Machine Learned Sentence Selection Strategies for Query-Biased Summarization

Donald Metzler and Tapas Kanungo
Yahoo! Labs

July 24, 2008
SIGIR Learning to Rank Workshop
Each summary consists of a title, abstract, and URL.
Overview of Query-Biased Summarization

• **Input:** query, document pair
• **Pre-processing Step**
  – Segment document into sentences or passages
  – Done offline at index time
• **Sentence Selection**
  – Identify sentences that are most relevant to the query
  – Return ranked list of sentences + scores
• **Construction**
  – Compress the sentences to maximize query term coverage, novelty, readability, etc.
  – Must make sure everything fits within screen real estate
This talk focuses entirely on the sentence selection task.

Typically framed as an information retrieval problem:
  - Find the sentences within the document that “best match” the query.

Many different features important for sentence selection.
Our Approach

• We cast the sentence selection problem as a machine learning problem

• Pros
  – Ability to use many different features
  – Principled parameter estimation

• Cons
  – Need training data
  – Less efficient than rule-based system
Related Work

- **Sentence selection**
  - Query independent summaries [Kupiec and Pederson ‘95]
  - Usefulness of query-biased summaries [Tombros and Sanderson ‘98]
  - TREC Novelty Track / DUC [2001-2004]
  - Machine learned query-biased summaries [Wang et al. ‘07]

- **Learning to rank**
  - Logistic regression, SVMs, maximum entropy, perceptrons, ranking SVMs, RankNet, LambdaRank, ordinal regression, gradient boosted decision trees, and on, and on, and on…
• What do we model?
  – Input: query / sentence pair
  – Output: real-valued “relevance” score

• Consider two ‘classes’ of models
  – Pairwise ranking models
  – Regression models
• Learn pairwise preferences
  – \( P \) is a set of pairwise preferences
  – \((s_1, s_2)\) in \( P \) \( \Rightarrow \) sentence 1 is preferred to sentence 2

• Shown to be effective for ranking problems
  – [Joachims, KDD ’02]
  – [Burges et al., ICML ’05]
Given a query / document pair, suppose our training data was the following:
- Sentence 1 is relevant
- Sentence 2 is non-relevant
- Sentence 3 is relevant
- Sentence 4 is non-relevant

Encode these judgments as pairwise preferences as follows:
- \( P = \{ (s_1, s_2), (s_1, s_4), (s_3, s_2), (s_3, s_4) \} \)

There are \((\# \text{ relevant}) \cdot (\# \text{ non-relevant})\) total elements in \(P\) for each query.
• Ranking SVMs
  – Equivalent to ‘classical’ SVMs learned over pairwise preferences
  – Uses hinge loss

• Formulation:

\[
\begin{align*}
\text{min} & \quad \frac{1}{2} \|w\|^2 + C \sum_{i,j} \xi_{i,j} \\
\text{s.t.} & \quad (w \cdot x_i - w \cdot x_j) \geq 1 - \xi_{i,j} \quad \forall (i,j) \in \mathcal{P} \\
& \quad \xi_{i,j} \geq 0 \quad \forall (i,j) \in \mathcal{P}
\end{align*}
\]
\[ w \cdot (x_i - x_j) > 0 \]

\[ w \cdot (x_i - x_j) < 0 \]

\[ w \cdot (x_i - x_j) = 0 \]

Hyperplane

Margin
• Directly models the response (human judgment)
• Have recently been shown to be highly effective for learning to rank
  – [Li et al., NIPS ’07]
  – [Zheng et al., NIPS ’07]
• Consider two regression models here
  – Support vector regression
  – Gradient boosted decision trees
• **ε-SVM regression**
  – ε-sensitive hinge loss over residuals
  – Use different costs for $y = 1$, $y = -1$
  – Constraints: $|y - f(x)| \leq \varepsilon$

• **Formulation:**

$$
\begin{align*}
\min \quad & \frac{1}{2}||w||^2 + C_+ \sum_{i:y_i=1}(\xi_i + \xi_i^*) + C_- \sum_{i:y_i=-1}(\xi_i + \xi_i^*) \\
\text{s.t.} \quad & y_i - w \cdot x_i - b \leq \varepsilon + \xi_i \\
& w \cdot x_i + b - y_i \leq \varepsilon + \xi_i^* \\
& \xi_i, \xi_i^* \geq 0
\end{align*}
$$
Illustration of $\varepsilon$-SVM regression and loss function from Smola and Schölkopf '03.
Gradient Boosted Decision Trees

• Additive model of the form:

\[ f_M(x) = \sum_{m=1}^{M} T(x; \Theta_m) \]

• Where \( T(x; \theta) \) is a regression tree with parameters \( \theta \)
• Adds a new tree to the model during each iteration
• The new tree is fit to the residuals of the loss from the \((m-1)^{th}\) stage

[Friedman '01]
For regression, the loss is mean squared error
   – Could also use more complex losses
   – Even those that are non-differentiable!

Trees learned during each iteration are typically ‘stumps’ (trees of depth 1)

Automatic feature selection built in
Gradient Boosted Decision Trees

Iteration #1

\( x_1 < 10 \)

- **NO**
  - 1
- **YES**
  - 5

Iteration #2

\( x_3 > 1 \)

- **NO**
  - -10
- **YES**
  - -2

\[ f(x_1 = 5, x_2 = 3, x_3 = 3) = 1 + (-2) + \ldots \]

\[ f(x_1 = 20, x_2 = 0, x_3 = 5) = 5 + (-2) + \ldots \]
Loss Functions

• Loss functions for our learning models
  – RankSVMs (hinge loss over pairs of scores)
  – SVR (hinge loss over residuals)
  – GBDTs (MSE)

• Our ultimate goal is to maximize some information retrieval metric, such as F1 or R-Precision
  – However, such metrics are non-differentiable and difficult to optimize directly
  – None of the learning methods use actually optimize these metrics

• We use a heuristic training procedure in order to implicitly maximize the metrics of interest
Algorithm 1 Evaluation Algorithm

for \( i = 1 \) to \( 5 \) do

\( (TRAIN, VALIDATE) \leftarrow \text{split}(TRAIN_i, p) \)

utility\(_{\text{max}} \leftarrow -\infty \)

for \( \theta \in \Theta \) do

model \leftarrow \text{train}(TRAIN; \theta) \)

utility \leftarrow \text{eval}(model, VALIDATE) \)

if utility > utility\(_{\text{max}} \) then

model\(_{\text{max}} \leftarrow \) model

end if

end for

output \( \text{rank}(TEST_i, model_{\text{max}}) \)

end for
Features

• Query dependent
  – Exact match
  – Overlap
  – Overlap w/ synonyms
  – Language modeling score

• Query independent
  – Sentence length
  – Sentence location
Sentence Filtering

- Scores produced by ML models can be used to rank sentences
- In practice, want to filter result set to include only the most relevant documents
- How can we choose the number of sentences to return to the abstract composer?
  - Fixed depth
    - Choose the same number of sentences per document
  - Global score threshold
    - Only choose those sentences with a score greater than some global threshold
**Fixed Depth Filtering**
*(Depth = 3)*

<table>
<thead>
<tr>
<th>(Q1, D1)</th>
<th>(Q1, D2)</th>
<th>(Q1, D3)</th>
</tr>
</thead>
<tbody>
<tr>
<td>10</td>
<td>5</td>
<td>10</td>
</tr>
<tr>
<td>5</td>
<td>2</td>
<td>10</td>
</tr>
<tr>
<td>3</td>
<td>1</td>
<td>10</td>
</tr>
<tr>
<td>2</td>
<td>1</td>
<td>10</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

**Green** = Retrieved, **Red** = Not Retrieved
Global Score Threshold Filtering (Threshold = 5)

(Q1, D1)

10
5
3
2
1

(Q1, D2)

5
2
1
1
1

(Q1, D3)

10
10
10
9
1

Green = Retrieved, Red = Not Retrieved
Evaluation

• Data
  – TREC Novelty track data from 2002-2004

• Human judgments
  – For every query / document pair, all of the sentences in the document are judged to be relevant or non-relevant to the query

• Data sets have differing characteristics

<table>
<thead>
<tr>
<th></th>
<th>N2002</th>
<th>N2003</th>
<th>N2004</th>
</tr>
</thead>
<tbody>
<tr>
<td>Query / Doc. Pairs</td>
<td>597</td>
<td>1187</td>
<td>1214</td>
</tr>
<tr>
<td>Avg. Sentences per Pair</td>
<td>52.1</td>
<td>31.9</td>
<td>30.5</td>
</tr>
<tr>
<td>Avg. Relevant Sentences per Pair</td>
<td>2.3</td>
<td>13.1</td>
<td>6.9</td>
</tr>
</tbody>
</table>
**Sentence Selection Evaluation**

<table>
<thead>
<tr>
<th></th>
<th>N2002</th>
<th>N2003</th>
<th>N2004</th>
</tr>
</thead>
<tbody>
<tr>
<td>LM</td>
<td>.2602</td>
<td>.5566</td>
<td>.3944</td>
</tr>
<tr>
<td>Ranking SVM</td>
<td>$\alpha$</td>
<td>$\alpha$</td>
<td>$\alpha$</td>
</tr>
<tr>
<td>SVR</td>
<td>$\alpha$</td>
<td>$\beta$</td>
<td>$\alpha$</td>
</tr>
<tr>
<td>GBDT</td>
<td>$\alpha$</td>
<td>$\beta$</td>
<td>$\alpha$</td>
</tr>
</tbody>
</table>

**R-Precision** for each data set and sentence selection approach. The $\alpha$, $\beta$, and $\delta$ subscripts indicate a statistically significant improvement over language modeling, ranking SVMs, and SVR, respectively, according to a one-tailed pair $t$-test with $p < 0.05$.

**Summary:**
Machine learned techniques always significantly better than language modeling. GBDTs significantly outperform ranking SVMs on two out of three data sets, and SVR on one data set.
**Sentence Filtering Evaluation**

<table>
<thead>
<tr>
<th></th>
<th>N2002</th>
<th></th>
<th>N2003</th>
<th></th>
<th>N2004</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Depth</td>
<td>Thresh.</td>
<td>Depth</td>
<td>Thresh.</td>
<td>Depth</td>
</tr>
<tr>
<td>Ranking SVM</td>
<td>.3411</td>
<td>.2474</td>
<td>.5794</td>
<td>.6330</td>
<td>.4416</td>
</tr>
<tr>
<td>SVR</td>
<td>.3350</td>
<td>.2880</td>
<td>.5791</td>
<td>.6503</td>
<td>.4407</td>
</tr>
<tr>
<td>GBDT</td>
<td>.3576</td>
<td>.3302</td>
<td>.5771</td>
<td>.6691</td>
<td>.4389</td>
</tr>
</tbody>
</table>

Comparison of result set filtering methods. For each data set, the optimal F1 measure for each technique is reported.

**Summary:**
Fixed depth thresholding better when there are very few relevant sentences per document and global score thresholding better when there are many relevant sentences per document.
Summary:
Global score thresholding with GBDTs is more stable across data sets than ranking SVMs and SVR. Setting the GBDT threshold to -0.55 results in an F1 that is within 2% of the optimal F1 on all three data sets.
Feature Importance Analysis

Novelty 2002

Overlap w/ Synonyms
Overlap
Sentence Length
LM Score
Sentence Location
Exact Match

Novelty 2003

Sentence Length
LM Score
Sentence Location
Overlap w/ Synonyms
Overlap
Exact Match
Conclusions

• Machine learned sentence selection is not only feasible, but very effective
• Regression-based model, and gradient boosted decision trees, in particular, are a robust model choice
• Different result set filtering techniques are appropriate for different types of data sets