This week’s topics

• Naïve Bayes (recap)

• Vector Space Models in NLP

• Latent Semantic Analysis

• Question-Answering

• Precision/Recall

• Watson

• Machine Translation
Change to Schedule

Wednesday Oct. 26: Support Vector Machines instead of Bayesian Networks
Readings for this week
(all on class webpage)

• B. Christian, *Mind vs. Machine*

• D. Ferrucci et al.,
  *Building Watson: An overview of the DeepQA project*

• P. F. Brown et al.,
  *A statistical approach to machine translation*

Optional:
• D. Bellos, *I, Translator*

• T. Adams,
  *Can Google break the computer language barrier?*
Chatbots

http://www.youtube.com/watch?v=Lr7qVQ3UoSk&feature=related
Naïve Bayes for text classification (Recap)

Assume we have a set of $C$ classes, and a set of training documents
\[ D = (d_1, d_2, \ldots, d_n). \]

Each document $d_i$ is represented as a list of term frequencies for the terms in the document:
\[ d_i = (t_1, t_2, \ldots, t_n) \]

Given a new document $d$, we classify $d$ as:
\[ \text{class}(d) = \arg \max_{c \in C} P(c \mid d) \]

By Bayes rule, we have:
\[ \text{class}_{NB}(d) = \arg \max_{c \in C} P(c \mid d) = \arg \max_{c \in C} \frac{P(c)P(d \mid c)}{P(d)} \]
\[ = \arg \max_{c \in C} P(c)P(d \mid c) \quad \text{(since the denominator doesn't depend on } c) \]
\[ = \arg \max_{c \in C} P(c) \prod_i P(t_i \mid c) \quad \text{(by Naive Bayes independence assumption)} \]
\[ = \arg \max_{c \in C} \left[ \log(P(c)) + \sum_i \log(P(t_i \mid c)) \right] \]
\[
\text{class}_{NB} = \arg \max_{c \in C} \left[ \log P(c) + \sum_{i} \log P(t_i | c) \right]
\]

(use with Add-1 Smoothing)
In-class exercise

Consider the following training set of book titles, in which stop words and punctuation have been removed and the remaining words have been stemmed. Assume total vocabulary is 100 words.

\[ \begin{align*}
\mathbf{d}_1 &: \text{“Introduction Artificial Intelligence”} \quad \text{Class: AI} \\
\mathbf{d}_2 &: \text{“Machine Intelligence: Introduction”} \quad \text{Class: AI} \\
\mathbf{d}_3 &: \text{“Mind Machine Cognitive System”} \quad \text{Class: AI} \\
\mathbf{d}_4 &: \text{“Introduction Operating System”} \quad \text{Class: OS} \\
\mathbf{d}_5 &: \text{“Operating System Element”} \quad \text{Class: OS} \\
\mathbf{d}_6 &: \text{“Machine Operating System”} \quad \text{Class: OS}
\end{align*} \]

Show how a naïve Bayes classifier trained on this training set would classify the following new example:

\[ \mathbf{d}_7 : \text{“Element Intelligence Machine System”} \]
Vector space model for text classification and information retrieval

A document is represented as a vector of word counts. E.g.,

“To be or not to be, that is the question. Whether 'tis nobler in the mind to suffer the slings and arrows of outrageous fortune, or to take arms against a sea of troubles, and by opposing end them?”

Vector \( \mathbf{d} = (c_1, c_2, \ldots, c_M) \), where \( M \) is the total number of words in the system’s dictionary.

\[
\begin{align*}
\text{arrows} & \quad \text{be} \quad \text{not} \quad \text{or} \quad \text{slings} \quad \text{to} \quad \text{...} \\
d & = (1 \quad \ldots \quad 2 \quad \ldots \quad 1 \quad \ldots \quad 2 \quad \ldots \quad 1 \quad \ldots \quad 4 \quad \ldots )
\end{align*}
\]
Suppose these terms are found in two on-line recipes:

<table>
<thead>
<tr>
<th>Recipe 1:</th>
<th>Recipe 2:</th>
</tr>
</thead>
<tbody>
<tr>
<td>Chicken: 8</td>
<td>Chicken: 6</td>
</tr>
<tr>
<td>Fried: 2</td>
<td>Fried: 0</td>
</tr>
<tr>
<td>Oil: 7</td>
<td>Oil: 0</td>
</tr>
<tr>
<td>Pepper: 4</td>
<td>Pepper: 0</td>
</tr>
</tbody>
</table>

\[ d_i = (8, 2, 7, 4) \]

\[ d_k = (6, 0, 0, 0) \]

Query: fried chicken
\[ q = (1, 1, 0, 0) \]
From Jurafsky and Martin, 2006

\[
\text{sim}(\vec{q}, \vec{d}_j) = \frac{\sum_{i=1}^{N} w_{i,q} \times w_{i,j}}{\sqrt{\sum_{i=1}^{N} w_{i,q}^2} \times \sqrt{\sum_{i=1}^{N} w_{i,j}^2}} = \frac{\text{dot-product}(\vec{q}, \vec{d}_j)}{\text{length}(\vec{q}) \times \text{length}(\vec{d}_j)} = \text{"normalized dot product"}
\]

**Figure 23.3** A graphical illustration of the vector model for information retrieval, showing the first two dimensions (fried and chicken) assuming that we use raw frequency in the document as the feature weights.
Suppose these terms are found in two on-line recipes (assume vocabulary size = 4)

Recipe 1:
Chicken: 8
Fried: 2
Oil: 7
Pepper: 4
d_i = (8, 2, 7, 4)

Recipe 2:
Chicken: 6
Fried: 0
Oil: 0
Pepper: 0
d_k = (6, 0, 0, 0)

Query: fried chicken
q = (1, 1, 0, 0)

\[
sim(q, d_j) = \frac{\sum_{i=1}^{N} w_{i,q} \times w_{i,j}}{\sqrt{\sum_{i=1}^{N} w_{i,q}^2} \times \sqrt{\sum_{i=1}^{N} w_{i,j}^2}}
\]
Suppose these terms are found in two on-line recipes
(assume vocabulary size = 4)

Recipe 1:
Chicken: 8
Fried: 2
Oil: 7
Pepper: 4
\[ \mathbf{d}_i = (8, 2, 7, 4) \]

Recipe 2:
Chicken: 6
Fried: 0
Oil: 0
Pepper: 0
\[ \mathbf{d}_k = (6, 0, 0, 0) \]

Query: fried chicken
\[ \mathbf{q} = (1, 1, 0, 0) \]

\[
\text{sim}(\mathbf{q}, \mathbf{d}_j) = \frac{\sum_{i=1}^{N} w_{i,q} \times w_{i,j}}{\sqrt{\sum_{i=1}^{N} w_{i,q}^2} \times \sqrt{\sum_{i=1}^{N} w_{i,j}^2}}
\]
Weaknesses of vector space model?

- Homonymy/Polysemy ("lie", "bore", "fly")

- Synonymy ("canine/dog", "party/celebration")
Improving performance of vector-space models

- Improve query:
Improving performance of vector-space models

- Improve query:
  - Relevance feedback
  - Query expansion

- Capture “meaning” from context:
  - Latent semantic analysis
Beyond N-grams: Latent Semantic Analysis

• Problem: How to capture semantic similarity between documents in a natural corpus (e.g., problems of homonymy, polysemy, synonymy, etc.)

• In general, N-grams, word frequencies, etc. often fail to capture semantic similarity, even with query expansion, etc.

• “LSA assumes that there exists a LATENT structure in word usage – obscured by variability in word choice”  (http://ir.dcs.gla.ac.uk/oldseminars/Girolami.ppt)
Latent Semantic Analysis
(Landauer et al.)

- From training data (large sample of documents), create term-by-document matrix.
Technical Memo Example

Titles:
c1: *Human machine interface* for Lab ABC computer applications
c2: A survey of *user* opinion of *computer system response time*
c3: The *EPS user interface* management system
c4: *System* and *human system engineering* testing of *EPS*
c5: Relation of *user*-perceived *response time* to error measurement

m1: The generation of random, binary, unordered *trees*
m2: The intersection *graph* of paths in *trees*
m3: *Graph minors IV*: Widths of *trees* and well-quasi-ordering
m4: *Graph minors*: A survey

A sample dataset consisting of the titles of 9 technical memoranda. Terms occurring in more than one title are italicized. There are two classes of documents - five about human-computer interaction (c1-c5) and four about graphs (m1-m4). This dataset can be described by means of a term by document matrix where each cell entry indicates the frequency with which a term occurs in a document.

*From Deerwester et al., Indexing by latent semantic analysis*
<table>
<thead>
<tr>
<th>Terms</th>
<th>c1</th>
<th>c2</th>
<th>c3</th>
<th>c4</th>
<th>c5</th>
<th>m1</th>
<th>m2</th>
<th>m3</th>
<th>m4</th>
</tr>
</thead>
<tbody>
<tr>
<td>human</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>interface</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>computer</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>user</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>system</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>2</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>response</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>time</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>EPS</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>survey</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>trees</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>graph</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>minors</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>
• Now apply “singular value decomposition” to this matrix

• SVD is similar to principal components analysis (if you know what that is)

• Basically, reduce dimensionality of the matrix by re-representing matrix in terms of “features” (derived from eigenvalues and eigenvectors), and using only the ones with highest value.

• Result: Each document is represented by a vector of features obtained by SVD.

• Given a new document (or query), compute its representation vector in this feature space, compute its similarity with other documents using cosine between vector angles. Retrieve documents with highest similarities.
### Result of Applying LSA

From http://lair.indiana.edu/courses/i502/lectures/lect6.ppt

<table>
<thead>
<tr>
<th></th>
<th>DOC1</th>
<th>DOC2</th>
<th>DOC3</th>
<th>DOC4</th>
<th>DOC5</th>
<th>DOC6</th>
<th>DOC7</th>
<th>DOC8</th>
<th>DOC9</th>
</tr>
</thead>
<tbody>
<tr>
<td>HUMAN</td>
<td>0.16</td>
<td>0.4</td>
<td>0.38</td>
<td>0.47</td>
<td>0.18</td>
<td>-0.05</td>
<td>-0.12</td>
<td>-0.16</td>
<td>-0.09</td>
</tr>
<tr>
<td>INTERFACE</td>
<td>0.14</td>
<td>0.37</td>
<td>0.33</td>
<td>0.4</td>
<td>0.16</td>
<td>-0.03</td>
<td>-0.07</td>
<td>-0.1</td>
<td>-0.04</td>
</tr>
<tr>
<td>COMPUTER</td>
<td>0.15</td>
<td>0.51</td>
<td>0.36</td>
<td>0.41</td>
<td>0.24</td>
<td>0.02</td>
<td>0.06</td>
<td>0.09</td>
<td>0.12</td>
</tr>
<tr>
<td>USER</td>
<td>0.26</td>
<td>0.84</td>
<td>0.61</td>
<td>0.7</td>
<td>0.39</td>
<td>0.03</td>
<td>0.08</td>
<td>0.12</td>
<td>0.19</td>
</tr>
<tr>
<td>SYSTEM</td>
<td>0.45</td>
<td>1.23</td>
<td>1.05</td>
<td>1.27</td>
<td>0.56</td>
<td>-0.07</td>
<td>-0.15</td>
<td>-0.21</td>
<td>-0.05</td>
</tr>
<tr>
<td>RESPONSE</td>
<td>0.16</td>
<td>0.58</td>
<td>0.38</td>
<td>0.42</td>
<td>0.28</td>
<td>0.06</td>
<td>0.13</td>
<td>0.19</td>
<td>0.22</td>
</tr>
<tr>
<td>TIME</td>
<td>0.16</td>
<td>0.58</td>
<td>0.38</td>
<td>0.42</td>
<td>0.28</td>
<td>0.06</td>
<td>0.13</td>
<td>0.19</td>
<td>0.22</td>
</tr>
<tr>
<td>EPS</td>
<td>0.22</td>
<td>0.55</td>
<td>0.51</td>
<td>0.63</td>
<td>0.24</td>
<td>-0.07</td>
<td>-0.14</td>
<td>-0.2</td>
<td>-0.11</td>
</tr>
<tr>
<td>SURVEY</td>
<td>0.1</td>
<td>0.53</td>
<td>0.23</td>
<td>0.21</td>
<td>0.27</td>
<td>0.14</td>
<td>0.31</td>
<td>0.44</td>
<td>0.42</td>
</tr>
<tr>
<td>TREES</td>
<td>-0.06</td>
<td>0.23</td>
<td>-0.14</td>
<td>-0.27</td>
<td>0.14</td>
<td>0.24</td>
<td>0.55</td>
<td>0.77</td>
<td>0.66</td>
</tr>
<tr>
<td>GRAPH</td>
<td>-0.06</td>
<td>0.34</td>
<td>-0.15</td>
<td>-0.3</td>
<td>0.2</td>
<td>0.31</td>
<td>0.69</td>
<td>0.98</td>
<td>0.85</td>
</tr>
<tr>
<td>MINORS</td>
<td>-0.04</td>
<td>0.25</td>
<td>-0.1</td>
<td>-0.21</td>
<td>0.15</td>
<td>0.22</td>
<td>0.5</td>
<td>0.71</td>
<td>0.62</td>
</tr>
</tbody>
</table>

In the above matrix we can now observe correlations:

\[ r(\text{human.user}) = 0.94 \]

\[ r(\text{human.minors}) = -0.83 \]
How does it find the latent associations?

By analyzing the contexts in which the words appear

The word *user* has co-occurred with words that *human* has co-occurred with (e.g., system and interface)

It downgrades associations when such contextual similarities are not found
Some General LSA Based Applications
From http://lsa.colorado.edu/~quesadaj/pdf/LSATutorial.pdf

Information Retrieval

Text Assessment
   Compare document to documents of known quality / content

Automatic summarization of text
   Determine best subset of text to portray same meaning

Categorization / Classification
   Place text into appropriate categories or taxonomies
Application: Automatic Essay Scoring
(in collaboration with Educational Testing Service)

Create domain semantic space

Compute vectors for essays, add to vector database

To predict grade on a new essay, compare it to ones previously scored by humans

From http://lsa.colorado.edu/~quesadaj/pdf/LSATutorial.pdf
Mutual information between two sets of grades:

human – human .90
LSA – human .81

From http://lsa.colorado.edu/~quesadaj/pdf/LSATutorial.pdf
Demo
(http://www.pearsonkt.com/cgi-bin/prenhall/phMenu.cgi?rubric=6&demo=1)
Question answering

http://www.youtube.com/watch?v=8BLzMCiR0G4

**Question answering**

<table>
<thead>
<tr>
<th>Question</th>
<th>Answer</th>
</tr>
</thead>
<tbody>
<tr>
<td>Where is the Louvre Museum located?</td>
<td>in Paris, France</td>
</tr>
<tr>
<td>What’s the abbreviation for limited partnership?</td>
<td>L.P.</td>
</tr>
<tr>
<td>What are the names of Odin’s ravens?</td>
<td>Huginn and Muninn</td>
</tr>
<tr>
<td>What currency is used in China?</td>
<td>the yuan</td>
</tr>
<tr>
<td>What kind of nuts are used in marzipan?</td>
<td>almonds</td>
</tr>
<tr>
<td>What instrument does Max Roach play?</td>
<td>drums</td>
</tr>
<tr>
<td>What’s the official language of Algeria?</td>
<td>Arabic</td>
</tr>
<tr>
<td>What is the telephone number for the University of Colorado, Boulder?</td>
<td>(303)492-1411</td>
</tr>
<tr>
<td>How many pounds are there in a stone?</td>
<td>14</td>
</tr>
</tbody>
</table>

**Figure 23.7** Some sample factoid questions and their answers.
Components of a question answering system

1. Question processing:
   
   – From question, create list of keywords that forms a query
2. Question classification

– What is the expected answer type?

“Who was the fourth U.S. president?”
   Expected answer type is proper noun, name of a person
“What is the name of China’s currency?”
   Expected answer is a proper noun, name of a currency
“What is the largest city in North America?”
   Expected answer is a proper noun, name of a city
“How long does it take to roast a turkey?”
   Expected answer is a number, in units of time

• Can use set of hand-coded rules
• Can use supervised machine learning
• In general, need ontologies!
3. Passage retrieval

– Submit query

– Extract set of potential answer passages from retrieved set of documents.

– Run *answer-type* classification on all passages. Filter out ones that don’t provide needed answer type.
– Rank remaining passages based on set of features

E.g.,

• Number of named entities of the right type present
• Number of question keywords present
• Rank of document containing passage
• Proximity of keywords from original query to each other.
• N-gram overlap between the passage and the question
4. Answer processing

- Extract specific answer from the passage

  • Use info about expected answer type together with regular expression patterns designed by programmer or learned
E.g.,

<AP> such as <QP>

(AP = answer phrase, QP = question phrase)

Example: “What is polysemy?”

– “linguistic terms such as polysemy”

<QP>, an <AP>

Example: “What is a caldera?“

– “the Long Valley caldera, a volcanic crater 19 miles long”
Another method for answer extraction: N-gram tiling

N-gram tiling algorithm:

- Present query

- For snippets returned from web search engine:
  - Compute all unigrams, bigrams, and trigrams
  - Weight each one as a function of number of snippets the N-gram occurred in
  - Score N-grams by how well they match predicted answer type (learn this from training data or build scorers by hand)
  - Concatenate best-scoring overlapping N-gram fragments into longer answers.
Example: Q: “What is the city in California with the largest population?”

Snippets:

“Los Angeles is, by California standards,“

“The largest city in California by population includes some of the most famous and beautiful beaches in the world.”

High-scoring n-grams:

“the largest city in California”
“Los Angeles is”
“California by population”
“by California standards”

Tiling: “Los Angeles is the largest city in California”
From Brill et al. (2002), An analysis of the AskMSR question-answering system
1. Determine expected answer type of question:

7 question types:
– who-question
– what-question
– where-question
– how-many question
– ...

Filters written manually.
2. Rewrite query as declarative sentence likely to be in text with answer:

“When was the paper clip invented?”

→ “the paper clip was invented”
→ “invented the paper clip”
→ “paper clips were invented”
→ “invented paper clips”
→ paper AND clip AND invented

Can do with simple rewrite rules.

Weights are given to rewrite rules (set manually).
3. **N-gram mining:**

Send query to search engine.

Collect top M page summaries (snippets)
– less computationally expensive than using entire web page

Extract unigrams, bigrams, and trigrams from each snippet.
4. N-gram filtering:

Filter and reweight N-grams according to how well each candidate N-gram matches expected answer type

(Uses human-constructed filters)
5. N-gram tiling:

Goal is to combine n-grams to produce longer answer

E.g., trigrams ABC and BCD become ABCD

“France is in” “is in Europe”

→ “France is in Europe”

Start with highest scoring N-gram as current answer

Iterate through remaining N-grams in order of score – for each one, see if it can be tiled with the current answer. If so, current answer ← tiling.
Results

• Tested on 500 queries from TREC data set

• 61% answered correctly.

• Relatively good performance at this competition!
Knowing when we don’t know

• However.... no answer is usually better than wrong answer. Need option of responding “I don’t know”.

• Built decision tree to predict whether system will answer correctly, based on set of features from question string:
  – unigrams, bigrams, sentence length, number of capitalized words, number of stop words, length of longest word.

• Decision tree didn’t work very well. (Are you surprised?)

• You can read about other attempts at this in their paper (in optional reading on the class web page)
Watson Videos

http://www.youtube.com/watch?v=12rNbGf2Wwo

What is Watson?

Adam Lally
Senior Software Engineer
IBM Research

www.ibmwatson.com
The hard part for Watson is **finding** and **justifying** the correct answer in the #1 spot...Computing a **confidence** that it’s right...and doing it fast enough to compete on Jeopardy!

- **Where was Einstein born?**
  
  *One day, from among his city views of Ulm, Otto chose a watercolor to send to Albert Einstein as a remembrance of Einstein’s birthplace.*

- **Welch ran this?**

  *If leadership is an art then surely Jack Welch has proved himself a master painter during his tenure at GE.*
The Jeopardy! Challenge: A compelling and notable way to drive and measure the technology of automatic Question Answering along 5 Key Dimensions

- $200
  If you're standing, it's the direction you should look to check out the wainscoting.

- $1000
  Of the 4 countries in the world that the U.S. does not have diplomatic relations with, the one that's farthest north

- $800
  In cell division, mitosis splits the nucleus & cytokinesis splits this liquid cushioning the nucleus

- Broad/Open Domain

- Complex Language

- High Precision

- Accurate Confidence

- High Speed
The Best Human Performance: *Our Analysis Reveals the Winner’s Cloud*

Top human players are remarkably good.

Computers?

Each dot represents an actual historical human Jeopardy! game

Winning Human Performance

Grand Champion Human Performance

2007 QA Computer System

More Confident

Less Confident
DeepQA: The Technology Behind Watson

Massively Parallel Probabilistic Evidence-Based Architecture

Generates and scores many hypotheses using a combination of 1000’s \textit{Natural Language Processing, Information Retrieval, Machine Learning} and \textit{Reasoning Algorithms}. These gather, evaluate, weigh and balance different types of \textit{evidence} to deliver the answer with the best support it can find.
Evidence Profiles summarize evidence analysis across many sources.

**Clue:** Chile shares its longest land border with this country.

Bolivia is more popular due to a commonly discussed border dispute.
Using Statistical Machine Learning different classes of evidence earn different weights

For example, Watson uses statistical machine learning to discover that Jeopardy! categories are only weak indicators of the answer type.

<table>
<thead>
<tr>
<th>U.S. CITIES</th>
<th>Country Clubs</th>
<th>Authors</th>
</tr>
</thead>
<tbody>
<tr>
<td>St. Petersburg is home to Florida's annual tournament in this game popular on shipdecks (Shuffleboard)</td>
<td>From India, the shashpar was a multi-bladed version of this spiked club (a mace)</td>
<td>Archibald MacLeish? based his verse play &quot;J.B.&quot; on this book of the Bible (Job)</td>
</tr>
<tr>
<td>Rochester, New York grew because of its location on this (the Erie Canal)</td>
<td>A French riot policeman may wield this, simply the French word for &quot;stick&quot; (a baton)</td>
<td>In 1928 Elie Wiesel was born in Sighet, a Transylvanian village in this country (Romania)</td>
</tr>
</tbody>
</table>
Through **training** Watson Evaluates and Selects documents worth analyzing for a given task.

For Jeopardy! Watson has analyzed and stored the equivalent of about 1 million books (e.g., encyclopedias, dictionaries, news articles, reference texts, plays, etc)

Too much irrelevant content requires unnecessary compute power
DeepQA: Incremental Progress in Precision and Confidence
Precision, Confidence & Speed

- **Deep Analytics** – We achieved champion-levels of Precision and Confidence over a huge variety of expression.

  ![Graph showing Precision and Confidence](image)

  Emily Dickinson 99%  
  Walt Whitman 60%  
  Barnard 10%

- **Speed** – By optimizing Watson’s computation for Jeopardy! on over 2,800 POWER7 processing cores we went from 2 hours per question on a single CPU to an average of just 3 seconds – fast enough to compete with the best.

- **Results** – in 55 real-time sparring games against former Tournament of Champion Players last year, Watson put on a very competitive performance in all games and winning 71% of the them!
Potential Business Applications

**Healthcare / Life Sciences:** Diagnostic Assistance, Evidence-Based, Collaborative Medicine

**Tech Support:** Help-desk, Contact Centers

**Enterprise Knowledge Management and Business Intelligence**

**Government:** Improved Information Sharing and Education
The Big Idea: Evidence-Based Reasoning over Natural Language Content

- **Deep Analysis** of clues/questions AND content
- Search for many possible answers based on different interpretations of question
- **Find, analyze and score EVIDENCE** from many different sources (not just one document) for each answer using many advanced NLP and reasoning algorithms
- **Combine evidence** and compute a confidence value for each possibility using statistical machine learning
- Rank answers based on confidence
- If top answer is above a threshold – buzz in else keep quiet
Watson is a lot more “scruffy” than “neat”
Evaluation: Types of tasks

- **Classification** (e.g., Task is to classify e-mail as spam/not spam)

- **Retrieval** (e.g., Task is to retrieve (from a database) all news articles about the Greek debt crisis)

- **Question answering** (e.g., use information sources to answer questions such as “Which country has the highest production of crude oil?”)
Evaluating classification algorithms

“Confusion matrix” for a given class $c$

<table>
<thead>
<tr>
<th>Actual</th>
<th>Predicted</th>
</tr>
</thead>
<tbody>
<tr>
<td>True (in class $c$)</td>
<td>True Positive</td>
</tr>
<tr>
<td></td>
<td>FalseNegative</td>
</tr>
<tr>
<td>False (not in class $c$)</td>
<td>FalsePositive</td>
</tr>
<tr>
<td></td>
<td>TrueNegative</td>
</tr>
</tbody>
</table>
Evaluating classification algorithms

• **Accuracy**: Fraction of correct answers out of all problems

• **Precision**: Fraction of true positives out of all predicted positives:
\[ P = \frac{TP}{TP + FP} \]

• **Recall**: Fraction of true positives out of all actual positives:
\[ R = \frac{TP}{TP + FN} \]
Example

Classification task: Is question a “pun”?

Data: 200 sample questions, of which 50 are puns

Results:

<table>
<thead>
<tr>
<th>Actual</th>
<th>Predicted</th>
</tr>
</thead>
<tbody>
<tr>
<td>True (in class <em>pun</em>)</td>
<td>True (in class <em>pun</em>)</td>
</tr>
<tr>
<td>False (not in class <em>pun</em>)</td>
<td>False (not in class <em>pun</em>)</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Precision? Recall?
Evaluating information retrieval algorithms

“Confusion matrix” for a given topic $c$

<table>
<thead>
<tr>
<th>Actual topic</th>
<th>Retrieved (predicted as class $c$)</th>
<th>Not retrieved (predicted as not in class $c$)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Relevant (on topic $c$)</td>
<td>TruePositive</td>
<td>FalseNegative</td>
</tr>
<tr>
<td>Not relevant (not on topic $c$)</td>
<td>FalsePositive</td>
<td>TrueNegative</td>
</tr>
</tbody>
</table>
Evaluating information retrieval algorithms

- Precision

\[ P = \frac{TP}{TP + FP} = \frac{\text{relevant and retrieved documents}}{\text{all retrieved documents}} \]

- Recall

\[ R = \frac{TP}{TP + FN} = \frac{\text{relevant and retrieved documents}}{\text{all relevant documents}} \]
Example

Retrieval task: Retrieve all articles on the Greek debt crisis

Data: 1000 news articles, of which 100 are on the Greek debt crisis

Results:

<table>
<thead>
<tr>
<th>Actual topic</th>
<th>Retrieved (predicted as on topic $c$)</th>
<th>Not retrieved (predicted as not on topic $c$)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Relevant (on topic $c$)</td>
<td>100</td>
<td>0</td>
</tr>
<tr>
<td>Not relevant (not on topic $c$)</td>
<td>200</td>
<td>700</td>
</tr>
</tbody>
</table>

Precision? Recall?
Evaluating question-answering algorithms

• Precision

\[ P = \frac{\text{correctly answered questions}}{\text{all questions answered}} \]

• Recall

\[ R = \frac{TP}{TP + FN} = \frac{\text{correctly answered questions}}{\text{all questions}} \]
Example

Question-answering task: Play Jeopardy!

Data: 40 questions

Total answered: 30

Total answered correctly: 15

Precision? Recall?
• In what cases would we care more about precision?

• In what cases would we care more about recall?
  – Classification
  – Information retrieval
  – Question answering
Next: Machine translation