Natural Language Processing (NLP) Applications of Deep Learning

(taken from IPAM / CIFAR 2012 summer school on deep learning, with parts coming from ACL'2012 tutorial on Deep Learning for NLP, with Richard Socher and Chris Manning)

Yoshua Bengio

IFT6266 lecture

Deep Learning models have already achieved impressive results for NLP

Neural Language Model

[Mikolov et al. Interspeech 2011]

Model \ WSJ task	Eval WER
KN5 Baseline	**17.2**
Discriminative LM	**16.9**
Recurrent NN combination	**14.4**

MSR MAVIS Speech System

[Dahl et al. 2012; Seide et al. 2011; following Mohamed et al. 2011]

"The algorithms represent the first time a company has released a deep-neural-networks (DNN)-based speech-recognition algorithm in a commercial product."

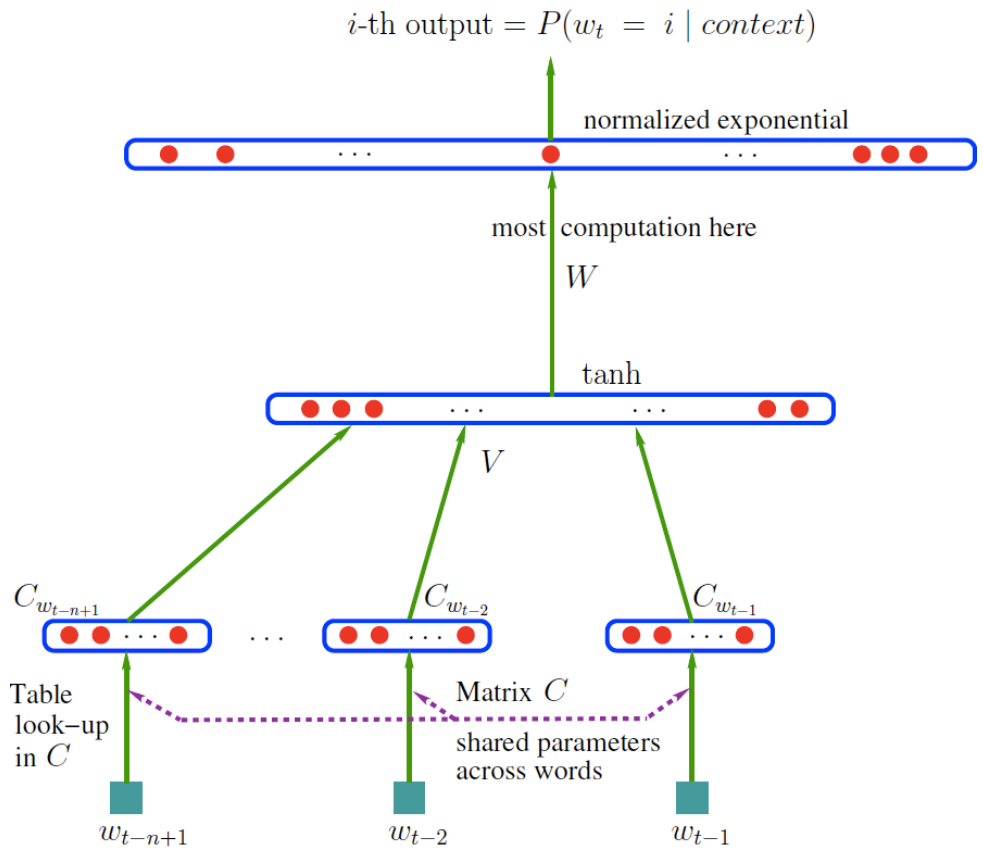
Acoustic model & training	Recog \ WER	RT03S FSH	Hub5 SWB
GMM 40-mix, BMMI, SWB 309h	1-pass –adapt	**27.4**	**23.6**
CD-DNN 7 layer x 2048, SWB 309h	1-pass –adapt	**18.5** (–33%)	**16.1** (–32%)
GMM 72-mix, BMMI, FSH 2000h	*k*-pass +adapt	**18.6**	**17.1**

Existing NLP Applications

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

Neural Language Model

- *Bengio et al NIPS'2000 and JMLR 2003 "A Neural Probabilistic Language Model"*
  - Each word represented by a distributed continuous-valued code
  - Generalizes to sequences of words that are semantically similar to training sequences



Language Modeling

- Predict P(next word | previous word)
- Gives a probability for a longer sequence
- Applications to Speech, Translation and Compression
- Computational bottleneck: large vocabulary V means that computing the output costs #hidden units x |V|.

The standard word representation

The vast majority of rule-based **and** statistical NLP work regards words as atomic symbols: hotel, conference, walk

In vector space terms, this is a vector with one 1 and a lot of zeroes

[0 0 0 0 0 0 0 0 0 0 1 0 0 0 0]

Dimensionality: 20K (speech) – 50K (PTB) – 500K (big vocab) – 13M (Google 1T)

We call this a "one-hot" representation. Its problem:

motel [0 0 0 0 0 0 0 0 0 0 1 0 0 0 0] AND
hotel [0 0 0 0 0 0 0 1 0 0 0 0 0 0 0] = 0

Distributional similarity based representations

You can get a lot of value by representing a word by means of its neighbors

> "You shall know a word by the company it keeps"
>
> (J. R. Firth 1957: 11)

One of the most successful ideas of modern statistical NLP

government debt problems turning into **banking** crises as has happened in

saying that Europe needs unified **banking** regulation to replace the hodgepodge

↖ These words will represent *banking* ↗

You can vary whether you use local or large context to get a more syntactic or semantic clustering

Class-based (hard) and soft clustering word representations

Class based models learn word classes of similar words based on distributional information (~ class HMM)

- Brown clustering (Brown et al. 1992)
- Exchange clustering (Martin et al. 1998, Clark 2003)
- Desparsification and great example of unsupervised pre-training

Soft clustering models learn for each cluster/topic a distribution over words of how likely that word is in each cluster

- Latent Semantic Analysis (LSA/LSI), Random projections
- Latent Dirichlet Analysis (LDA), HMM clustering

Neural word embeddings as a distributed representation

Similar idea, but thing of each dimension as an attribute, not as a cluster membership

Combine vector space semantics with the prediction of probabilistic models (Bengio et al. 2003, Collobert & Weston 2008, Turian et al. 2010)

In all of these approaches, including deep learning models, a word is represented as a dense vector (TODO: sparsity)

linguistics =

$$\begin{pmatrix} 0.286 \\ 0.792 \\ -0.177 \\ -0.107 \\ 0.109 \\ -0.542 \\ 0.349 \\ 0.271 \end{pmatrix}$$

Neural word embeddings - visualization

Advantages of the neural word embedding approach

Compared to a method like LSA, neural word embeddings can become more meaningful through adding supervision from one or multiple tasks

- For instance, sentiment is usually not captured in unsupervised word embeddings but can be in neural word vectors

We can build representations for large linguistic units

- See below

Contrastive Sampling of Negative Examples (Collobert et al. JMLR 2011)

Idea: A word and its context is a positive training sample; a random word in that same context gives a negative training sample:


cat chills on a mat


cat chills Jeju a mat

Similar: Implicit negative evidence in Contrastive Estimation, (Smith and Eisner 2005)

A neural network for learning word vectors

How do we formalize this idea? Ask that

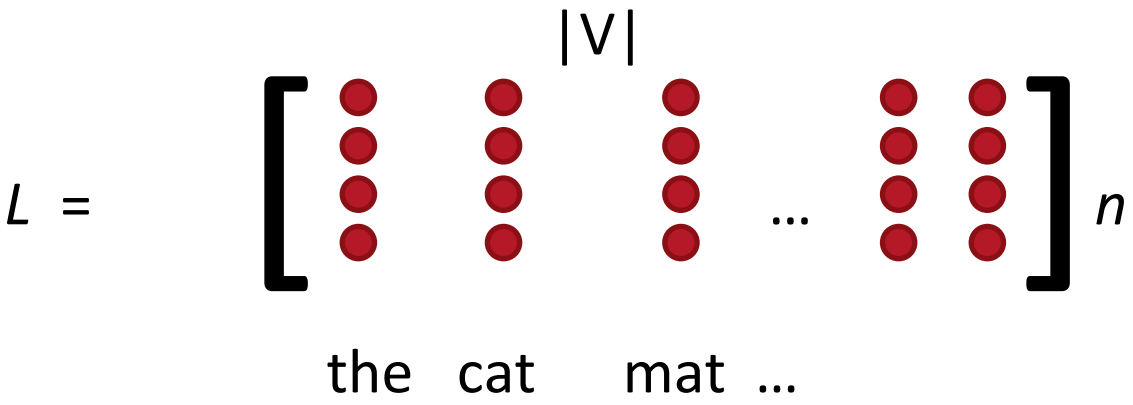
score(cat chills on a mat) > score(cat chills Jeju a mat)

How do we compute the score?

- With a neural network
- Each word is associated with an *n*-dimensional vector

Word embedding matrix

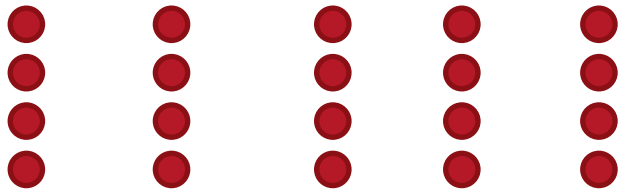
- Initialize all word vectors randomly to form a word embedding matrix $L \in \mathbb{R}^{n \times |V|}$



- These are the word features we want to learn
- Also called a look-up table
  - Conceptually you get a word's vector by left multiplying a one-hot vector *e* by *L*: $x = Le$

Word vectors as input to a neural network

- score(cat chills on a mat)
- To describe a phrase, retrieve (via index) the corresponding vectors from *L*



- Then concatenate them to 5*n* vector:
- *x* =[●●●● ●●●● ●●●● ●●●● ●●●●]
- How do we then compute score(x)?

The secret sauce is the unsupervised pre-training on a large text collection

(Collobert & Weston 2008; Collobert et al. 2011)

	POS WSJ (acc.)	NER CoNLL (F1)
State-of-the-art*	97.24	89.31
Supervised NN	**96.37**	**81.47**
Unsupervised pre-training followed by supervised NN**	**97.20**	**88.87**
+ hand-crafted features***	97.29	89.59

* Representative systems: POS: (Toutanova et al. 2003), NER: (Ando & Zhang 2005)

** 130,000-word embedding trained on Wikipedia and Reuters with 11 word window, 100 unit hidden layer – **for 7 weeks!** – then supervised task training

***Features are character suffixes for POS and a gazetteer for NER

Supervised refinement of the unsupervised word representation helps

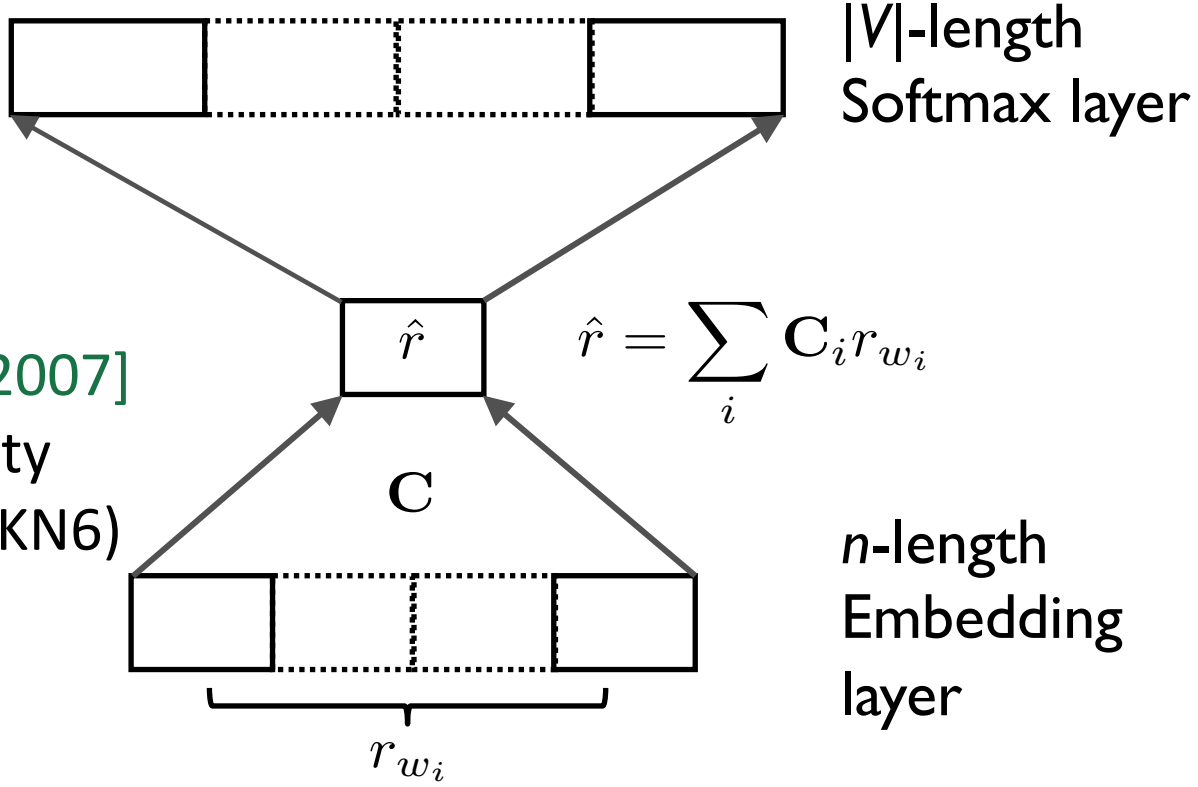
	POS WSJ (acc.)	NER CoNLL (F1)
Supervised NN	96.37	81.47
NN with Brown clusters	96.92	87.15
Fixed embeddings*	**97.10**	**88.87**
C&W 2011**	**97.29**	**89.59**

* Same architecture as C&W 2011, but word embeddings are kept constant during the supervised training phase

** C&W is unsupervised pre-train + supervised NN + features model of last slide

Bilinear Language Model

- Even a linear version of the Neural Language Model works better than n-grams

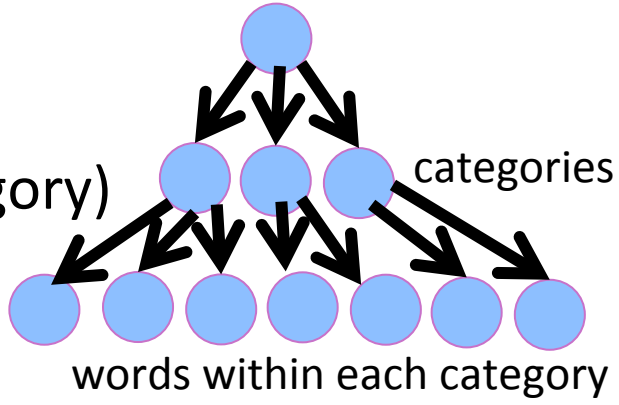


- [Mnih & Hinton 2007]
- APNews perplexity down from 117 (KN6) to 96.5

Language Modeling Output Bottleneck

- [Schwenk et al 2002]: only predict most frequent words (short list) and use n-gram for the others

- *[Morin & Bengio 2005; Blitzer et al 2005; Mnih & Hinton 2007,2009; Mikolov et al 2011]*: hierarchical representations, multiple output groups, conditionally computed, predict
  - P(word category | context)
  - P(sub-category | context, category)
  - P(word | context, sub-category, category)

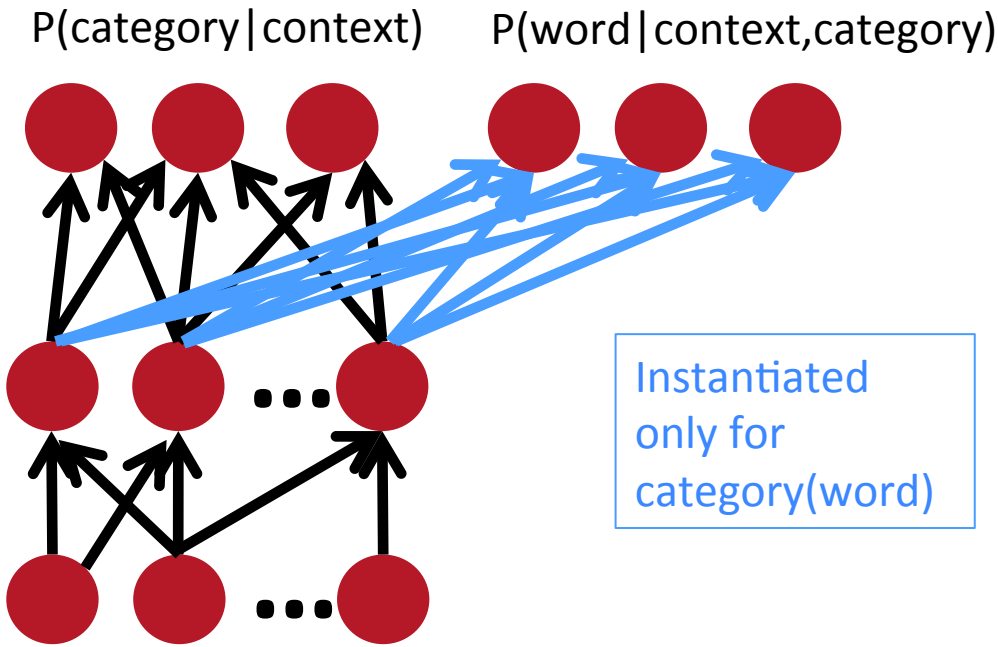


  - Hard categories, can be arbitrary [Mikolov et al 2011]

Language Modeling Output Bottleneck: Hierarchical word categories

Compute

P(word|category,context)
only for
category=category(word)



Language Modeling Output Bottleneck: Sampling Methods

- Importance sampling to recover next-word probabilities [Bengio & Senecal 2003, 2008]

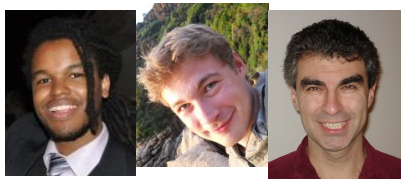
- Contrastive Sampling of negative examples, with a ranking loss [Collobert et al, 2008, 2011]
  - (no probabilities, ok if the goal is just to learn word embeddings)

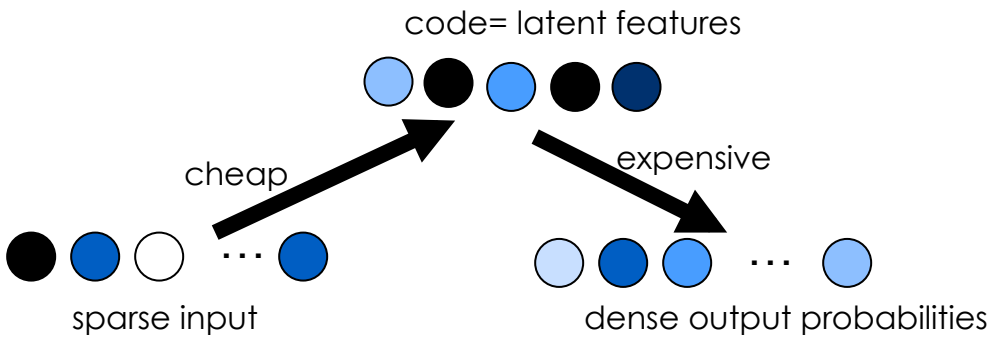
- Importance sampling for reconstructing bag-of-words [Dauphin et al 2011]

Sampled Reconstruction Trick

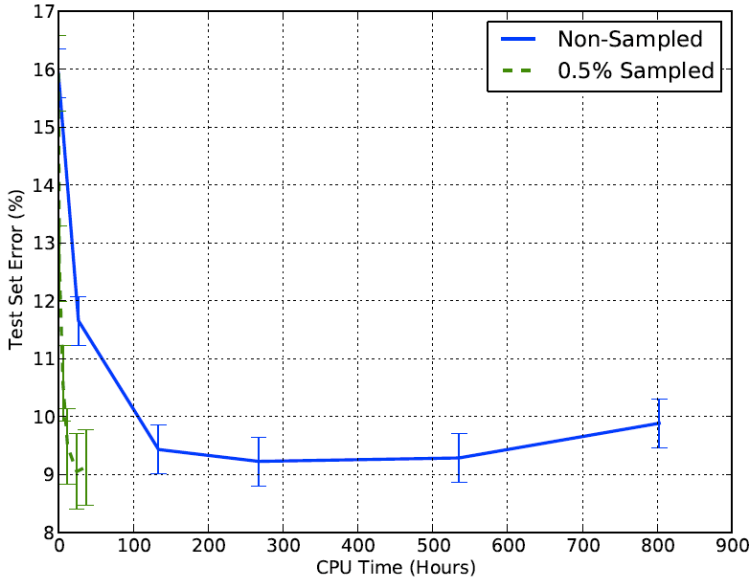
[Dauphin et al, ICML 2011]

- Auto-encoders and RBMs reconstruct the input, which is sparse and high-dimensional





- Applied to bag-of-words input for sentiment analysis, with denoising auto-encoders



- Always reconstruct the non-zeros in the input, and reconstruct as many randomly chosen zeros

Representing Sparse High-Dimensional Stuff: Sampled Reconstruction

$$L(\mathbf{x}, \mathbf{z}) = \sum_{k}^{d} \quad H(\mathbf{x}_k, \mathbf{z}_k)$$

Stochastic reweighted loss

$$\hat{L}(\mathbf{x}, \mathbf{z}) = \sum_{k} \frac{\mathbf{p}_k}{\mathbf{q}_k} H(\mathbf{x}_k, \mathbf{z}_k)$$

Sample which inputs to reconstruct

$$\hat{\mathbf{p}} \in \{0, 1\}^d \text{ with } \hat{\mathbf{p}} \sim P(\hat{\mathbf{p}}|\mathbf{x})$$

$$\mathbf{q}_k = E[\hat{\mathbf{p}}_k | k, \mathbf{x}, \tilde{\mathbf{x}}]$$

Importance sampling reweighting

$$\text{Let } \mathcal{C}(\mathbf{x}, \tilde{\mathbf{x}}) = \{k : \mathbf{x}_k = 1 \text{ or } \tilde{\mathbf{x}}_k = 1\}$$

Minimum-variance: guess wrong reconstructions

$$P(\hat{\mathbf{p}}_k = 1 | \mathbf{x}_k) = \begin{cases} 1 & \text{if } k \in \mathcal{C}(\mathbf{x}, \tilde{\mathbf{x}}) \\ |\mathcal{C}(\mathbf{x}, \tilde{\mathbf{x}})|/d_x & otherwise \end{cases}$$

As many randomly chosen other bits

Recurrent Neural Net Language Modeling for ASR

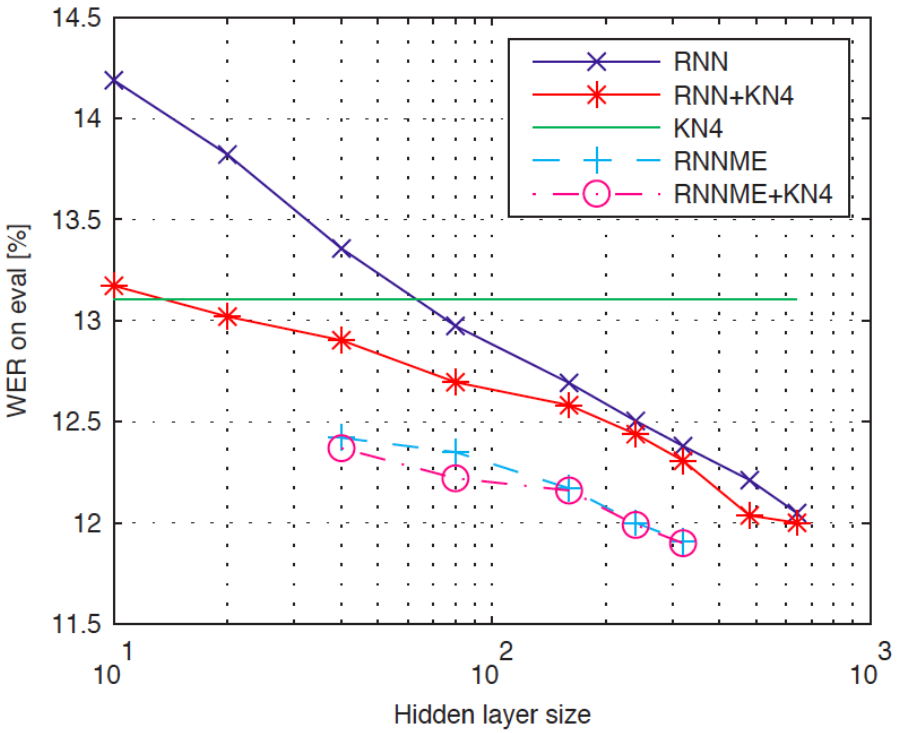
- [Mikolov et al 2011]

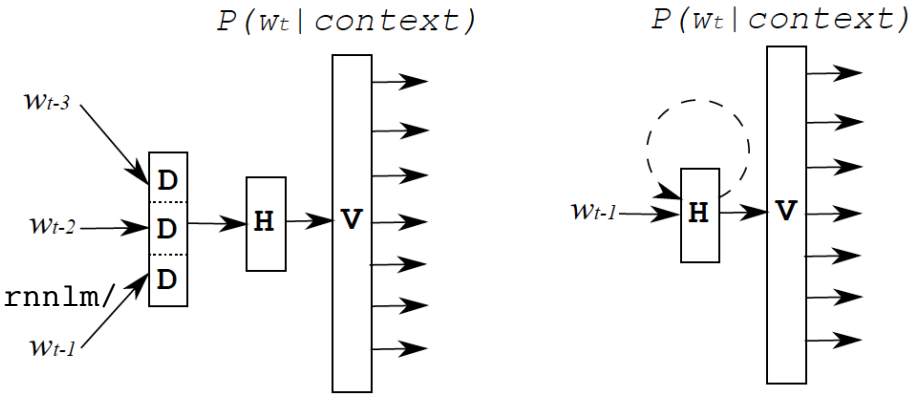
Bigger is better…
experiments on Broadcast News NIST-RT04

perplexity goes from 140 to 102

Paper shows how to train a recurrent neural net with a single core in a few days, with > 1% absolute improvement in WER

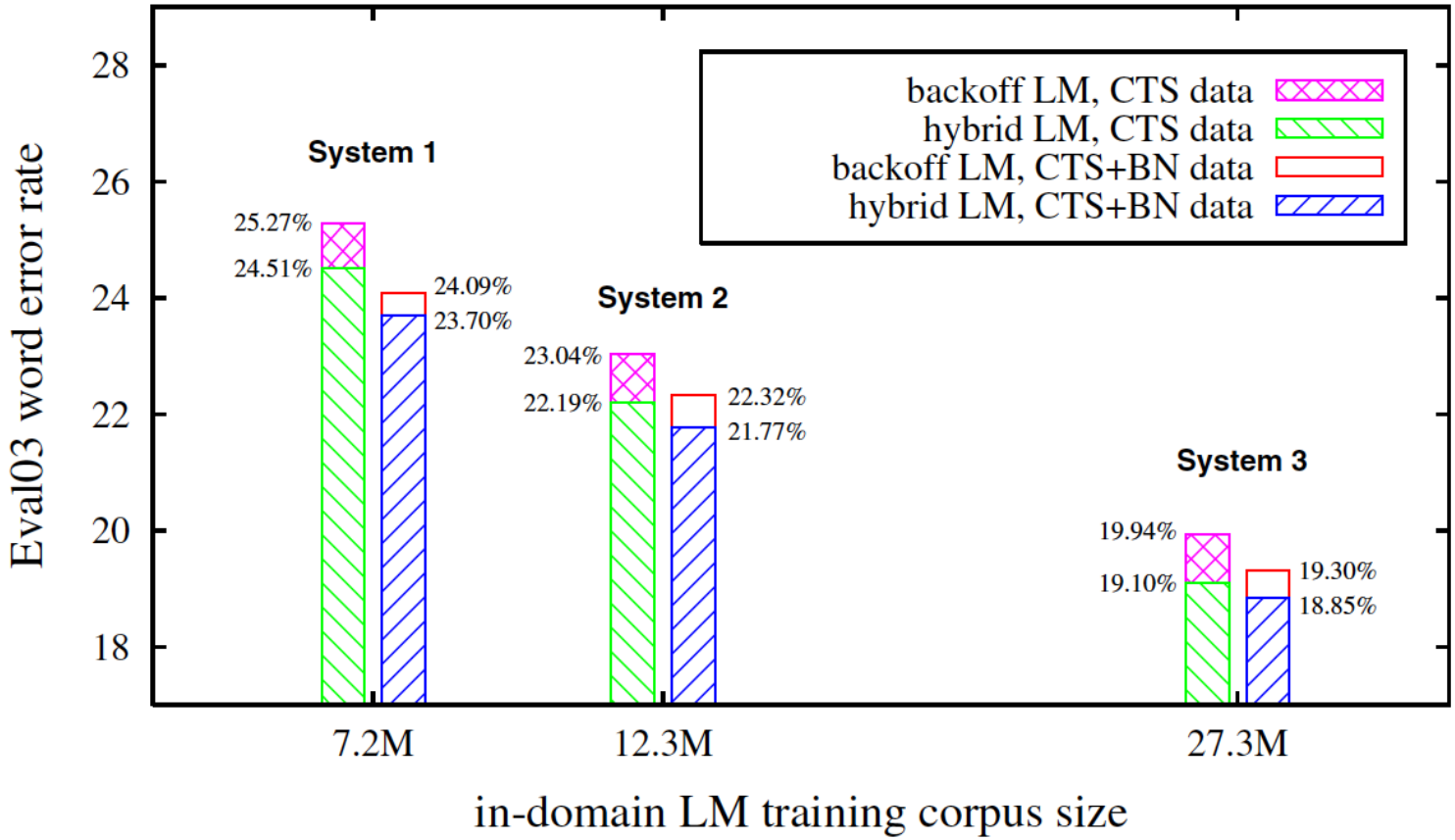
Code: `http://www.fit.vutbr.cz/~imikolov/rnnlm/`





Neural Net Language Modeling for ASR

- [Schwenk 2007], real-time ASR, perplexity AND word error rate improve (CTS evaluation set 2003), perplexities go from 50.1 to 45.5

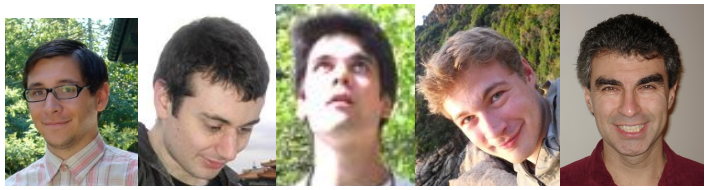


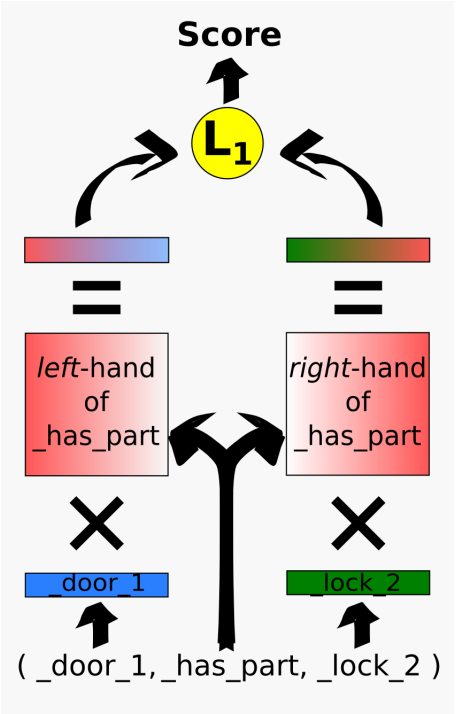
Application to Statistical Machine Translation

- Schwenk (NAACL 2012 workshop on the future of LM)
  - 41M words, Arabic/English bitexts + 151M English from LDC
- Perplexity down from 71.1 (6 Gig back-off) to 56.9 (neural model, 500M memory)
- +1.8 BLEU score (50.75 to 52.28)
- Can take advantage of longer contexts
- Code: `http://lium.univ-lemans.fr/cslm/`

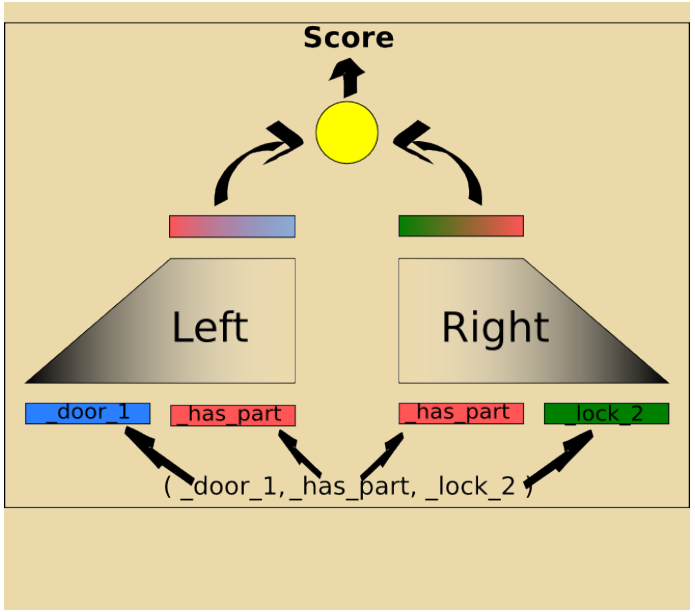
Modeling Semantics

Learning Structured Embeddings of Knowledge Bases, (Bordes, Weston, Collobert & Bengio, AAAI 2011)

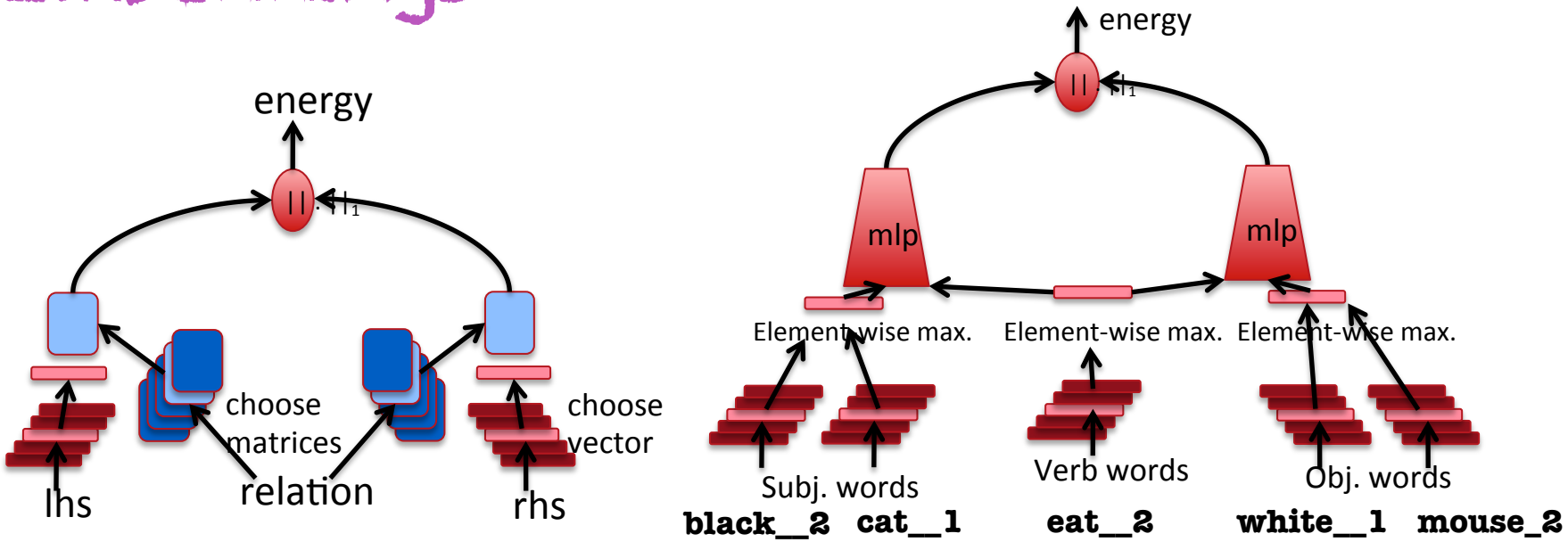




Joint Learning of Words and Meaning Representations for Open-Text Semantic Parsing, (Bordes, Glorot, Weston & Bengio, AISTATS 2012)



Modeling Relations: Operating on Embeddings



Model (lhs, relation, rhs)

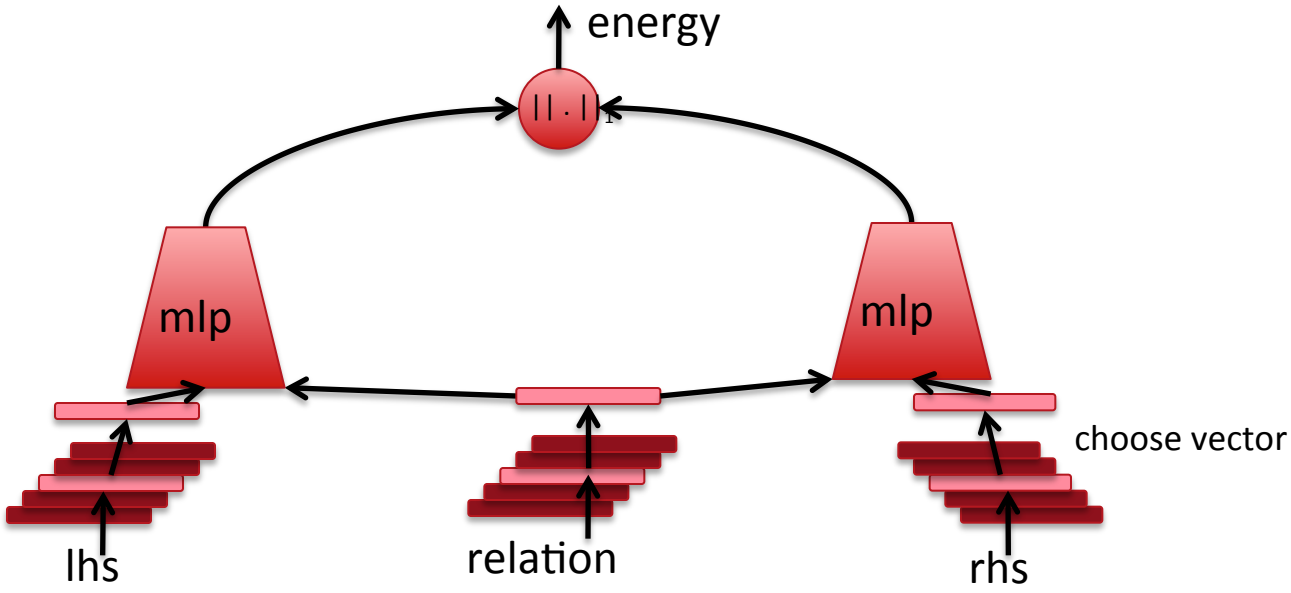
Each concept = 1 embedding vector

Each relation = 2 matrices. **Matrix or mlp acts as operator.**

Ranking criterion

Energy = low for training examples, high o/w

Allowing Relations on Relations



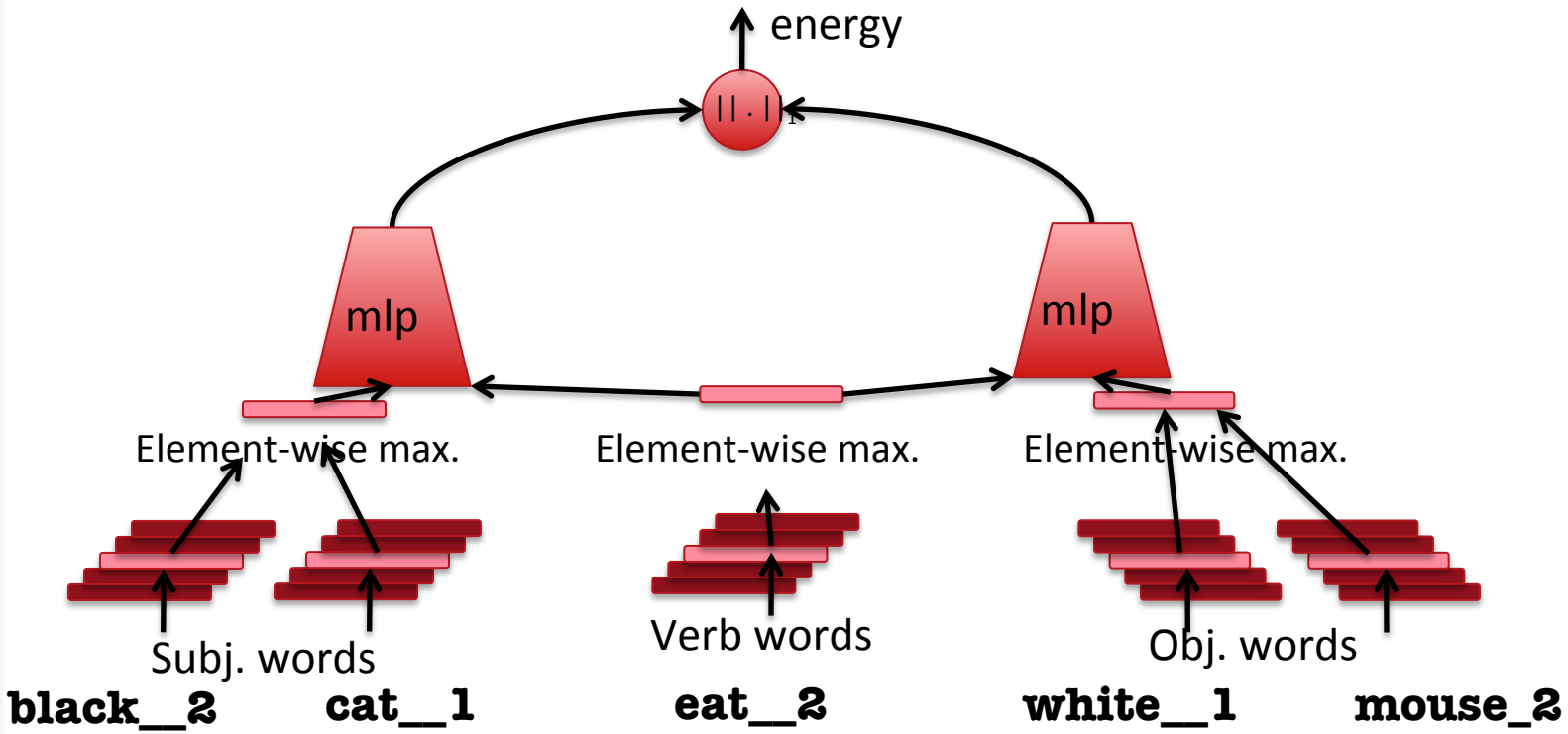
Verb = relation. Too many to have a matrix each.

Each concept = 1 embedding vector

Each relation = 1 embedding vector

Can handle relations on relations on relations

Training on Full Sentences

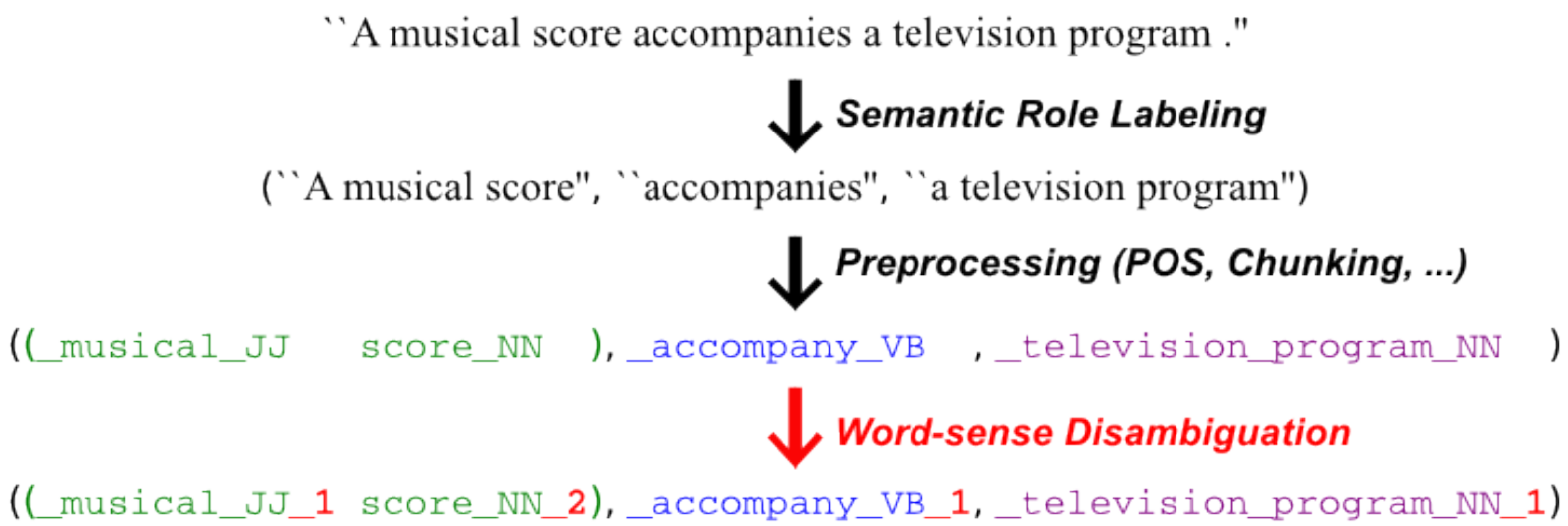


Use SENNA (Collobert et al 2011) = embedding-based NLP tagger for Semantic Role Labeling, breaks sentence into (subject, verb, object) phrases

→ Use max-pooling to aggregate embeddings of words inside each part

Open-Text Semantic Parsing

- 3 steps:



- last formula defines the Meaning Representation (MR).

Training Criterion

- Intuition: if an entity of a triplet was missing, we would like our model to predict it correctly i.e. to give it the lowest energy. For example, this would allow us to answer questions like "what is part of a car?"
- Hence, for any training triplet $x_i = (lhs_i, rel_i, rhs_i)$ we would like:

  (1) $E(lhs_i, rel_i, rhs_i) < E(lhs_j, rel_i, rhs_i)$,

  (2) $E(lhs_i, rel_i, rhs_i) < E(lhs_i, rel_j, rhs_i)$,

  (3) $E(lhs_i, rel_i, rhs_i) < E(lhs_i, rel_i, rhs_j)$,

That is, the energy function E is trained to rank training samples below all other triplets.

Contrastive Sampling of Neg. Ex.=
pseudo-likelihood + uniform sampling of negative variants

Train by stochastic gradient descent:

1. Randomly select a positive training triplet $x_i = (lhs_i, rel_i, rhs_i)$.
2. Randomly select constraint (1), (2) or (3) and an entity $\tilde{e}$:
   - If constraint (1), construct negative triplet $\tilde{x} = (\tilde{e}, rel_i, rhs_i)$.
   - Else if constraint (2), construct $\tilde{x} = (lhs_i, \tilde{e}, rhs_i)$.
   - Else, construct $\tilde{x} = (lhs_i, rel_i, \tilde{e})$.
3. If $E(x_i) > E(\tilde{x}) - 1$ make a gradient step to minimize:

   $\max(0, 1 - E(\tilde{x}) + E(x_i))$.
4. Constraint embedding vectors to norm 1

Question Answering: implicitly adding new relations to WN or FB

	Model (All)	TextRunner
lhs	**_army_NN_1**	*army*
rel	**_attack_VB_1**	*attacked*
top ranked *rhs*	_troop_NN_4	*Israel*
	_armed_service_NN_1	*the village*
	_ship_NN_1	*another army*
	_territory_NN_1	*the city*
	_military_unit_NN_1	*the fort*
top ranked *lhs*	_business_firm_NN_1	*People*
	_person_NN_1	*Players*
	_family_NN_1	*one*
	_payoff_NN_3	*Students*
	_card_game_NN_1	*business*
rel	**_earn_VB_1**	*earn*
rhs	**_money_NN_1**	*money*

MRs inferred from text define triplets between WordNet synsets.

Model captures knowledge about relations between nouns and verbs.

→ Implicit addition of new relations to WordNet!

→ Generalize Freebase!

Embedding Nearest Neighbors of Words & Senses

_mark_NN	_mark_NN_1	_mark_NN_2
_indication_NN	_score_NN_1	_marking_NN_1
_print_NN_3	_number_NN_2	_symbolizing_NN_1
_print_NN	_gradation_NN	_naming_NN_1
_roll_NN	_evaluation_NN_1	_marking_NN
_pointer_NN	_tier_NN_1	_punctuation_NN_3

_take_VB	_canary_NN	_different_JJ_1
_bring_VB	_sea_mew_NN_1	_eccentric_NN
_put_VB	_yellowbird_NN_2	_dissimilar_JJ
_ask_VB	_canary_bird_NN_1	_same_JJ_2
_hold_VB	_larus_marinus_NN_1	_similarity_NN_1
_provide_VB	_mew_NN	_common_JJ_1

Word Sense Disambiguation

- Senseval-3 results
  (only sentences with Subject-Verb-Object structure)

MFS=most frequent sense

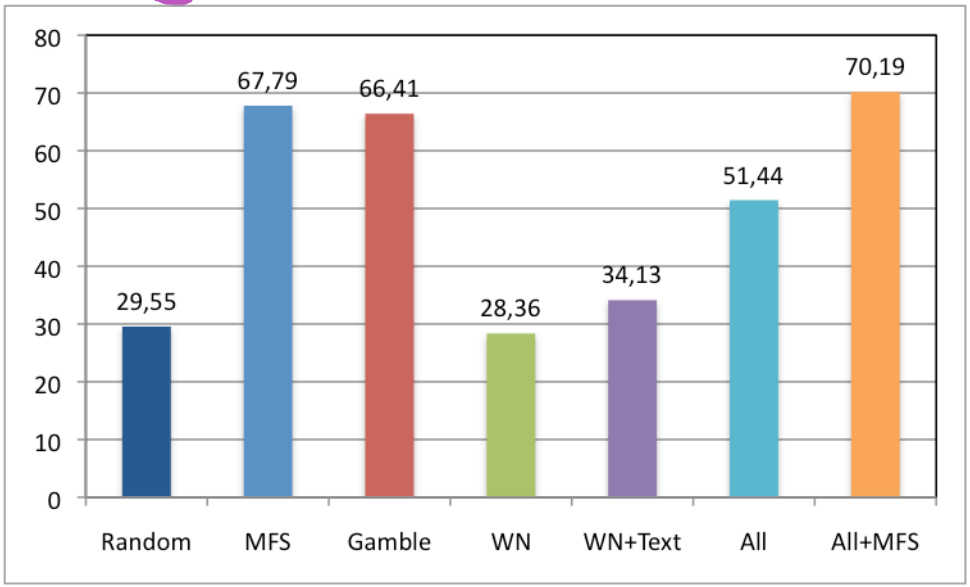
All=training from all sources

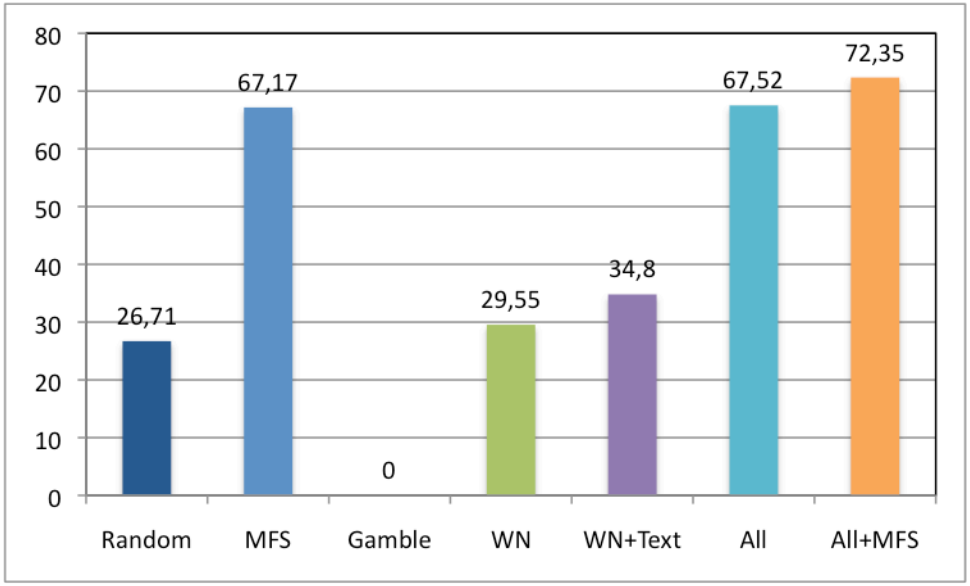
Gamble=Decadt et al 2004

(Senseval-3 SOA)

- XWN results

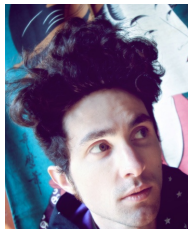
XWN = eXtended WN



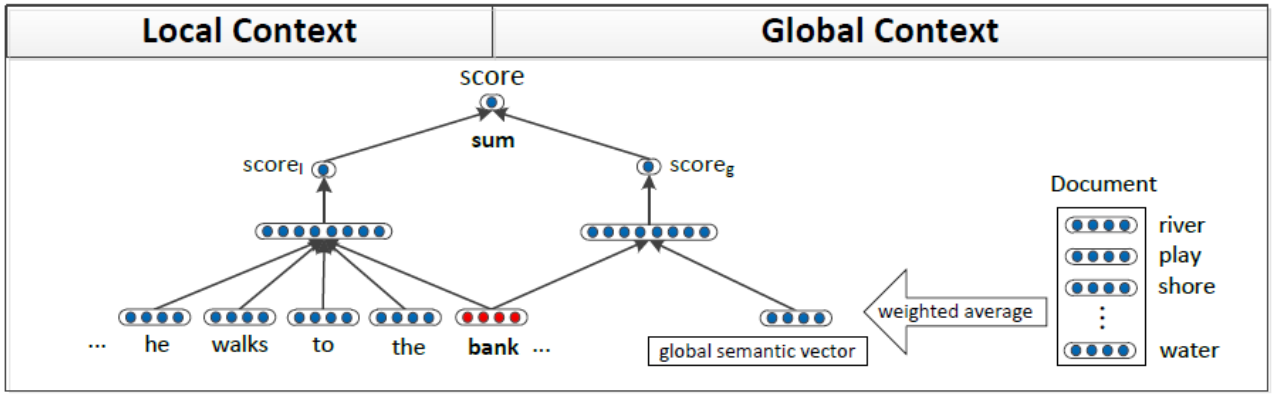


Learning Multiple Word Vectors

- Tackles problems with polysemous words

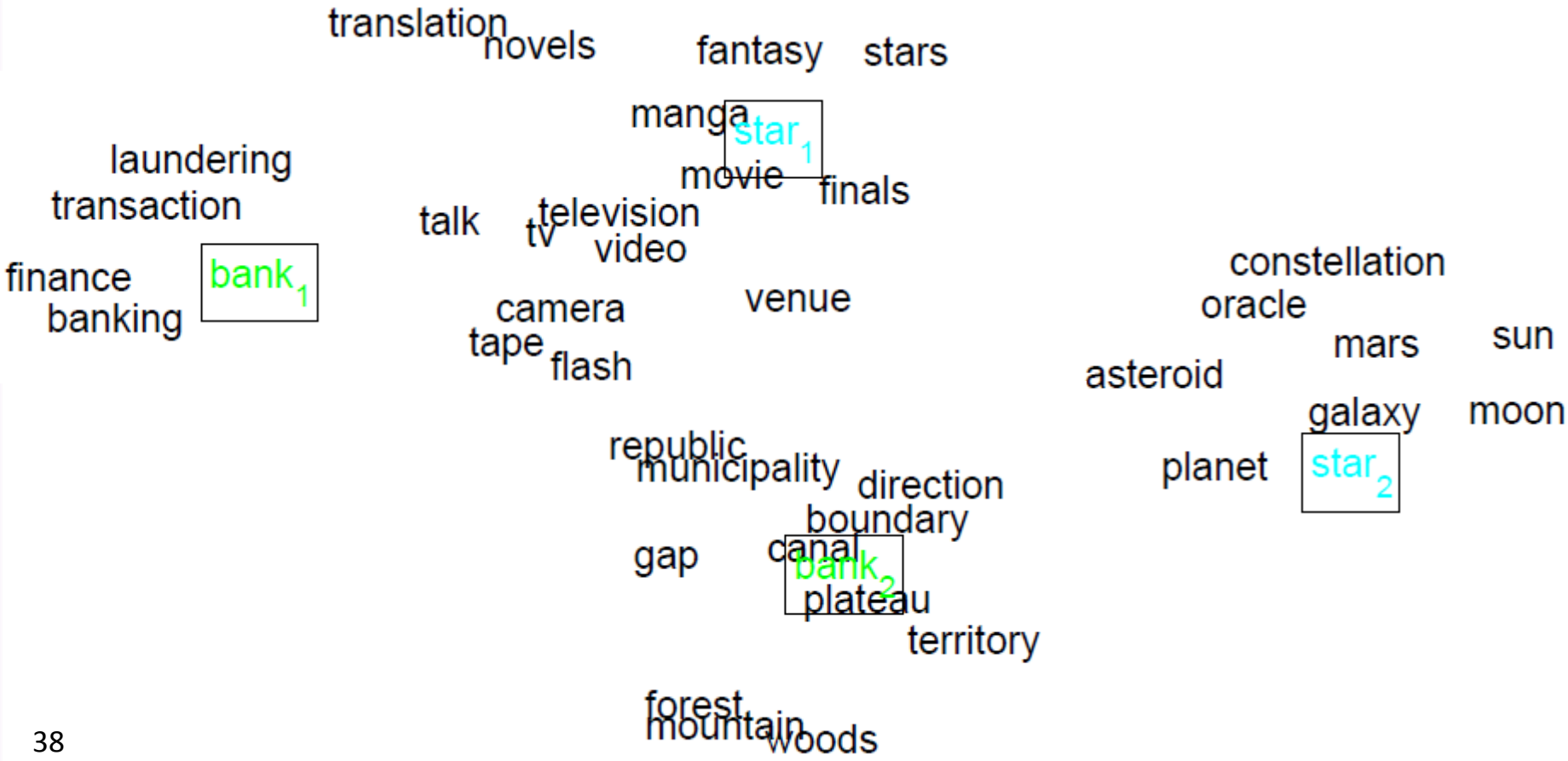


- Can be done with both standard tf-idf based methods [Reisinger and Mooney, NAACL 2010]
- Recent neural word vector model by [Huang et al. ACL 2012] learns multiple prototypes using both local and global context
- State of the art correlations with human similarity judgments

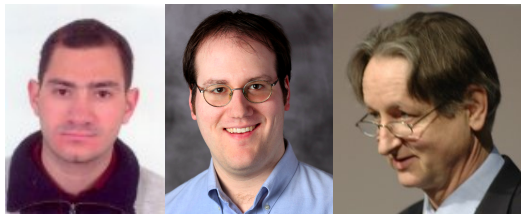


Learning Multiple Word Vectors

- Visualization of learned word vectors from Huang et al. (ACL 2012)



Phoneme-Level Acoustic Models



- [Mohamed et al, 2011, IEEE Tr.ASLP]

- Unsupervised pre-training as Deep Belief Nets (a stack of RBMs), supervised fine-tuning to predict phonemes
- Phoneme classification on TIMIT:
  - CD-HMM: 27.3% error
  - CRFs: 26.6%
  - Triphone HMMs w. BMMI: 22.7%
  - Unsupervised DBNs: 24.5%
  - Fine-tuned DBNs: 20.7%
- Improved version by Dong Yu is RELEASED IN MICROSOFT'S ASR system for Audio Video Indexing Service

Domain Adaptation for Sentiment Analysis

- [Glorot et al, ICML 2011] beats SOTA on Amazon benchmark, 25 domains
- Embeddings pre-trained in denoising auto-encoder
- Disentangling effect (features specialize to domain or sentiment)