Efficient Web Spidering with Reinforcement Learning

Author: Jason Rennie, Andrew McCallum
Presenter: Bo Zhang
Outline

- Web Spidering Overview
- Web Spidering as Reinforcement Learning
- Experimental Setup
- Future Work and Related Work
Web Spidering Overview

- Agents that explore the hyperlink graph of the web
- Key to high coverage by search engines
  - Alta Vista, Hotbot
- Aim to find more distinct web pages
- Avoid off-topic area
  - To find pages of particular kind on a particular topic
Web Spidering Overview

- Cora - domain specific search engine for CS research papers
  - Finds title, authors, abstract, references
  - Resolves forward/backward references
  - [www.cora.justresearch.com](http://www.cora.justresearch.com) :( not working
Web Spidering as Reinforcement Learning

Reinforcement Learning

- State set: $s \in S$
- Action set: $a \in A$
- Transition Function: $T : S \times A \rightarrow S$
- Reward Function: $R : S \times A \rightarrow \mathbb{R}$
- Goal: to learn a policy, a mapping from states to actions: $\pi : S \rightarrow A$
Web Spidering as Reinforcement Learning

More …

- Discount factor: \( 0 \leq \gamma < 1 \)
  - Sooner reward is better than later reward
- Value of each state: \( V^\pi(s) = \sum_{t=0}^{\infty} \gamma^t r_t \)
  - For policy: \( \pi \)
- Optimal policy: \( \pi^* \)
  - Value function of Optimal policy: \( V^* \)
- Value of selecting action \( a \) from state \( s \)
  \[ Q^*(s, a) = R(s, a) + \gamma V^*(T(s, a)) \]
Web Spidering as Reinforcement Learning

More …

- Optimal Policy: \( \pi^*(s) = \arg\max_a Q^*(s, a) \)
Cheese-finding v.s. Spidering

Maze, mouse, cheese
- Receives a reward for finding a piece
- Only immediate reward
- To act optimally, must count future rewards

Spidering
- On-topic documents are immediate reward
- Action: follow a link
- State: locations to be consumed
- Number of actions is large and dynamic
Web Spidering as Reinforcement Learning

Why Reinforcement Learning is the proper frame work

- Performance is measured in terms of reward over time
- The environment presents situations with delayed reward
Practical Approximation

Goal: practically solved

Problems

- State space is huge: $2^{\text{(# of on-topic doc)}}$
- Action space is large: # of URLs

Assumptions for simplification

- State is independent of the documents consumed, collapse all states into one
- Actions are distinguished by the the words in the neighborhood of the hyperlink
Practical Approximation

- With these assumptions
  - Q function becomes a mapping:
    - bag-of-words -> scalar (sum of future reward)
  - Two sub-problems
    - Assigning Q values to hyperlinks in training set
    - Learning a mapping from text to Q values
Value Criterion

Assigning Q values to hyperlinks

- Simplest mapping
  - 1 to those points to a research paper
  - 0 to others
- Equivalent to RL framework with $\gamma = 0$.

- Move involved criteria
  - Calculate discounted sum over rewards of the hyperlinks from the web page $\gamma > 0$.
Value Criterion

A, B – hyperlinks
Circles – documents
Hexagon – spider
Filled-in circles – reward
Immediate reward always better than future reward
Value Criterion

Why immediate reward > future reward?
- Action A: retrieves a paper, reward 1
- Action B: a web page links to 1000 papers
- If use A, then B, reward is “1,0,1,1,…”
- If use B, then A, reward is “0,1,1,1,…”

Conclusion:
- Achieve reward as early as possible
Neighborhood Text

To compare known hyperlinks to unknown hyperlinks, use neighborhood text of the hyperlinks.

- E.g. anchor text of a hyperlink

Neighborhood Text

- Association each hyperlink with two sets of words
  - The anchor text and tokens from the URL
  - The full text of the web page where the hyperlink is located
  - Each hyperlink is identified by the two sets
Naive Bayes

Terms

- A document class: $c_j$
- Document frequency: $P(c_j)$
- A word: $w_t$  Vocabulary: $V$
- The frequency that the classifier expects the word to occur in documents of the class: $P(w_t|c_j)$
- A document: $d_i$
Naive Bayes

- Naïve Bayes assumptions to classify documents:
  - Word occur independently
- Calculate probability of each class with given evidence of the document: \( P(c_j | d_i) \)
- The kth word in document:
  
  \[
  P(c_j | d_i) \propto P(c_j) P(d_i | c_j) \prod_{k=1}^{|[d_i]|} P(w_{d_{ik}} | c_j)
  \]
Naive Bayes

Goal: To learn the parameters:

\[ P(c_j) \quad P(w_t | c_j) \]

Method: Using a set of labeled training documents

(See paper for detailed formulas)
Regression as Classification

To construct the model

- Discretize Q values: the discounted sum of future reward values of training data
- Place the hyperlinks into bins according to Q values
- Run Naïve Bayes text classifier
Regression as Classification

- To determine the Q value of an unknown hyperlink
  - Compute the probability of the class membership for each bin
  - Compute a weighted average of the bins’ average Q value
Experiment Setup

Data

- CS department of Brown, Cornell, U of Pittsburgh, U of Texas Austin
- 53,012 web pages and ps files, 592,216 hyperlinks
- Target: computer science research paper
- Criteria for research paper:
  - Abstract, Introduction, references, bibliography
  - 200 ps files, 95% precision, 2,263 papers identified
Experiments

- Baseline: FIFO action queue
  - Follow hyperlinks in order, breath-first spider
- 3 spiders
  - Immediate spider: $\gamma = 0$, two classes (0,1)
  - Future spider: $\gamma > 0$
  - Distance spider: combining two spiders
Distance Spider

Properties of distance spider
- Rank research papers above all others
- Prioritize others by their future rewards
- No discounted sum for keeping Q<=1

<table>
<thead>
<tr>
<th>Max Reward</th>
<th>Distance to nearest reward</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0.000</td>
</tr>
<tr>
<td>1–100</td>
<td>0.500</td>
</tr>
<tr>
<td>101+</td>
<td>0.500</td>
</tr>
</tbody>
</table>
Evaluation Metrics

- Number of pages retrieved before half of the papers retrieved
  - Simple, intuitive, but not may have incredible difficulty for more

- Sum of reward
  - Each reward is discounted by one minus the percent of web pages to be retrieved
  - Calculating area under curve, Integral scores
Results

Spidering CS Departments

Percent Research Papers Found

Percent Hyperlinks Followed

Future
Distance
Breadth-First
Results

- Distance and Future performs significantly better than breath-first.
- Surprising fact:
  - Immediate spider outperforms distance and future.
  - However, distance is significantly better than immediate spider at the early stages.
Immediate v.s. Distance

[Graph showing the comparison between Immediate, Distance, and Breadth-First methods in spidering CS departments]
Results – early stages
Discussion

Why immediate spider works so well

- Some words are commonly used by CS department and professors
- E.g. “technical”, “report”, and “papers”
Future Work

Facts

- Distance spider performs better at early stages
- Immediate spider performs better in the long run

Combine the properties of two spiders
Future Work

- Wrong Assumption
  - State of a spider is not important for identifying the value of an action.
  - E.g. choosing action A may change the Q value of action B
  - Using more features of a link, e.g. HTML, and web around the link