Lecture 22:

Document and Web Applications

Sam Roweis

April 1, 2002

Machine Learning Problems for Text/Web Data

- Information Extraction / Entity Tagging / Disambiguation
  1. Get the author/title/company name/course title from this document or web page.
     e.g. Citeseer, WebKB
  2. Find the contact info (phone number/fax/email/etc) for specific people or positions at an organization.
     e.g. MarketIntelligence
  3. Find the salary/location/job title/description for a job posting on the web.
     e.g. FlipDog
- Basic idea is documents $\rightarrow$ databases.

Machine Learning Problems for Text/Web Data

- Document / Web Page Classification or Detection
  1. Does this document/web page contain an example of thing X?
     e.g. Job advertisements (FlipDog), spam email harvesting.
  2. Is this document/web page of type Y?
     e.g. Course homepages (US government)?
  3. Should we block this page/document? e.g. Spam detection, pornography content filtering.
  4. Directed crawling. Active crawling.

Machine Learning Problems for Text/Web Data

- Searching / Indexing / Collaborative Filtering
  1. Sometimes called "information retrieval".
  2. Find the most "relevant" document/product/movie/song given these search terms.
     e.g. Google
  3. Find the documents/products most like the list given.
     e.g. Amazon Recommendations
  4. Index collections of crosslinked items into an automatic hierarchy.
     e.g. Citeseer

Document Classification Models

- Basic models for classification:
  1. Naive Bayes
  2. Logistic Regression (MaxEnt)
  3. Support Vector Machines
  4. Decision Trees
  5. Winnow
- More sophisticated models:
  1. Mixtures of Naive Bayes
  2. Latent Probabilistic Semantic Indexing
  3. Latent Dirichlet Allocation
  4. Boosted Decision Trees

Some Tips

- Must use stop word lists and stemming.
- Naive Bayes is an excellent model, and often very hard to beat.
  Always add one to your counts.
- MaxEnt/Logistic Regression: don't use iterative scaling to update parameters, use conjugate gradient instead.
  Always use a quadratic prior on the weights.
- Feature selection is key.
  Some common approaches: max mutual info with class label; most frequent non-stopwords, non-stopwords appearing in most documents.

Document Features for Classification

- Binary word occurrence
- Word counts / log counts
- Binary presence on one or more lists of names, cities, companies, states, countries, products, television shows, etc.
- TFIDF: Term Frequency * Inverse Document Frequency
- Trigger Phrases

Information Extraction Models

- Use latent variable models where latent variable indicates entity groupings and observables are words or other document features.
- Hidden Markov Models
- Maximum Entropy Markov Models
- Conditional Random Fields
- LocalBayes

Labelled vs. Unlabelled Data

- For information extraction / named-entity tagging, most models require labelled data, which can be very difficult to get in large quantities.
  At WhizBang such data is generated by a a special "DataShack" in which CMU students hand label pages and documents.
- Sometimes it is possible to "bootstrap" up from a small amount of labelled data to a larger amount.
  e.g. word sense disambiguation with LocalBayes

Recommendation/Collaborative Filtering

- The most basic recommendation system is table-lookup into the past: given what you already like, recommend things that other people who liked what you do also liked.
- This only works if you have an enormous amount of data (e.g. weather prediction, Amazon).
- In general, we must group documents/products together based on co-occurrence and then extrapolated from our limited database to discover which items to recommend.
  e.g. Aspect Model

Web Searching Models

- Most web search engines work by using a combination of two technologies:
  1. Highly efficient index and search for retrieving a list of pages that contain the keywords you search for, or a set of words that mean roughly the same thing. (This is a databases problem.)
  2. A method of *ranking* the matching results so that the "best" or "most relevant" pages come earlier on the list. (This is a machine learning problem.)
- Virtually all ranking algorithms are eigenvector methods applied to the link matrix of the web.
  e.g. Google, hubs and authorities
- "Bibliometrics" treats citations in documents like links on the web.

Summarization

- Generate a small amount of text that summarizes a larger document.
  e.g. Google news.
- Very hard problem because the computer has to generate some believable content.
- Easier problem: exceprt a small amount of original text or audio or video that best captures the entire document.

Evaluation Metrics

- Perplexity: average number of plausible alternatives on test set.
- Precision vs. Recall curves.
- F-score: $(\beta^2 + 1)PR/(\beta^2 P + R)$
- N-best performance.
- Ranking performance.