

Neural Language Models

Sargur N. Srihari

srihari@cedar.buffalo.edu

This is part of lecture slides on [Deep Learning](http://www.cedar.buffalo.edu/~srihari/CSE676):
<http://www.cedar.buffalo.edu/~srihari/CSE676>

Topics in NLP

1. N-gram Models
2. Neural Language Models
3. High-dimensional Outputs
4. Combining Neural LMs with n-grams
5. Neural Machine Translation
6. Attention Models
7. Historical Perspective

Topics in Neural Language Models

1. Word Embedding as Distributed Representation
2. Visualizing word embeddings using t-SNE
3. Word-to-Vec training
4. Sub-word models
5. Glove: unsupervised learning of embeddings
6. Power of embeddings

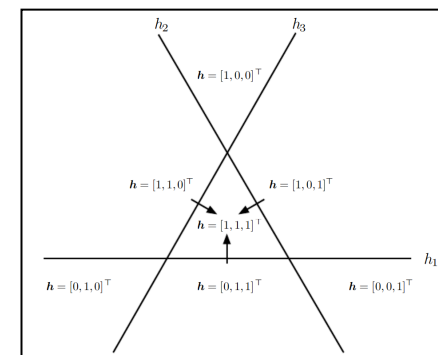
Distributed vs Symbolic representation

1. Distributed Representation

– Ex: extract n binary features (hidden units)

- A hidden unit divides R^d into pair of half-spaces
 - With n units no. of regions distinguished: $\sum_{j=0}^d \binom{n}{j} = 2^n$
 - With $O(nd)$ parameters we can represent $O(2^n)$ regions in input space
 - We can get $3^2=9$ regions with $3 \times 2=6$ parameters

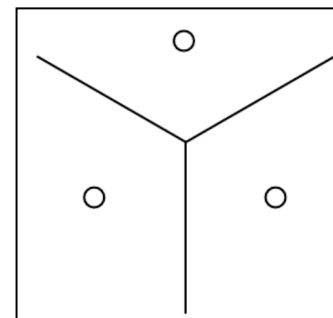
$$d=2, n=3, \mathbf{h}=[h_1, h_2, h_3]$$



2. Symbolic Representation

– For $O(2^n)$ regions, we need $O(2^n)$ examples

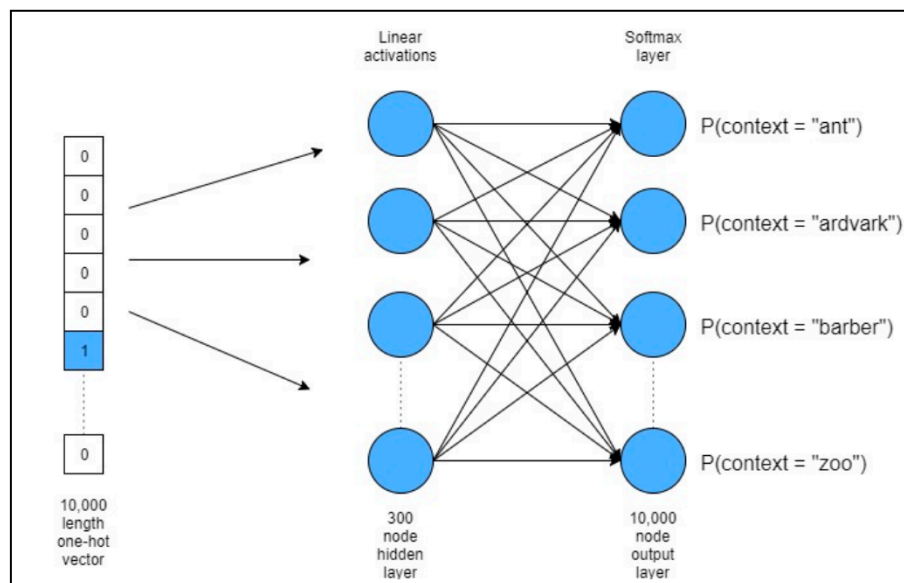
- For $3^2=9$ regions we would need 9 samples
 - For $d=2$, that would be 18 parameters
 - For one-hot representation $d=|V|$



- Fewer parameters means fewer samples needed
- One-hot vector is symbolic
- Embedded representation is a distributed

Learning Embeddings

- Starting with words in one-hot vector space
 - Whose dimensionality is vocabulary size (say, 10,000)
 - Every word is at distance $\sqrt{2}$ from every other word
- Embed them in a space of lower dimension (300)
 - Where words appearing in same context are nearby

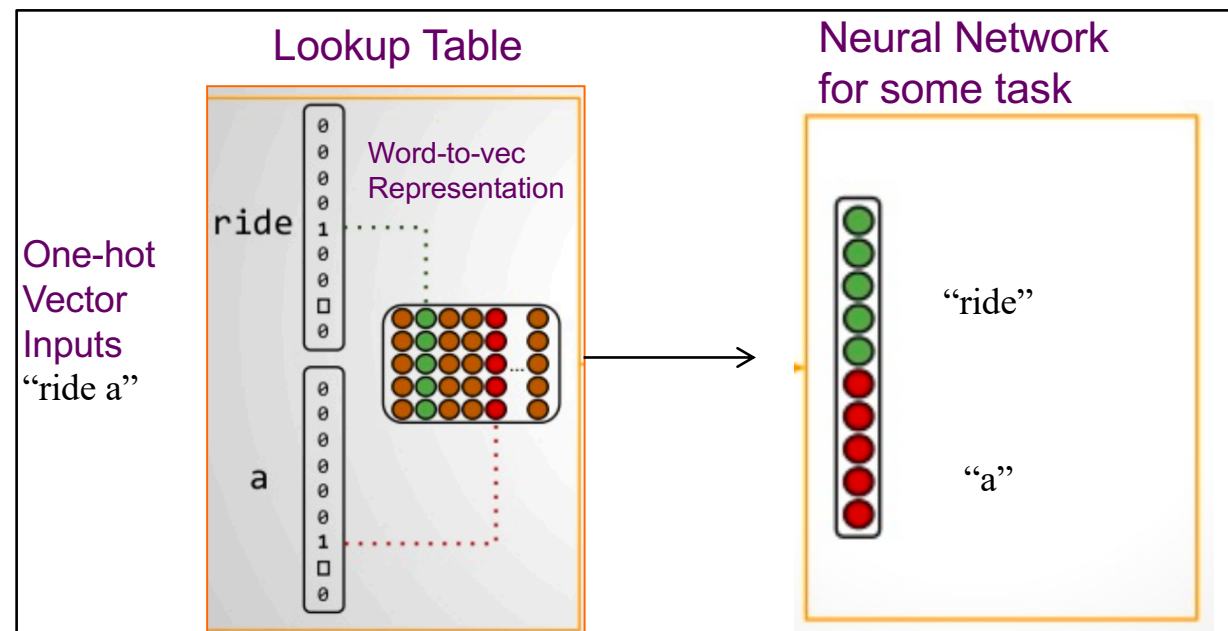


Word Embedding Values

- A word embedding W : words $\rightarrow R^n$
 - Word W is mapped into a vector of $n = 200$ to 300 dimensions
 $W(\text{'cat'})=(0.2, -0.4, 0.7, \dots)$ $W(\text{'mat'})=(0.0, 0.6, -0.1, \dots)$
- Typically a lookup-table, parameterized by a matrix θ ,
 - With a row for each word: $W_{\theta}(w_n)=\theta_n$

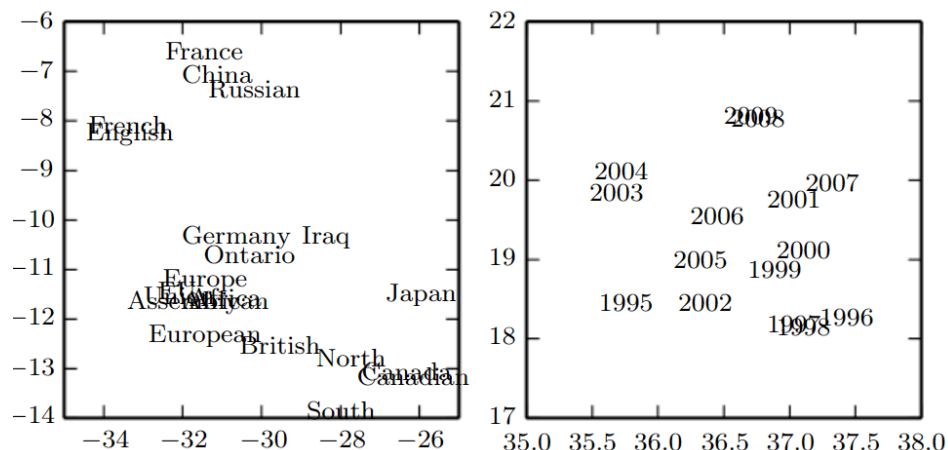
Representing a phrase

One-hot-vector
to Embedded
Representation

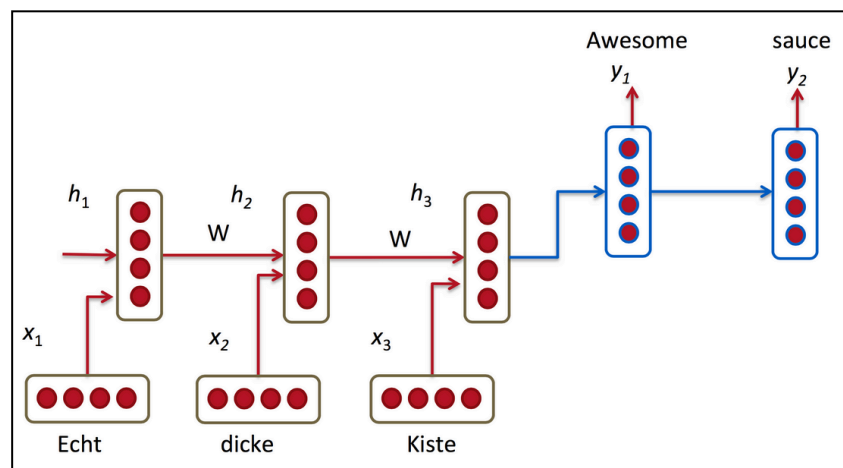


Similar words in embedding

- Word embeddings in a machine translation model
 - Zooming in on areas where related words have embeddings nearby
 - (countries, dates)



Neural Network for Machine Translation



Visualizing word embedding

- **t-SNE**: technique for visualizing high-dimensional data

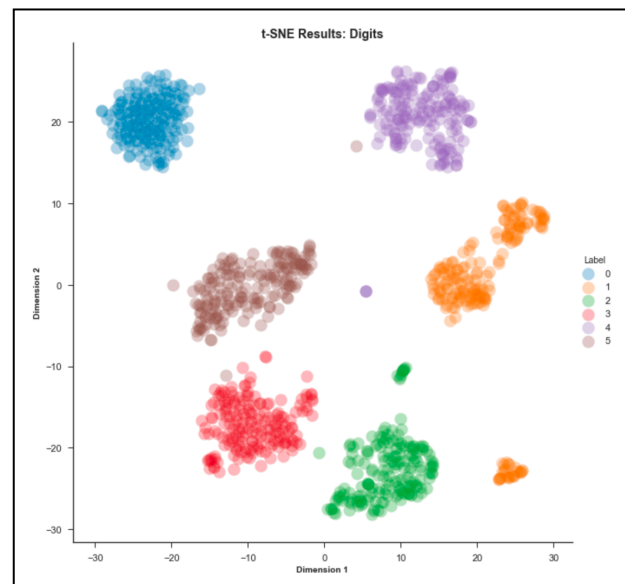
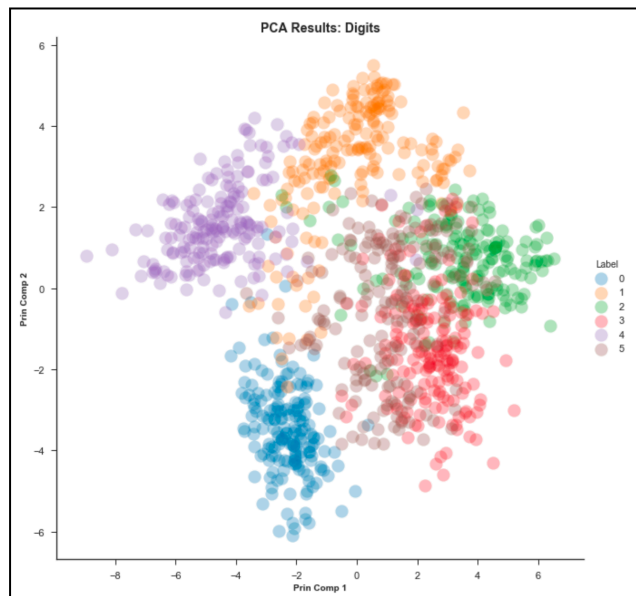


– Resulting map makes intuitive sense

- Similar words are close together

T-SNE for visualizing high-dimensional data

- t-Distributed Stochastic Neighbor Embedding
- t-SNE Differs from PCA
 - Preserves small pairwise distances (local similarities)
 - PCA preserves large pairwise distances to maximize variance



Mechanics of t-SNE

- Stage 1: Construct distribution over pairs x_i, x_j so that

- Similar objects have high probability
- Dissimilar ones low probability

$$p_{j|i} = \frac{\exp(-\|x_i - x_j\|^2 / 2\sigma_i^2)}{\sum_{k \neq i} \exp(-\|x_i - x_k\|^2 / 2\sigma_i^2)}$$

- Stage 2: Define low-dimensional distribution

$$q_{j|i} = \frac{\exp(-\|y_i - y_j\|^2)}{\sum_{k \neq i} \exp(-\|y_i - y_k\|^2)}$$

- Minimize KL-divergence between the two distributions

$$C = \sum_i KL(P_i || Q_i) = \sum_i \sum_j p_{j|i} \log \frac{p_{j|i}}{q_{j|i}}$$

P_i is conditional probability over all other data points given data-point x_i
 Q_i is conditional probability over all other map points given map point y_i

– Using gradient descent

$$\frac{\delta C}{\delta y_i} = 2 \sum_j (p_{j|i} - q_{j|i} + p_{i|j} - q_{i|j})(y_i - y_j)$$

- Choosing σ_i

– Binary search for σ_i to produce a P_i

- With a user-specified perplexity

$$\text{Perp}(P_i) = 2^{H(P_i)} \quad H(P_i) = - \sum_j p_{j|i} \log_2 p_{j|i}$$

– Perplexity is a smooth measure of effective no of neighbors

Neural Language Models

- Unlike class-based n -gram models
 - Neural Language Models are able to recognize that two words are similar
 - without losing the ability to encode each word as distinct from others

Word-to-vec:

Represent noun by co-occurrences with 25 verbs*

Semantic feature values:

“celery”

0.8368, eat
0.3461, taste
0.3153, fill
0.2430, see
0.1145, clean
0.0600, open
0.0586, smell
0.0286, touch

...

...

0.0000, drive
0.0000, wear
0.0000, lift
0.0000, break
0.0000, ride

High semantic
values

Low semantic
values

Semantic feature values:

“airplane”

0.8673, ride
0.2891, see
0.2851, say
0.1689, near
0.1228, open
0.0883, hear
0.0771, run
0.0749, lift

...

...

0.0049, smell
0.0010, wear
0.0000, taste
0.0000, rub
0.0000, manipulate

* in a trillion word text collection

Strength of NLMs

- Share statistical strength between one word (and its context) and other similar words and contexts
- Distributed representation allows model to treat words that have features in common similarly
- Overcomes curse of dimensionality for sequences
 - Handled by relating each training sentence to an exponential number of similar sentences

Importance of Word Embedding

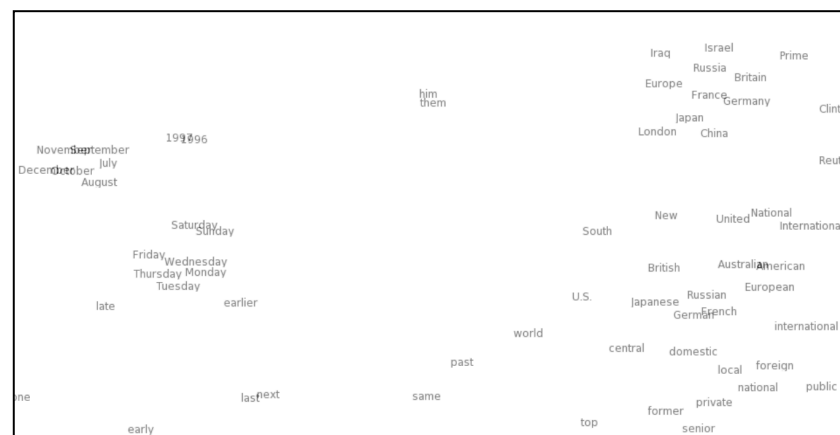
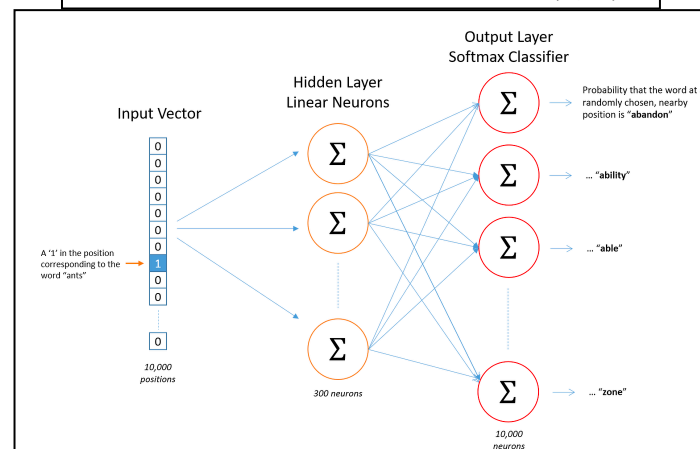
- Neural networks in other domains also define embeddings
 - E.g., convolutional neural network provides an image embedding
- Embedding in NLP is more interesting since natural language does not originally lie in a real-valued vector space

Word-to-Vec Training

- Training Data
- Word-to-vec
 - One-hot vector mapped to embedded vector of 300
- Word embedding
 - Similar words are close together

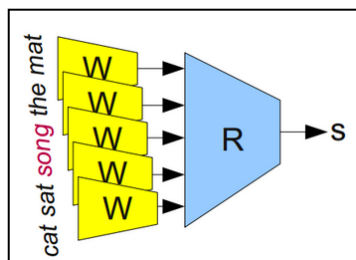
Proxy
Task

Source Text	Training Samples
The quick brown fox jumps over the lazy dog. →	(the, quick) (the, brown)
The quick brown fox jumps over the lazy dog. →	(quick, the) (quick, brown) (quick, fox)
The quick brown fox jumps over the lazy dog. →	(brown, the) (brown, quick) (brown, fox) (brown, jumps)
The quick brown fox jumps over the lazy dog. →	(fox, quick) (fox, brown) (fox, jumps) (fox, over)



Another word-to-vec trainer

- W initialized with random vectors for each word
- Learns vectors to perform some task
 - Task: tell whether 5-gram is valid or broken
 - Training data: legal 5-grams, e.g., cat sat on the mat
 - Make half of them nonsensical by switching with a random word (cat sat **song** the mat)



$$R(W(\text{cat}), W(\text{sat}), W(\text{on}), W(\text{the}), W(\text{mat})) = 1$$

$$R(W(\text{cat}), W(\text{sat}), W(\text{song}), W(\text{the}), W(\text{mat})) = 0$$

- Need to learn parameters for W and R
 - R is not as interesting as W
 - Entire point of task is to learn W

Words closest in the embedding

- Words with embeddings closest to a given word

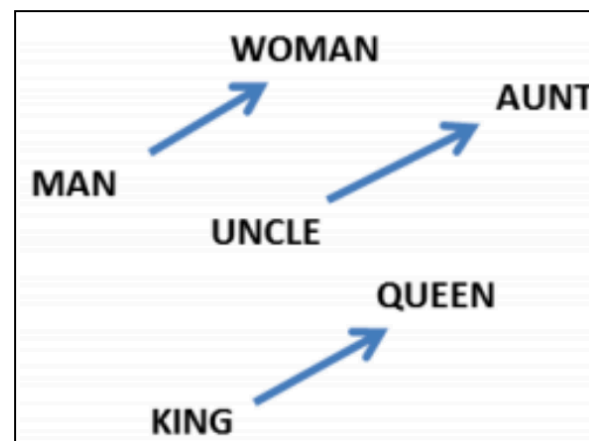
FRANCE	JESUS	XBOX	REDDISH	SCRATCHED	MEGABITS
AUSTRIA	GOD	AMIGA	GREENISH	NAILED	OCTETS
BELGIUM	SATI	PLAYSTATION	BLUISH	SMASHED	MB/S
GERMANY	CHRIST	MSX	PINKISH	PUNCHED	BIT/S
ITALY	SATAN	IPOD	PURPLISH	POPPED	BAUD
GREECE	KALI	SEGA	BROWNISH	CRIMPED	CARATS
SWEDEN	INDRA	PSNUMBER	GREYISH	SCRAPED	KBIT/S
NORWAY	VISHNU	HD	GRAYISH	SCREWED	MEGAHERTZ
EUROPE	ANANDA	DREAMCAST	WHITISH	SECTIONED	MEGAPIXELS
HUNGARY	PARVATI	GEFORCE	SILVERY	SLASHED	GBIT/S
SWITZERLAND	GRACE	CAPCOM	YELLOWISH	RIPPED	AMPERES

Power of Word Embeddings

- Similar words being close together allows us to generalize from one sentence to a class of similar sentences
- Not just word for synonym but switching a word for a word in a similar class
- E.g., wall is blue → wall is red
wall is blue → ceiling is red

Word embeddings and analogies

- Analogies between words are encoded in difference vectors between words
 - E.g., constant male-female difference vector
 - $W(\text{woman}) - W(\text{man}) \approx W(\text{aunt}) - W(\text{uncle})$
 - $W(\text{woman}) - W(\text{man}) \approx W(\text{queen}) - W(\text{king})$
- Not surprising, since
 - we write “she is the aunt” but “he is the uncle”



Word embeddings & relationship pairs

- More sophisticated relationships are encoded

Relationship	Example 1	Example 2	Example 3
France - Paris	Italy: Rome	Japan: Tokyo	Florida: Tallahassee
big - bigger	small: larger	cold: colder	quick: quicker
Miami - Florida	Baltimore: Maryland	Dallas: Texas	Kona: Hawaii
Einstein - scientist	Messi: midfielder	Mozart: violinist	Picasso: painter
Sarkozy - France	Berlusconi: Italy	Merkel: Germany	Koizumi: Japan
copper - Cu	zinc: Zn	gold: Au	uranium: plutonium
Berlusconi - Silvio	Sarkozy: Nicolas	Putin: Medvedev	Obama: Barack
Microsoft - Windows	Google: Android	IBM: Linux	Apple: iPhone
Microsoft - Ballmer	Google: Yahoo	IBM: McNealy	Apple: Jobs
Japan - sushi	Germany: bratwurst	France: tapas	USA: pizza

All these are side-effects

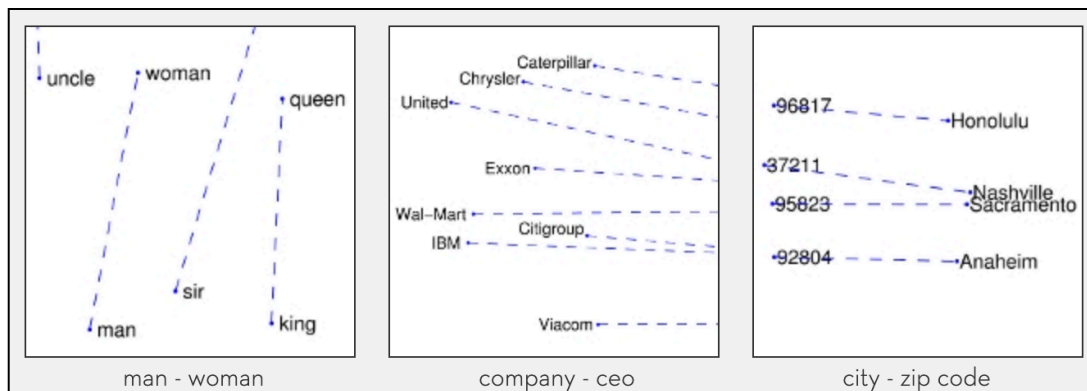
- All these properties of W are side-effects
 - We didn't try to have similar words close together
 - We didn't try to have analogies encoded with difference vectors
- All we tried to do was a simple task, whether a sentence was valid
 - These properties popped out of optimization process
- Neural networks learn better ways to represent data automatically

Sub-word Models

- Same architectures as for word level models
 - But use smaller “word pieces”: letter pairs, characters
- Word embeddings can be composed from character embeddings
 - Generates embeddings for unknown words
 - Similar spellings share similar embeddings
 - Solves OOV (out-of-vocabulary) problem
- Character-level embeddings outperformed word level embedding in English-Czech MT

Glove: Global Vectors for Word Representation

- Unsupervised learning to obtain vector representations for words.
 - Training performed on aggregated global word-word co-occurrence statistics from a corpus, and the resulting representations showcase interesting linear substructures of the word vector space.
 - Vector differences *man - woman*, *king - queen* and *brother - sister* roughly equal.



Importance of Word Embedding

- Neural networks in other domains also define embeddings
 - E.g., convolutional neural network provides an image embedding
- Embedding in NLP is more interesting since it does not originally lie in a real-valued vector space
- Using distributed representations is also used with PGM hidden variables