Computational Intelligence Laboratory

Lecture 5
Embeddings

Thomas Hofmann

ETH Zurich – cil.inf.ethz.ch

20 March 2020

Section 1

Motivation: Word Embeddings

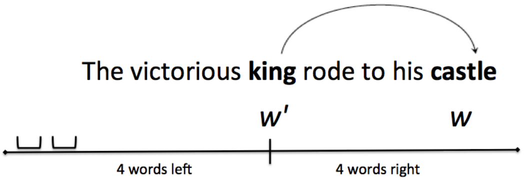
Motivation: Embeddings

- Lexical Semantics
  - natural language: atomic units of meaning are **symbols** – words or phrases
  - symbols rarely carry their meaning "on them"
  - meaning of a word: its use in language (Wittgenstein, 1953)
- Semantic Representation
  - given: examples of word uses in a corpus (word occurrences)
  - goal: learn word representations that capture word meanings
  - most basic representation: **embed symbols in vector space**
  - vector space structure (e.g. angles, distances) should relate to word meaning
  - applies more broadly to other symbols (identifiable events)

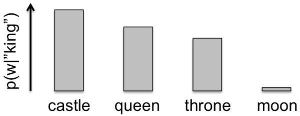
Distributional Context Models

- Predict context word given "active" word = skip-gram model

$$p_\theta(w|w') = \text{probability that } w \text{ occurs in context window of } w'$$



- Distributional semantics model = distribution of co-occurring words determines lexical semantics



Section 2

Basic Model

Context Model Likelihood

- Objective function (log-likelihood) = predictive score

$$\mathcal{L}(\theta; \mathbf{w}) = \sum_{t=1}^{T} \sum_{\triangle \in \mathcal{I}} \log p_\theta(w^{(t+\triangle)} \,|\, w^{(t)})$$

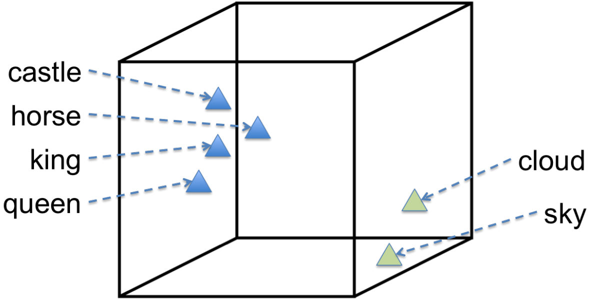
  - $\mathbf{w} = w^{(1)}, \ldots, w^{(T)}$, sequence of words (implicitly padded)
  - window of offsets $\mathcal{I} = \{-R, \ldots, -1, 1, \ldots, R\}$
  - alternatively: words within the same sentence

- Maximum likelihood estimation: $\hat{\theta} = \arg\max_\theta \mathcal{L}(\theta; \mathbf{w})$
  - prefer model that assigns high probability to observed context
  - key question: how to define an appropriate model $p_\theta(w \,|\, w')$?

Latent Vector Model: Basic Model

- Latent vector representation of words = embedding

$$w \mapsto (\mathbf{x}_w, b_w) \in \mathbb{R}^{d+1}, \quad \text{(vector + bias)}$$



Latent Vector Model: Basic Model

- Latent vector representation of words = embedding

$$w \mapsto (\mathbf{x}_w, b_w) \in \mathbb{R}^{d+1}, \quad \text{(vector + bias)}$$

- Define log-bilinear model

$$\log p_\theta(w \,|\, w') = \langle \mathbf{x}_w, \mathbf{x}_{w'} \rangle + b_w + const.$$

  - symmetric bilinear form fitted to log-probabilities
  - normalization constant (see below)

- Main effects:
  - unspecific: $b_w \uparrow \implies p_\theta(w \,|\, w') \uparrow \;\forall w'$
  - specific: $\angle(\mathbf{x}_w, \mathbf{x}_{w'}) \downarrow \implies p_\theta(w \,|\, w') \uparrow$
  - inner products: interactions; biases: marginals

Latent Vector Model: Basic Model (cont'd)

- Exponentiating $\Longrightarrow$ soft-max

$$p_\theta(w \mid w') = \frac{\exp\left[\langle \mathbf{x}_w, \mathbf{x}_{w'} \rangle + b_w\right]}{Z_\theta(w')}$$

  - partition function (normalization constant):

$$Z_\theta(w') := \sum_{v \in \mathcal{V}} \exp\left[\langle \mathbf{x}_v, \mathbf{x}_{w'} \rangle + b_v\right]$$

  - model parameters:

$$\theta = ((\mathbf{x}_w, b_w)_{w \in \mathcal{V}}) \in \mathbb{R}^{(d+1)\cdot|\mathcal{V}|}$$

Section 3

Skip-Gram Model

Latent Vector Model: Challenges

- Log-likelihood of basic model

$$\mathcal{L}(\theta; \mathbf{w}) = \sum_{t=1}^{T} \sum_{\triangle \in \mathcal{I}} \Big[$$

$$b_{w(t+\triangle)} \quad \text{ok}$$

$$+\langle \mathbf{x}_{w(t+\triangle)}, \mathbf{x}_{w(t)} \rangle \quad \text{bi-linear} \longleftarrow \#1$$

$$-\log \sum_{v \in \mathcal{V}} \exp \left[\langle \mathbf{x}_v, \mathbf{x}_{w(t)} \rangle + b_v \right] \quad \text{large cardinality} \longleftarrow \#2$$

$$\Big]$$

Modification # 1: Context Vectors

- Distinguish output vocabulary $\mathcal{V}$ and input vocabulary $\mathcal{C}$
- Introduce two different embeddings
  - $\mathbf{x}_w$: output embeddings, $w \in \mathcal{V}$
  - $\mathbf{y}_w$: input embeddings, $w \in \mathcal{C}$
- Use mixed inner products

$$\log p_\theta(w \,|\, w') = \langle \mathbf{x}_w, \mathbf{y}_{w'} \rangle + b_w$$

- Discussion
  - Pros: modelling flexibility; Cons: model dimensionality
  - simpler model $\mathbf{x}_w = \mathbf{y}_w$ for $w \in \mathcal{V} \cap \mathcal{C}$ (not commonly used)

Modification # 2: Objective

- Alternatives to maximum likelihood:
  - Contrastive divergence (word2vec, Mikolov et al. 2013)
  - Negative sampling (Mikolov et al. 2013)
  - Pointwise mutual information (Levy & Goldberg 2014)
  - Weighted squared loss (GloVe, Pennigton et al. 2013)
- Active area of research ...

Negative Sampling

- Reduce estimation to binary classification $\Longrightarrow$ noise contrastive estimation (Gutmann & Hyvärinen, 2010)
- Simplified version: negative sampling
  - $p_n(i,j)$: probability to generate negative example of word pairs (w_i, w_j) – can be defined quite arbitrary
  - observed pairs $\Longrightarrow$ positive training examples $\triangle^+$
  - pairs sampled from p_n $\Longrightarrow$ negative training examples $\triangle^-$
- Perform logistic regression, $\sigma(z) := \frac{1}{1+\exp(-z)}$, i.e. maximize

$$\mathcal{L}(\theta) = \sum_{(i,j)\in\triangle^+} \log\sigma(\langle \mathbf{x}_i, \mathbf{y}_j \rangle) + \sum_{(i,j)\in\triangle^-} \log\sigma(-\langle \mathbf{x}_i, \mathbf{y}_j \rangle)$$

Negative Sampling (cont'd)

- How to sample negative examples?
- Distribution p_n
  - re-use active words (from data) $\Longrightarrow$ defines w_i
  - sample "random" context words: $w_j \propto P(w_j)^\alpha$, e.g. $\alpha = 3/4$
  - (exponent dampens frequent words)
- How many negative samples?
  - oversample by a factor k
  - practical choices $k = 2 - 20$, smaller for larger data sets

Negative Sampling & PMI

- Bayes optimal discriminant for $\mathcal{L}$

$$h_{ij}^* = \sigma^{-1}\left(\frac{\kappa p(w_i, w_j)}{\kappa p(w_i, w_j) + (1-\kappa) p_n(w_i, w_j)}\right)$$
$$= \log \frac{p(w_i, w_j)}{p_n(w_i, w_j)} + \log \frac{\kappa}{1-\kappa}$$

  where $\kappa = 1/(k+1)$.

- For $k = 1$ (no oversampling) and $p_n(w_i, w_j) = p(w_i)p(w_j)$: **pointwise mutual information**

$$\langle \mathbf{x}_i, \mathbf{y}_j \rangle \approx \mathsf{PMI}(w_i, w_j)$$

Section 4

GloVe

Co-Occurrence Matrix

- Summarize data in co-occurrence matrix

$$\mathbf{N} = (n_{ij}) \in \mathbb{N}^{|\mathcal{V}|\cdot|\mathcal{C}|},$$

$$n_{ij} = \#\text{ occurrences of } w_i \in \mathcal{V} \text{ in context of } w_j \in \mathcal{C}$$

  - e.g. $w_i =$ "castle", $w_j =$ "king", then $n_{ij} =$ how often did word "castle" occur in a context of word "king"

- Practicalities
  - $\mathbf{N}$ can be computed in one pass over the text corpus
  - sparse matrix, most entries 0

GloVe Objective

- Weighted least squares fit of log-counts

$$\mathcal{H}(\theta; \mathbf{N}) = \sum_{i,j} f(n_{ij}) \left(\underbrace{\log n_{ij}}_{\text{target}} - \underbrace{\log \tilde{p}_\theta(w_i | w_j)}_{\text{model}} \right)^2 ,$$

with unnormalized distribution

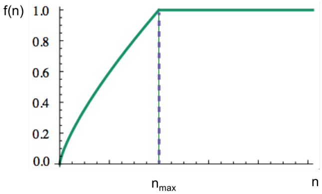
$$\tilde{p}_\theta(w_i | w_j) = \exp\left[\langle \mathbf{x}_i, \mathbf{y}_j \rangle + b_i + c_j\right]$$

and weighting function f

GloVe Weighting

- Weighting function

$$f(n) = \min\left\{1, \left(\frac{n}{n_{\max}}\right)^{\alpha}\right\}, \quad \alpha \in (0;1] \text{ e.g. } \alpha = \frac{3}{4}$$



- Motivation
  - cut-off at $n_{\max}$: limit influence of large counts (frequent words)
  - $f(n) \to 0$ for $n \to 0$: as small counts are (very!) noisy
  - specific form with exponent α: heuristically chosen

Normalized vs. Unnormalized Models

- Normalized model
  - requires computation of partition function
  - general case over state space Ω

$$p(\omega) = \frac{\exp\left[h(\omega)\right]}{\sum_{\omega' \in \Omega} \exp\left[h(\omega')\right]}$$

  - log-likelihood

$$\mathcal{L} = \sum_t \log p(\omega_t)$$

  - $h(\omega) \uparrow \implies p(\omega) \uparrow \implies \log p(\omega) \uparrow \implies \mathcal{L} \uparrow$
    (higher prob. better)
  - counterbalanced by normalization: cannot be large everywhere

Normalized vs. Unnormalized Models (cont'd)

- Unnormalized model
  - no computation of partition function

$$\tilde{p}(\omega) = \exp\left[h(\omega)\right]$$

  - use two-sided loss function
  - GloVe: quadratic loss with log-counts as targets
  - $\tilde{p}(\omega)$ should neither be too large nor too small

Matrix Decomposition

- Absorb bias into vectors (wlog)

$$x_{w,d-1} = 1, \;\; x_{w,d} = b_w \quad \text{and} \quad y_{w,d-1} = c_w, \;\; y_{w,d} = 1.$$

- Define

$$\mathbf{M} = (m_{ij}), \quad m_{ij} := \log n_{ij}$$
$$\mathbf{X} := \left[\mathbf{x}_{w_1} \cdots \mathbf{x}_{w_{|\mathcal{V}|}}\right], \quad \mathbf{Y} := \left[\mathbf{y}_{w_1} \cdots \mathbf{y}_{w_{|\mathcal{C}|}}\right]$$

Matrix Decomposition (cont'd)

- GloVe with $f := 1$ solves a *matrix factorization* problem

$$\min_{\mathbf{X},\mathbf{Y}} \|\mathbf{M} - \mathbf{X}^\top \mathbf{Y}\|_F^2$$

- GloVe: separate weight for each entry (data-dependent) $\implies$ need to go beyond SVD
  - **Exercise**: GloVe with $f(n_{ij}) := \begin{cases} 1 & \text{if } n_{ij} > 0, \\ 0 & \text{otherwise.} \end{cases}$

    solves a *matrix completion* problem

$$\min_{\mathbf{X},\mathbf{Y}} \sum_{ij\,:\,n_{ij}>0} \left(m_{ij} - (\mathbf{X}^\top \mathbf{Y})_{ij}\right)^2$$

GloVe Optimization (no!)

- Non-convex problem: hard to find global minimum
- Gradient descent (aka steepest descent)

$$\theta^{\text{new}} \leftarrow \theta^{\text{old}} - \eta \nabla_\theta \mathcal{H}(\theta; \mathbf{N}), \quad \eta > 0 \text{ (step size)}$$

  - $\theta = ((\mathbf{x}_w)_{w \in \mathcal{V}}, (\mathbf{y}_w)_{w \in \mathcal{C}})$, embeddings = parameters
  - full gradient: often too expensive to compute 🚫

GloVe Optimization (yes!)

- Use stochastic optimization to find local minimum
- Stochastic gradient descent (SGD):
  - sample (i, j) such that $n_{ij} > 0$ uniformly at random
  - perform "cheap" update (single entry and sparse)

$$\mathbf{x}_i^{\text{new}} \leftarrow \mathbf{x}_i + 2\eta f(n_{ij}) \left(\log n_{ij} - \langle \mathbf{x}_i, \mathbf{y}_j \rangle\right) \mathbf{y}_j$$

$$\mathbf{y}_j^{\text{new}} \leftarrow \mathbf{y}_j + 2\eta f(n_{ij}) \left(\log n_{ij} - \langle \mathbf{x}_i, \mathbf{y}_j \rangle\right) \mathbf{x}_i$$

Word Similarity

Nearest words to frog:

1. frogs
2. toad
3. litoria
4. leptodactylidae
5. rana
6. lizard
7. eleutherodactylus

litoria

leptodactylidae

rana

eleutherodactylus

Affine Embedding Structure

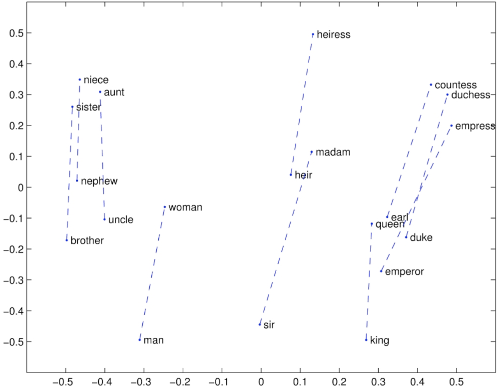
- Word vector analogies

a:b :: c:?

man:woman :: king:?

$$d = \arg\max_i \frac{(x_b - x_a + x_c)^T x_i}{||x_b - x_a + x_c||}$$

- $2d$-projection



Word Embeddings: Discussion

- Word embeddings can model analogies and relatedness (see previous examples)
  - ... but: antonyms ("cheap" vs. "expensive") are usually not well captured
- Word embeddings $\Longrightarrow$ sentence or document embeddings
  - simple: aggregation
  - sophisticated: convolutional or recurrent neural networks
  - use cases: language models, sentiment analysis, text categorization, machine translation, etc.
  - ... more about this in our "Natural Language Processing" class