Semi-supervised Learning of Compact Document Representations with Deep Networks

Marc’Aurelio Ranzato  
*New York University*  
*NY, USA*

Martin Szummer  
*Microsoft Research*  
*Cambridge, UK*
Synonymous queries give different results

Query: learning from partially labeled data

Query: semi-supervised learning
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
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
  - Semantic Hashing [Salakhutdinov & Hinton 2007]
    Binary code, achieved by adding noise during training
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 \( x \)  
Bag of words

Encoder

Code 1

Encoder

Code 2

Encoder

Code 3

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**

[Bengio, Lamblin et al NIPS 06]

**GREEDY**

No fine tuning from deep layers back to inputs

**CHEAP**

max likelihood

Stochastic gradient descent
**Semi-supervised Greedy Learning**

Couple each encoder with a **decoder** and a **classifier**

Learn **layer by layer**

[Bengio, Lamblin et al NIPS 06]

<table>
<thead>
<tr>
<th>Semi-supervised Deep Auto Encoder</th>
<th>Deep Belief Net</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Training</strong></td>
<td></td>
</tr>
<tr>
<td>semi-supervised training layer-by-layer</td>
<td>1) unsupervised pre-training layer-by-layer</td>
</tr>
<tr>
<td>no fine-tuning of whole net</td>
<td>2) supervised fine-tune whole net</td>
</tr>
<tr>
<td><strong>Training objectives</strong></td>
<td></td>
</tr>
<tr>
<td>single objective: combine reconstruction and classification likelihoods</td>
<td>1) pretraining: model P(x)</td>
</tr>
<tr>
<td>deterministic (backprop), 4 epochs</td>
<td>2) fine-tuning: model P(y</td>
</tr>
<tr>
<td></td>
<td>1) pretraining: sampling-based</td>
</tr>
<tr>
<td></td>
<td>2) fine-tuning: backprop</td>
</tr>
<tr>
<td></td>
<td>&gt;&gt; 4 epochs</td>
</tr>
</tbody>
</table>
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

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

- Model the input count vector with a conditional Poisson distrib.
  
  **Decoder**: $x \sim \text{Poiss}(\lambda), \quad \lambda = \beta \exp(W_D z + b_D)$

- The encoder and the decoder mirror each other
  
  **Encoder**: $z = \text{logistic}(W_E \log(x+1) + b_E)$

- 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

Neoplasms
Parasitic
Musculoskeletal
Digestive System
Bacterial Infections and Mycoses

Ranzato & Szummer
Word neighbors in code space

Neighboring word stems to a given word in the 7-dimensional feature space to which documents of Reuters are mapped after learning. (2000 – 200 – 100 – 7)

<table>
<thead>
<tr>
<th>Word Stems</th>
<th>Neighboring Word Stems</th>
</tr>
</thead>
<tbody>
<tr>
<td>livestock</td>
<td>beef, meat, pork, cattle</td>
</tr>
<tr>
<td>port</td>
<td>ship, vessel, freight</td>
</tr>
<tr>
<td>plantat</td>
<td>coffee, cocoa, rubber, palm</td>
</tr>
<tr>
<td>barrel</td>
<td>oil, crude, opec, refineri</td>
</tr>
<tr>
<td>lend</td>
<td>rate, debt, bond, downgrad</td>
</tr>
</tbody>
</table>
Classification of partially labeled documents

Gaussian SVM is applied to all representations (but we could reuse the top-level classifier instead of training SVM)

Labelled samples used for:
1) training SVM
2) in semi-supervised representations
Classification of partially labeled documents

- 20 newsgroups data
- Gaussian SVM is applied to all representations (but we could reuse the top-level classifier instead of training SVM)

Labelled samples used for:
1) training SVM
2) in semi-supervised representations
Classification of partially labeled documents

Gaussian SVM is applied to all representations (but we could reuse the top-level classifier instead of training SVM).

Labelled samples used for:
1) training SVM
2) in semi-supervised representations

Ranzato & Szummer
Deep vs Linear (LSI & TF-IDF)

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

LSI is worse for retrieval than tf-idf

A deep nonlinear model is better than a linear one!
Deep vs Shallow

Deep: 4 layers
Shallow: 1 layer

Reuters-21578
A deep model is better than a shallow model!
Deep vs Shallow

A deep model is better than a shallow model!
Deep vs Shallow

Deep: 4 layers
Shallow: 1 layer

tf-idf (2000)
shallow (40)
shallow (10)
deep (3)
shallow (3)
deep (10)
deep (40)

Reuters-21578

Deep vs Shallow
Deep vs DBN vs SESM

![Graph showing precision-recall curves for deep (7), tf-idf (2000), deep (7): 2000-200-100-7, and Reuters-21578.]
Deep vs DBN vs SESM

![Graph showing precision-recall curves for different models including deep (7), DBN pre-trained (20), and tf-idf (2000).]
Deep vs DBN vs SESM

- DBN pre-trained (20)
- DBN fine-tuned (20)
- Deep (7)
- tf-idf (2000)

**Graph Details:**
- **Y-axis (PRECISION):** 0.2 to 0.7
- **X-axis (RECALL):** $10^{-3}$ to $10^{0}$
Deep vs DBN vs SESM

![Graph comparing recall and precision for different models]

- Deep (20)
- DBN pre-trained (20)
- DBN fine-tuned (20)
- Deep (7)
- TF-IDF (2000)

Legend:
- Deep: 2000-200-100-20
- DBN: 2000-200-100-20
Deep vs DBN vs SESM

![Precision vs Recall Graph]

- Deep (7)
- DBN pre-trained (20)
- DBN fine-tuned (20)
- Deep (20)
- Binary (1000)
- TF-IDF (2000)

Parameters:
- deep (20) & DBN: 2000-200-100-20
Vocabulary size

Ranzato & Szummer

Large vocabulary size gives better performance!
Summary

- **Deep Semi-supervised auto-encoders**
  - Efficient inference
  - Efficient semi-supervised learning
  - Compact and informative features

- Semi-supervised vs Unsupervised
  - Supervision helps

- Deep vs shallow
  - Deep is needed to create very compact representations

- Autoencoders can give competitive accuracy to Deep Belief Nets. Autoencoders possibly train faster

- 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!