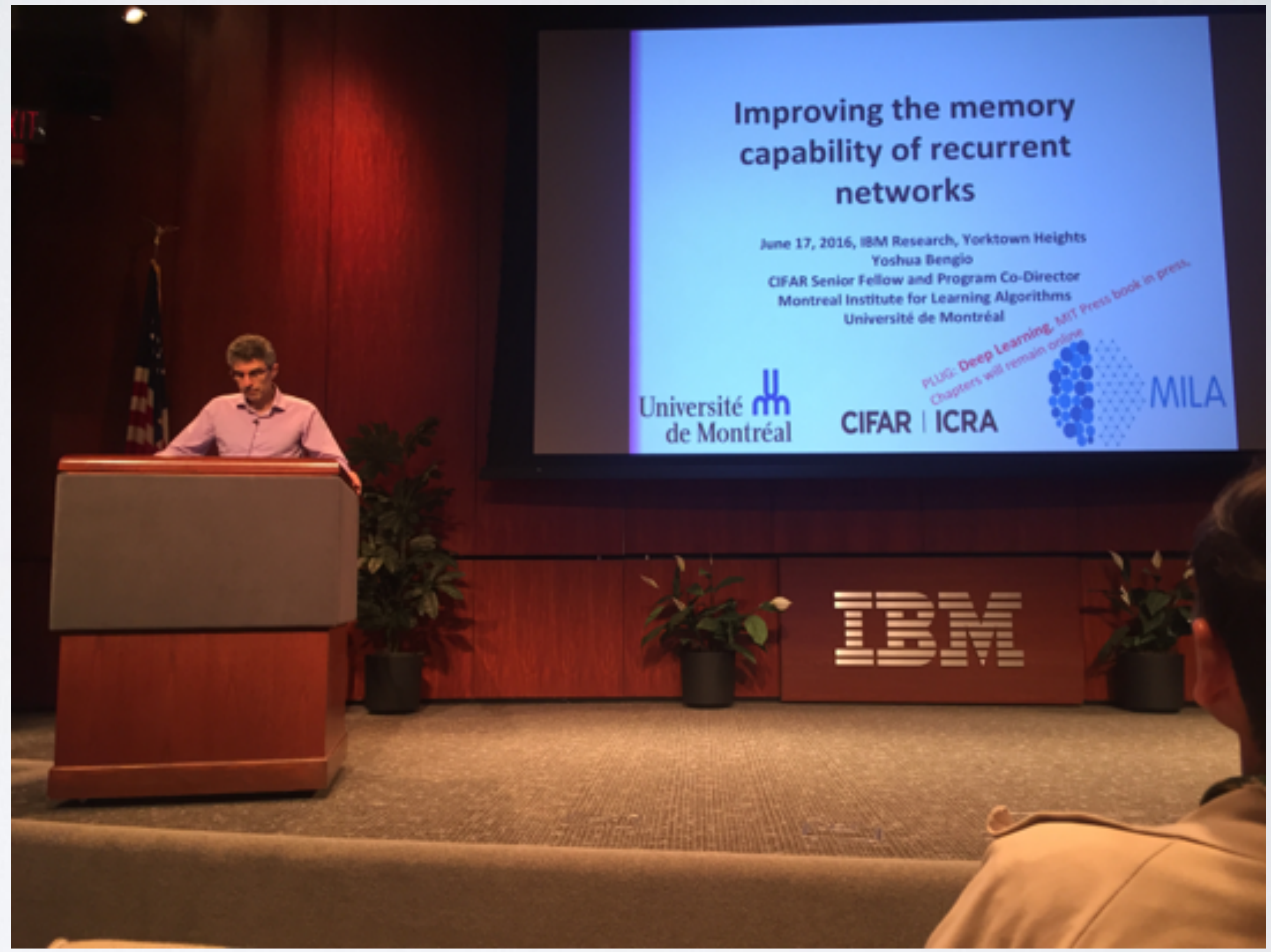
Evaluation:
Calculate P(isMatch | Q,C; lambda)

Current status:
F-score is available for sentence embedding.
F-score is available for QT classifier.

Still trying to evaluate the whole system, missing F-score

THINGS FORWARD

- Sequence model - RNN?
- Coreference
- Entity recognition
- Entity linking
- Ontology



Slides available at www.maochen.org/playground/playground.html