COMPARATIVE DEEP LEARNING FOR CONTENT-BASED MEDICAL IMAGE RETRIEVAL

ADITYA SRIRAM

DECEMBER 1st, 2016
PAPER SYNOPSIS

• BIG Data supported Diagnosis
  • Content-Based Image Retrieval

• Exploring Deep Learning architectures for CBIR using IRMA dataset, composed of x-ray modalities
  • Autoencoder
  • Convolutional Autoencoder

• Create a compressive algorithm that exposes salient features of each train and test image
  • Compare these features using KD-Tree, sort by closest neighbors
  • Get IRMA Error for validation

• Purpose is to provide specialists with the closest match in the database, and supporting them to make a more informed decision
Exploiting BIG Data

- Number of images **doubles** every 5 years
- More than 80% is unstructured
- No tagging (in spite of DICOM)
  - ROI will come
- Veracity particularly significant
  - Artifacts in MRI, X-ray etc.
  - Speckle noise in Ultrasound
Medical Images

- 2 Trillion images per year
- That’s approx. 450 Exabytes
- The past 10 years: 4.5 Zettabytes
- Breast Ultrasound imaging in North-America: **100+ Petabytes**
  
  [Detect breast malignancies for a better diagnosis according to BI-RADS]

Compare: Wikipedia (10 GB), Web (1 PB), LHC (15 PB)
Human Genomics (7000 PB)
CONTENT BASED IMAGE RETRIEVAL

- It is an image search technology
- Quantifying low-level image features to represent the high-level semantic contents depicted in the images
- You want to learn the content of the image

Step 1: Query Image
Step 2: Extract Features
Step 3: Similarity Matching and Feedback
Step 4: Retrieve Similar Images
Query/Test Image

CBIR Map

Extract Primitive Features

Similarity Measure

Matched Result

Relevance Feedback Algorithm

Extracted Features

BIG Image Trained Data
CBIR Benefits

- Provide specialists support for making a more informed decision
- Early diagnostics
- Personalized medicine
IRMA Dataset

• Benchmarking dataset for CBIR

• Stands for **Image Retrieval in Medical Applications**

• Developed at **AACHEN University of Technology**, Germany
  • Dept. of Diagnostic Radiology, Medical Informatics, Division of Medical Image Processing and Chair of Computer Science (VI)

<table>
<thead>
<tr>
<th>Test Images</th>
<th>Train Images</th>
</tr>
</thead>
<tbody>
<tr>
<td>12,677 (with labels)</td>
<td>1,733 without classification</td>
</tr>
</tbody>
</table>
IRMA Code

TTTTT-DDDD-AAAA-BBBB

Image Modality  Body Orientation  Body Region  Biological System

IRMA Error: 0.03653

<table>
<thead>
<tr>
<th>Worst Score</th>
<th>Best Score</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>
Data-set Challenges

- Very Challenging data-set
- Imbalance of categorical distributions
- Size variations among images
- Variations in brightness, scale, and presence of unrelated landmarks within images
Imbalance Categorical Distribution
Noise and Size Variation

[Images of X-rays of various body parts]
Illumination, the Devil
DEEP LEARNING

WE NEED TO GO DEEPER
Let’s Talk **Deep Learning**

- “Rebranding” of Neural Networks from the 70s
- What has changed?
  - Multiple hidden layers
  - GPU goodness!
- Use Deep Learning to model **high level abstractions** of data
  - Want to generalize data!
- 2 Networks of concern
  - *Autoencoder*
  - *Convolutional Autoencoder*
- Experimental computation:
  - 1, 3, 5 Hidden Layers
  - 32x32, 64x64, and 128x128 images
Autoencoder

• Invented by Geoffrey Hinton of University of Toronto
• **Unsupervised** learning model
• Dimensionality Reduction
• To extract features by compressing data
• **Deepest layer** provides the feature vector required for retrieval
• Only needs to be trained once, model is reproducible

**Encoder:** $h(t) = f_\theta(x^{(t)}), \{x^{(1)}, \ldots, x^{(T)}\}$, where $h$ is **feature vector** or **representation**

**Decoder:** maps from feature space back into input space, producing reconstructed input back.
Autoencoder

Input

Encoding DBN

Decoding DBN

Compressed Feature Vector

Output
Convolutional Autoencoder

- Train CNN on training image labels (already done)
- Use any of the Dense layers or even Conv layers as features for retrieval
- In this project, instead of training on labels we reconstruct the input image
- Technique is called Convolution Autoencoder (CAE)
- Now, we can use any Conv (preferably deep) layers as our features.

Benefits:
- Smooth
- Sharpen
- Intensify
- Enhance
- Various other operations
Convolutional Autoencoder

![Convolutional Autoencoder Diagram]
Deep Learning

Retrieval Procedure

Step 1
Encode all training images as vectors

Step 2
Index trained images

Step 3
For given image, find encoded value (using KD-Tree)

Step 4
Retrieve Closest match based on Euclidean distance
Deep Learning

Optimal Network

- **Over-fitting**: Network performs bad on unseen test data. Use of regularization techniques could help!
  - Used *Gaussian Noise* and *Dropout*

- **Train-Validation split**: Need to take care of extreme categorical imbalance in the training data
  - Currently set at **20% Validation split**

- **Early stopping**: How many epochs should I train? Perhaps, validation accuracy can give us some hints?

- **Optimizers**: Plenty are available, mprop, adagrad, SGD, mini-SGD, ADAM.
# Autoencoder

## Experimental Results

<table>
<thead>
<tr>
<th>Architecture Breakdown</th>
<th>32 x 32</th>
<th>64 x 64</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>1024 &gt; 256</td>
<td>1024 &gt; 256 &gt; 64</td>
</tr>
<tr>
<td>IRMA Error</td>
<td>375</td>
<td>409</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Architecture Breakdown</th>
<th>128 x 128</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>16384 &gt; 4096</td>
</tr>
<tr>
<td>IRMA Error</td>
<td>395</td>
</tr>
</tbody>
</table>
Autoencoder
Experimental Results

- Acquired 78% accuracy
- **Input**: Pre-processed 32 x 32 image
- **Deepest Layer**: 16 x 16
- 1 Hidden Layer: \(1024 > 256\)

<table>
<thead>
<tr>
<th>Architecture Breakdown</th>
<th>IRMA Error</th>
</tr>
</thead>
<tbody>
<tr>
<td>32 x 32</td>
<td></td>
</tr>
<tr>
<td>(1024 &gt; 256)</td>
<td>375</td>
</tr>
<tr>
<td>(1024 &gt; 256 &gt; 64)</td>
<td>409</td>
</tr>
<tr>
<td>(1024 &gt; 256 &gt; 64 &gt; 16)</td>
<td>481</td>
</tr>
</tbody>
</table>
## Convolutional Autoencoder

### Experimental Results

<table>
<thead>
<tr>
<th>Architecture Breakdown</th>
<th>32 x 32</th>
<th>64 x 64</th>
<th>128 x 128</th>
</tr>
</thead>
<tbody>
<tr>
<td>IRMA Error</td>
<td>414</td>
<td>435</td>
<td>436</td>
</tr>
<tr>
<td>32 &gt; 16</td>
<td>32 &gt; 16 &gt; 8</td>
<td>32 &gt; 16 &gt; 8 &gt; 4</td>
<td>128 &gt; 64</td>
</tr>
<tr>
<td>408</td>
<td>411</td>
<td>431</td>
<td>463</td>
</tr>
<tr>
<td>411</td>
<td>MemoryError</td>
<td>436</td>
<td>463</td>
</tr>
<tr>
<td>8 &gt; 4</td>
<td>64 &gt; 32 &gt; 16 &gt; 8</td>
<td>128 &gt; 64 &gt; 32 &gt; 16</td>
<td>388</td>
</tr>
</tbody>
</table>
Convolutional Autoencoder

Experimental Results

- Acquired 77% accuracy
- **Input:** Pre-processed 64 x 64 image
- **Deepest Layer:** 8 x 8
- **5 Hidden Layer:** $64 > 32 > 16 > 8$

<table>
<thead>
<tr>
<th>Architecture Breakdown</th>
<th>64 x 64</th>
<th>64 x 64</th>
<th>64 x 64</th>
</tr>
</thead>
<tbody>
<tr>
<td>IRMA Error</td>
<td>435</td>
<td>431</td>
<td><strong>388</strong></td>
</tr>
</tbody>
</table>

![Model Loss Graph](image)
## Literature Comparison

<table>
<thead>
<tr>
<th>Method</th>
<th>Error for IRMA</th>
</tr>
</thead>
<tbody>
<tr>
<td>Camlica et al. [1]</td>
<td>146.55</td>
</tr>
<tr>
<td>TAUbiomed 95 9 1246120389711 [2]</td>
<td>169.50</td>
</tr>
<tr>
<td>Idiap 3 9 1245417716666</td>
<td>178.93</td>
</tr>
<tr>
<td>Idiap 3 9 1245417671272</td>
<td>227.19</td>
</tr>
<tr>
<td>FEITIJS 96 9 1245937057229</td>
<td>242.46</td>
</tr>
<tr>
<td>VPA SabanciUniv 63 9 1245419336923</td>
<td>261.16</td>
</tr>
<tr>
<td>MedGIFT 77 9 1245961041705</td>
<td>317.53</td>
</tr>
<tr>
<td>VPA SabanciUniv 63 9 1246033855761</td>
<td>320.61</td>
</tr>
<tr>
<td>IRMA</td>
<td>359.29</td>
</tr>
<tr>
<td>MedGIFT 77 9 1246044416990</td>
<td>420.91</td>
</tr>
<tr>
<td>VPA SabanciUniv 63 9 1245936277557</td>
<td>574.00</td>
</tr>
<tr>
<td>DEU 97 9 1246226037987</td>
<td>642.50</td>
</tr>
<tr>
<td>DEU 97 9 1245952673253</td>
<td>710.10</td>
</tr>
</tbody>
</table>
Main Challenge

- Challenge with imbalance dataset
  - Generalizing becomes difficult with a bias dataset
- High-dimensionality is resource heavy
- Blurred reconstructions
- Trail-end-Error for best network
Graphical User Interface

CBIR for X-ray Images

Train Network
- Training Directory: [input]
- Testing Directory: [input]
- Train Method: Autoencoder
- No. of Hidden Layers: [input]
- Optimizer: ADAM
- Input Dimension: 32 x 32
- Training Path
- Testing Path
- Change

Index

Load Trained Model
- Load Encoded Model
- Load Full Model
- Train Network

Search Similarity
- Input Image: [input]
- Browse Image
- Change

Best Match

Query Image

Program Status

Developed By: Aditya Sriram
CONCLUSION

Both Deep Learning algorithm, Autoencoder and Convolutional Autoencoder, compress images to extract salient features. Use these features to compare train and test dataset using KD-Tree (which includes Euclidean Distance)

Best Result for an Autoencoder is 375

Created a GUI that shows End-to-End training and retrieval

The features are validated using IRMA Benchmark data-set. Provides support for specialists to have a more informed decision
Future Work

- Use CNN as a comparative study, should reduce error
- Use smarter approaches to pre-process images
- Normalization vs. Raw data as input
- Balance the dataset by increasing those categories that are low
REFERENCES


REFERENCES (cont’d)


Discussion

- Does Autoencoder extract features or only compress data?
- Autoencoders are not mathematically proven, how will specialists adopt?
- Normalize images/Preprocess? How?
- How to deal with noise/artefacts?
- How to learn/optimize?
- Is Deep Learning complimenting Big Data?
Thank You

END OF PRESENTATION