Deep networks for information retrieval

Martin Szummer
Microsoft Research, Cambridge UK

In collaboration with:
Marc’Aurelio Ranzato, New York University
Current Representations in retrieval systems

- **Query alteration**
  - spelling correction, stemming, query expansions (synonyms, acronyms)

- **Inverted index lookup**
  - Given a word, find documents containing them (and positions within documents) of these words
  - AND-set: documents that contain ALL query words

- **Forward index lookup**
  - Document features: PageRank, spam score, age, #clicks

- **Ranking**
  - Order the AND-set according to quality of match between query - document feature match
  - Match measures: TF-IDF, BM25
Better Representations

Goal: Capture document or query topics to handle synonymy and semantics, while remaining

- **Compact**
  - Forward index is stored in RAM
  - 40 billion index size: each representation bit costs 5 Gb of RAM

OR:

- **Sparse**
  - Can then be fit in inverted index
  - Example: document represented by its words

Can deep networks fit the bill?
Distributed Representations in information retrieval

- Exponential Family Harmoniums [Welling 2004]
- Rate Adapting Poisson Model [Gehler et al 2006]

Shallow

- Deep Belief Nets [several works by people here]
  - Semantic Hashing [Salakhutdinov & Hinton 2007]
    Binary code, achieved by adding noise during training

Deep
Computational Efficiency

- Neural networks computational cost for train and test:
  - linear in number of layers (depth)
  - quadratic in #units in adjacent layers (width)

- Deep & narrow often cheaper than shallow & wide
Exploit both labeled and unlabeled documents

- Unsupervised pretraining, then supervised finetuning
  - only unsupervised
  - only supervised

but for a given task, how do we ensure pretraining gets us to the right region in space?

Inject label information early:
- Semi-supervised training of the bottleneck layer
- Semi-supervised training of all layers
Outline

- Learning Representations of Text Documents
- Model and Learning Algorithm
- Experiments
  - Visualization
  - Classification
  - Retrieval
Our model: Deep Semi-Supervised Encoder

Calculate the representation by feeding input through a stack of encoders

Input \( x \)  
\[ Z_i \]

Encoder \( (W_{i-1}, b_{i-1}) \)

Code \( Z_i \)

Encoder \( (W_i, b_i) \)

Code \( Z_{i+1} \)

Encoder \( (W_{i+1}, b_{i+1}) \)

Code \( Z_{i+2} \)

Representation used for classification / retrieval

Sparse input, so cheap
Semi-supervised Greedy Learning

Couple each encoder with a decoder and a classifier
Learn layer by layer

GREEDY
No fine tuning from deep layers back to inputs

CHEAP
max likelihood Stochastic gradient descent
Model: 1st stage

- Model the input count vector with a conditional Poisson distrib.

  \[ \text{Decoder: } x \sim \text{Poiss}(\lambda), \quad \lambda = \beta \exp(W_D z + b_D) \]

- The encoder and the decoder mirror each other

  \[ \text{Encoder: } z = \text{logistic} \left( W_E \log(x + 1) + b_E \right) \]
Model: 1st stage

- Model the input count vector with a conditional Poisson distrib.

  **Decoder**: \( x \sim Poiss(\lambda), \quad \lambda = \beta \exp(W_Dz + b_D) \)

- The encoder and the decoder mirror each other

  **Encoder**: \( z = \text{logistic} \left( W_E \log(x+1) + b_E \right) \)

- Objective: reconstruct the input AND predict the label (if available)

  \[
  L = E_R + \alpha_C E_C
  \]
Model: higher stages

- Model the input vector with a conditional Gaussian distribution
  \[ x \sim N(W_D Z + b_D, \sigma) \]

- The encoder and the decoder mirror each other
  \[ Z = \text{logistic} \ (W_E X + b_E) \]
Model: higher stages

- Model the input vector with a conditional Gaussian distribution
  \[ x \sim N(W_D Z + b_D, \sigma) \]

- The encoder and the decoder mirror each other
  \[ Z = \text{logistic} \ (W_E X + b_E) \]

- The code has to be able to reconstruct the input as well as to predict the label, if available.
  \[ L = E_R + \alpha_c E_C \]

- Parameter learning: min L w.r.t. the parameters by stochastic gradient descent
Visualization of codes on Ohsumed corpus

4 hidden layers
30689 – 100 – 10 – 5 – 2
Classification with few labeled documents

- 20 newsgroups data
- Architecture
- 2000-200-100-50-20
- Learned features are better than TF-IDF
- Smaller codes provide more regularization
- Can re-use the top layer classifier instead of training an SVM
Learning with a semi-supervised objective is better than using a strictly supervised or a strictly unsupervised one. $L = E_R + \alpha C E_C$

Learning our model is more computationally efficient and gives better P/R than RAP.

Semi-supervised objective & learning are better!
Linear VS Nonlinear VS Deep nonlinear

The deep nonlinear model greatly outperforms a (shallow) linear model.

A nonlinear model is better than a linear one!
A deep model is better than a shallow model!
Summary

- Efficient inference
- Efficient semi-supervised learning
- **Compact** and informative features
  - The **deep** architecture seems to produce more informative features than the shallow one
- Can be integrated in a larger system whose parameters are updated by gradient descent (e.g. a ranker)
Perspectives

- Beyond bag of words
  - Proximity models
  - Language models
  - Linguistic information: Part of speech, grammar, clicks

- Binary representations

- Sparse codes: could be used in the inverted index

Thank you & Happy Birthday!