Learning to Rank using Language Models and SVMs

Berlin Chen
Department of Computer Science & Information Engineering
National Taiwan Normal University

References:
2. Raymond J. Mooney’s teaching materials
Discriminatively-Trained Language Models (1/9)

• A simple document-based language model (LM) for information retrieval can be represented by

\[ P(Q | D \text{ is } R) = \prod_{n=1}^{N} \left[ m_1 P(q_n | D) + m_2 P(q_n | \text{Corpus}) \right] \]

– The use of general corpus LM \( P(q_n | \text{Corpus}) \) is for probability smoothing and better retrieval performance

– Conventionally, the mixture weights \( m_1, m_2 \) \( (m_1 + m_2 = 1) \) are empirically tuned or optimized by using the Expectation-Maximization (EM) algorithm

Discriminatively-Trained Language Models (2/9)

- For those documents with training queries, $m_1$ and $m_2$ can be estimated by using the Minimum Classification Error (MCE) training algorithm
  - The ordering of relevant documents $D^*$ and irrelevant documents $D'$ in the ranked list for a training query exemplar $Q$ is adjusted to preserve the relationships $D^* \prec D'$; i.e., $D^*$ should precede $D'$ on the ranked list
  - A learning-to-rank algorithm
  - Documents thus can have different weights

Discriminatively-Trained Language Models (3/9)

- **Minimum Classification Error (MCE) Training**
  - Given a query $Q$ and a desired relevant doc $D^*$, define the **classification error function** as:

  $$E(Q, D^*) = \frac{1}{|Q|} \left[ -\log P(Q|D^* \text{ is } R) + \max_{D'} \log P(Q|D' \text{ is not } R) \right]$$

  Also can take all irrelevant doc in the answer set into consideration

  “$>0$”: means misclassified; “$\leq 0$”: means a correct decision

- Transform the error function to the **loss function**

  $$L(Q, D^*) = \frac{1}{1 + \exp(-\alpha E(Q, D^*) + \beta)}$$

  - In the range between 0 and 1
    - $\alpha$: controls the slope
    - $\beta$: controls the offset
Discriminatively-Trained Language Models (4/9)

- Minimum Classification Error (MCE) Training
  - Apply the loss function to the MCE procedure for iteratively updating the weighting parameters
  - Constraints:
    \[ m_k \geq 0, \quad \sum_k m_k = 1 \]
  - Parameter Transformation, (e.g., Type I HMM)
    \[ m_1 = \frac{e^{\tilde{m}_1}}{e^{\tilde{m}_1} + e^{\tilde{m}_2}} \quad \text{and} \quad m_2 = \frac{e^{\tilde{m}_2}}{e^{\tilde{m}_1} + e^{\tilde{m}_2}} \]
    - Iteratively update \( m_1 \) (e.g., Type I HMM)
      \[ \tilde{m}_1(i + 1) = \tilde{m}_1(i) - \varepsilon(i) \cdot \frac{\partial L(Q, D^*)}{\partial \tilde{m}_1} \]
      \[ D^* = D^*(i) \]
      - Where,
        \[ \nabla_{D^*},\tilde{m}_1 = \varepsilon(i) \cdot \frac{\partial L(Q, D^*)}{\partial \tilde{m}_1} = \varepsilon(i) \cdot \frac{\partial L(Q, D^*)}{\partial E(Q, D^*)} \cdot \frac{\partial E(Q, D^*)}{\partial \tilde{m}_1} \]
        \[ = \varepsilon(i) \cdot \alpha \cdot L(Q, D^*) \cdot [1 - L(Q, D^*)] \]
Discriminatively-Trained Language Models (5/9)

- Minimum Classification Error (MCE) Training
  - Iteratively update \( m_1 \) (e.g., Type I HMM)

\[
\frac{\partial E(Q, D^*)}{\partial \tilde{m}_1} = -\frac{1}{|Q|} \sum_{q_n \in Q} \log \left[ \frac{e^{\tilde{m}_1}}{e^{\tilde{m}_1} + e^{\tilde{m}_2}} P(q_n | D^*) + \frac{e^{\tilde{m}_2}}{e^{\tilde{m}_1} + e^{\tilde{m}_2}} P(q_n | \text{Corpus}) \right] \]

\[
= -\frac{1}{|Q|} \sum_{q_n \in Q} \left[ \frac{-e^{\tilde{m}_1}}{(e^{\tilde{m}_1} + e^{\tilde{m}_2})^2} \left[ e^{\tilde{m}_1} P(q_n | D^*) + e^{\tilde{m}_2} P(q_n | \text{Corpus}) \right] + \frac{e^{\tilde{m}_1}}{e^{\tilde{m}_1} + e^{\tilde{m}_2}} P(q_n | D^*) \right] \]

\[
= \frac{e^{\tilde{m}_1}}{e^{\tilde{m}_1} + e^{\tilde{m}_2}} - \frac{1}{|Q|} \sum_{q_n \in Q} \left[ \frac{e^{\tilde{m}_1}}{e^{\tilde{m}_1} + e^{\tilde{m}_2}} P(q_n | D^*) \right] \]

\[
= -\left[ -m_1 + \frac{1}{|Q|} \sum_{q_n \in Q} m_1 P(q_n | D^*) \right],
\]
Discriminatively-Trained Language Models (6/9)

- Minimum Classification Error (MCE) Training
  
  \[
  \nabla_{D^*, \tilde{m}_1} (i) = -\varepsilon (i) \cdot \alpha \cdot L(Q, D^*) \cdot [1 - L(Q, D^*)] \\
  \cdot \left[ -m_1 (i) + \frac{1}{|Q|} \sum_{q_n \in Q} \frac{m_1 (i) P(q_n | D^*)}{m_1 (i) P(q_n | D^*) + m_2 (i) P(q_n | \text{Corpus})} \right],
  \]

  \[m_1 (i + 1) = \frac{e^{\tilde{m}_1 (i+1)}}{e^{\tilde{m}_1 (i+1)} + e^{\tilde{m}_2 (i+1)}}\]

  \[= \frac{e^{\tilde{m}_1 (i)} e^{-\nabla_{D^*, \tilde{m}_1} (i)}}{e^{\tilde{m}_1 (i)} e^{-\nabla_{D^*, \tilde{m}_1} (i)} + e^{\tilde{m}_2 (i)} e^{-\nabla_{D^*, \tilde{m}_2} (i)}}\]

  \[= \frac{e^{\tilde{m}_1 (i)} e^{-\nabla_{D^*, \tilde{m}_1} (i)}}{(e^{\tilde{m}_1 (i)} + e^{\tilde{m}_2 (i)})} \left[ \frac{e^{\tilde{m}_1 (i)} e^{-\nabla_{D^*, \tilde{m}_1} (i)}}{(e^{\tilde{m}_1 (i)} + e^{\tilde{m}_2 (i)})} + \frac{e^{\tilde{m}_2 (i)} e^{-\nabla_{D^*, \tilde{m}_2} (i)}}{(e^{\tilde{m}_1 (i)} + e^{\tilde{m}_2 (i)})} \right],\]

  \[= \frac{m_1 (i) \cdot e^{-\nabla_{D^*, \tilde{m}_1} (i)}}{m_1 (i) \cdot e^{-\nabla_{D^*, \tilde{m}_1} (i)} + m_2 (i) \cdot e^{-\nabla_{D^*, \tilde{m}_2} (i)}},\]
Discriminatively-Trained Language Models (7/9)

- Minimum Classification Error (MCE) Training
  - Final Equations
    - Iteratively update \( m_1 \)
      
      \[
      \nabla_{D^*, \tilde{m}_1} (i) = -\varepsilon(i) \cdot \alpha \cdot L(Q, D^*) \cdot [1 - L(Q, D^*)] \\
      \cdot \left[ -m_1(i) + \frac{1}{|Q|} \sum_{q_n \in Q} m_1(i) P(q_n|D^*) + m_2(i) P(q_n|Corpus) \right] \\
      \]

      \[
      m_1(i + 1) = \frac{m_1(i) \cdot e^{-\nabla_{D^*, \tilde{m}_1}(i)}}{m_1(i) \cdot e^{-\nabla_{D^*, \tilde{m}_2}(i)} + m_2(i) \cdot e^{-\nabla_{D^*, \tilde{m}_2}(i)}}
      \]

    - \( m_2 \) can be updated in the similar way
Discriminatively-Trained Language Models (8/9)

- Experimental results with MCE training

<table>
<thead>
<tr>
<th>Average Precision</th>
<th>Word-level</th>
<th>Syllable-level</th>
<th>Fusion</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Uni</td>
<td>Uni+Bi*</td>
<td></td>
</tr>
<tr>
<td>TQ/TD</td>
<td>0.6459</td>
<td>0.6858</td>
<td>0.7329</td>
</tr>
<tr>
<td>(0.6327)</td>
<td>(0.5718)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>TDT2</td>
<td>0.5810</td>
<td>0.6300</td>
<td>0.6914</td>
</tr>
<tr>
<td>(0.5658)</td>
<td>(0.5307)</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

- The results for the syllable-level indexing features were significantly improved
Discriminatively-Trained Language Models (9/9)

• Similar treatments have been recently applied to Document Topic Models (e.g., PLSA) and Word Topic Models (WTM) with good success

• For example, the ranking formula for PLSA can be represented by

\[
P(q|D) = \alpha \left( \beta \cdot \left[ \sum_{T_k} P(q|T_k)P(T_k|D) \right] + (1 - \beta) \cdot P(q|\text{Corpus}) \right) + (1 - \alpha) \cdot P(q|D)
\]

\[
= \sum_{T_k} \alpha \beta \cdot P(q|T_k)P(T_k|D) + \alpha (1 - \beta) \cdot P(q|\text{Corpus}) + (1 - \alpha) \cdot P(q|D)
\]

\[
= \sum_{T_k} \left[ \alpha \beta \cdot P(q|T_k) + \alpha (1 - \beta) \cdot P(q|\text{Corpus}) + (1 - \alpha) \cdot P(q|D) \right]P(T_k|D)
\]

– The weighting parameters \( \alpha \) and \( \beta \) document topic distributions \( P(T_k|D) \) can be trained by the MCE algorithm
Vector Representations

• Data points (e.g., documents) of different classes (e.g., relevant/non-relevant classes) are represented as vectors in a $n$-dimensional vector space
  – Each dimension has to do with a specific feature, whose value usually is normalized

  \[ \mathbf{w}^T \mathbf{x} + b \]

• Support vector machines (SVM)
  – Look for a decision surface (or hyperplane) that is maximally far away from any data point
  – Margin: the distance from the decision surface to the closest data points on either side (or the support vectors)
  – SVM is a kind of large-margin classifier
**Support Vectors**

- SVM is fully specified by a small subset of the data (i.e., the support vectors) that defines the position of the separator (the decision hyperplane).

  ![Margin Maximization Diagram]

  The support vectors are 5 points right up against the margin of the classifier.

  \[ \mathbf{w}^T \mathbf{x} + b \]

  Intercept term

  Normal (weight) vector of the hyperplane

- Maximization of the margin
  - If there are no points near the decision surface, then there are no very uncertain classification decisions.
  - Also, a slight error in measurement or a slight document variation will not cause a misclassification.
Formulation of SVM with Algebra (1/2)

- Assume here that data points are linearly separable
- Euclidean distance of a point to the decision boundary

1. The shortest distance between a point \( \bar{x} \) to a hyperplane is perpendicular to the plane, i.e., parallel to \( \overrightarrow{w} \)

2. The point on the plane closest to \( \bar{x} \) is \( \bar{x}' \)

\[
\bar{x}' = \bar{x} - yr \frac{\overrightarrow{w}}{|\overrightarrow{w}|}
\]

\[
\Rightarrow \overrightarrow{w}^T \left( \bar{x} - yr \frac{\overrightarrow{w}}{|\overrightarrow{w}|} \right) + b = 0
\]

\[
\Rightarrow r = \frac{y(\overrightarrow{w}^T \bar{x} + b)}{|\overrightarrow{w}|} \quad \text{or} \quad \left| \frac{\overrightarrow{w}^T \bar{x} + b}{|\overrightarrow{w}|} \right|
\]

3. We can scale \( y(\overrightarrow{w}^T \bar{x} + b) \), the so-called “functional margin”, as we please; for example, to 1

Therefore, the margin defined by the support vectors is expressed by

\[
\frac{2}{|\overrightarrow{w}|}
\]

(i.e., for support vectors \( y(\overrightarrow{w}^T \bar{x} + b) = 1 \); while for the others \( y(\overrightarrow{w}^T \bar{x} + b) \geq 1 \)
Formulation of SVM with Algebra (2/2)

- SVM is designed to find \( \vec{w} \) and \( b \) that can maximize the geometric margin
  \[
  - \frac{2}{|\vec{w}|} \quad \text{(maximization of \( \frac{2}{|\vec{w}|} \) is equivalent to minimization of \( \frac{1}{2} \vec{w}^T \vec{w} \))}
  \]
  - For all \( \{\vec{x}_i, y_i\} \in D, \ y_i \left( \vec{w}^T \vec{x}_i + b \right) \geq 1 \)

Mathematical formulation (assume linear separability)

- **Primal Problem**
  - Minimize \( L_p \) with respect to \( \vec{w} \) and \( b \)
  
  \[
  \begin{align*}
  \min & \quad \frac{1}{2} \vec{w}^T \vec{w} \\
  \text{subject to} & \quad y_i \left( \vec{w}^T \vec{x}_i + b \right) \geq 1, \forall \ i \\
  \end{align*}
  \]

  \[
  L_p = \left[ \frac{1}{2} \vec{w}^T \vec{w} - \sum_{i=1}^{N} \alpha_i \left[ y_i \left( \vec{w}^T \vec{x}_i + b \right) - 1 \right] \right] \quad \text{(\( \alpha_i \geq 0 \))}
  \]

  \[
  = \frac{1}{2} \vec{w}^T \vec{w} - \sum_{t=1}^{N} \alpha_i y_i \left( \vec{w}^T \vec{x}_i + b \right) + \sum_{t=1}^{N} \alpha_i
  \]

  \[\text{To be minimized} \quad \text{To be maximized} \]

  \[
  \frac{\partial L_p}{\partial \vec{w}} = 0 \Rightarrow \vec{w} = \sum_{t=1}^{N} \alpha_i y_i \vec{x}_i
  \]

  \[
  \frac{\partial L_p}{\partial b} = 0 \Rightarrow \sum_{t=1}^{N} \alpha_i y_i = 0
  \]

IR – Berlin Chen 14
Formulation of SVM with Algebra (3/3)

- **Dual problem** (plug 2 and 3 into 1)
  - Maximize $L_d$ with respect to $\alpha_i$

\[
L_d = \frac{1}{2} \bar{w}^T \bar{w} - \sum_{i=1}^{N} \alpha_i y_i (\bar{w}^T \bar{x}_i + b) + \sum_{i=1}^{N} \alpha_i
\]

\[
= \frac{1}{2} \bar{w}^T \bar{w} - \bar{w}^T \sum_{i=1}^{N} \alpha_i y_i \bar{x}_i - b \sum_{i=1}^{N} \alpha_i y_i + \sum_{i=1}^{N} \alpha_i
\]

\[
= -\frac{1}{2} \bar{w}^T \bar{w} + \sum_{i=1}^{N} \alpha_i
\]

\[
= -\frac{1}{2} \sum_{i=1}^{N} \sum_{j=1}^{N} \alpha_i \alpha_j y_i y_j \bar{x}_i^T \bar{x}_j + \sum_{i=1}^{N} \alpha_i
\]

Subject to the constraints that $\sum_{i=1}^{N} \alpha_i y_i = 0$ and $\alpha_i \geq 0 \ \forall i$

- Most $\alpha_i$ are 0 and only a small number have $\alpha_i > 0$ (they are support vectors)

- Have to do with the number of training instances, but not the input dimension

A convex quadratic-optimization problem
Dealing with Nonseparability (1/2)

- Datasets that are linearly separable (with some noise) work out great:

- But what are we going to do if the dataset is just too hard?

- How about mapping data to a higher-dimensional space?
Dealing with Nonseparability (2/2)

- General idea: The original feature space can always be mapped by a function \( \varphi(\cdot) \) to some higher-dimensional feature space where the training set is separable.

**Kernel trick**

\[
\Phi : \tilde{x} \rightarrow \varphi(\tilde{x})
\]

**Purposes:**
- Make non-separable problem separable
- Map data into better representational space
Kernel Trick (1/2)

• The SVM decision function for an input $\tilde{x}$ at a high-dimensional (the transformed ) space can be represented as

$$f(\tilde{x}) = \text{sign} \left( \tilde{w}^T \varphi(\tilde{x}) + b \right)$$

$$= \text{sign} \left( \sum_{i=1}^{N} \alpha_i y_i \varphi(\tilde{x}_i)^T \varphi(\tilde{x}) + b \right)$$

$$= \text{sign} \left( \sum_{i=1}^{N} \alpha_i y_i K(\tilde{x}_i, \tilde{x}) + b \right)$$

– A kernel function $K(\tilde{x}_i, \tilde{x})$ is introduced, defined by the inner (dot) product of points (vectors) in the high-dimensional space

• $K(\tilde{x}_i, \tilde{x})$ can be computed simply and efficiently in terms of the original data points

• We wouldn’t have to actually map from $\tilde{x} \rightarrow \varphi(\tilde{x})$
  (however, we still can directly compute $K(\tilde{x}_i, \tilde{x}) = \varphi(\tilde{x}_i)^T \varphi(\tilde{x})$)
Kernel Trick (2/2)

• Common Kernel Functions
  – Polynomials of degree $q$: $K(\tilde{u}, \tilde{v}) = (\tilde{u}^T \tilde{v} + 1)^q$
  – Polynomial of degree two (quadratic kernel)
    $$K(\tilde{u}, \tilde{v}) = (\tilde{u}^T \tilde{v} + 1)^2$$
    two-dimensional points
    $$= (u_1 v_1 + u_2 v_2 + 1)^2 \quad \text{(where } \tilde{u}^T = [u_1, u_2], \tilde{v}^T = [v_1, v_2])$$
    $$= 1 + 2u_1 v_1 + 2u_2 v_2 + 2u_1 u_2 v_1 v_2 + u_1^2 v_1^2 + u_2^2 v_2^2$$
  $$\phi(\tilde{u}) = [1, \sqrt{2}u_1, \sqrt{2}u_2, \sqrt{2}u_1 u_2, u_1^2, u_2^2]^T$$

  – Radial-basis function (Gaussian distribution): $K(\tilde{u}, \tilde{v}) = e^{-(\tilde{u} - \tilde{v})^2/(2\sigma^2)}$

  – Sigmoidal function: $K(\tilde{u}, \tilde{v}) = \tanh(2\tilde{u}^T \tilde{v} + 1)$

The above kernels are not always very useful in text classification!
Soft-Margin Hyperplane (1/2)

- Even for very high-dimensional problems, data points could be linearly inseparable
- We can instead look for the hyperplane that incurs the least error
  - Define slack variables \( \xi_i \geq 0 \) that store the variation from the margin for each data points

- Reformulation the optimization criterion with slack variables
  - Find \( \bar{\mathbf{w}}, b \), and \( \xi_i \geq 0 \) such that
  
  \[
  \frac{1}{2} \mathbf{w}^T \mathbf{w} + C \sum_{i=1}^{N} \xi_i
  \]

  is minimum

  - For all \( \{\bar{x}_i, y_i\} \in \mathcal{D}, y_i (\mathbf{w}^T \bar{x}_i + b) \geq 1 - \xi_i \)

\[
\hat{L}_p = \frac{1}{2} \mathbf{w}^T \mathbf{w} + C \sum_{i=1}^{N} \xi_i - \sum_{i=1}^{N} \alpha_i \left[ y_i (\mathbf{w}^T \bar{x}_i + b) - 1 + \xi_i \right] + \sum_{i=1}^{N} \mu_i \xi_i
\]
Soft-Margin Hyperplane (2/2)

– Dual Problem

\[
\hat{L}_d = \sum_{i=1}^{N} \alpha_i - \frac{1}{2} \sum_{i=1}^{N} \sum_{j=1}^{N} \alpha_i \alpha_j y_i y_j \bar{x}_i^T \bar{x}_j
\]

subject to \( \sum_{i=1}^{N} \alpha_i y_i = 0 \) and \( 0 \leq \alpha_i \leq C \ \forall \ i \)

• Neither slack variables \( \xi_i \) nor their Lagrange multipliers \( \mu_i \) appear in the dual problem!
• Again, \( \bar{x}_i \) with non-zero \( \alpha_i \) will be support vectors
• Solution to the dual problem is:

\[
\tilde{w} = \sum_{i=1}^{N} \alpha_i y_i \bar{x}_i
\]

\[
b = y_k (1 - \xi_k) - \tilde{w}^T \bar{x}_k \quad \text{for} \quad k = \arg \max_k \alpha_k
\]

– Parameter \( C \) can be viewed as a way to control overfitting – a regularization term

• The larger the value \( C \), the more we should pay attention to each individual data point
• The smaller the value \( C \), the more we can model the bulk of the data
Using SVM for Ad-Hoc Retrieval (1/2)

- For example, documents are simply represented by two-dimensional vectors $\psi(d_i, q)$ consisting of cosine score and term proximity.

$\textit{Figure 15.7}$ A collection of training examples. Each $R$ denotes a training example labeled relevant, while each $N$ is a training example labeled nonrelevant.
Using SVM for Ad-Hoc Retrieval (2/2)

- Examples: Nallapati, Discriminative Models for Information Retrieval, *SIGIR 2004*

  - Basic Features used in SVM

    | Feature | Feature |
    |---------|---------|
    | $\sum_{q_i \in Q \cap D} \log(c(q_i, D))$ | $\sum_{q_i \in Q \cap D}(log\left(\frac{C}{c(q_i, C)}\right))$ |
    | $\sum_{i=1}^{n}(1 + \frac{c(q_i, D)}{|D|})$ | $\sum_{i=1}^{n}\log(1 + \frac{c(q_i, D)}{|D|} \cdot idf(q_i))$ |
    | $\sum_{q_i \in Q \cap D}\log(idf(q_i))$ | $\sum_{i=1}^{n}\log(1 + \frac{c(q_i, D)}{|D|} \cdot \frac{|C|}{c(q_i, C)})$ |

- Compared with LM and ME (maximum entropy) models

<table>
<thead>
<tr>
<th>Train Test</th>
<th>Disks 1-2 (151-200)</th>
<th>Disk 3 (101-150)</th>
<th>Disks 4-5 (401-450)</th>
<th>WT2G (426-450)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Disks 1-2 (101-150)</td>
<td>LM ($\mu^* = 1900$)</td>
<td><strong>0.2561 (6.75e-3)</strong></td>
<td>0.1842</td>
<td>0.2377 (0.80)</td>
</tr>
<tr>
<td></td>
<td>SVM</td>
<td>0.2145</td>
<td>0.1877 (0.3)</td>
<td>0.2356</td>
</tr>
<tr>
<td></td>
<td>ME</td>
<td>0.1513</td>
<td>0.1240</td>
<td>0.1803</td>
</tr>
<tr>
<td>Disk 3 (51-100)</td>
<td>LM ($\mu^* = 500$)</td>
<td><strong>0.2605 (1.08e-4)</strong></td>
<td>0.1785 (0.11)</td>
<td>0.2503 (0.21)</td>
</tr>
<tr>
<td></td>
<td>SVM</td>
<td>0.2064</td>
<td>0.1728</td>
<td>0.2432</td>
</tr>
<tr>
<td></td>
<td>ME</td>
<td>0.1599</td>
<td>0.1221</td>
<td>0.1719</td>
</tr>
<tr>
<td>Disks 4-5 (301-350)</td>
<td>LM ($\mu^* = 450$)</td>
<td><strong>0.2592 (1.75e-4)</strong></td>
<td>0.1773 (7.9e-3)</td>
<td><strong>0.2516 (0.036)</strong></td>
</tr>
<tr>
<td></td>
<td>SVM</td>
<td>0.2078</td>
<td>0.1646</td>
<td>0.2355</td>
</tr>
<tr>
<td></td>
<td>ME</td>
<td>0.1413</td>
<td>0.0978</td>
<td>0.1403</td>
</tr>
<tr>
<td>WT2G (401-425)</td>
<td>LM ($\mu^* = 2400$)</td>
<td><strong>0.2524 (4.6e-3)</strong></td>
<td>0.1838 (0.08)</td>
<td><strong>0.2487 (0.046)</strong></td>
</tr>
<tr>
<td></td>
<td>SVM</td>
<td>0.2199</td>
<td>0.1744</td>
<td><strong>0.2487 (0.046)</strong></td>
</tr>
<tr>
<td></td>
<td>ME</td>
<td>0.1353</td>
<td>0.0969</td>
<td>0.1441</td>
</tr>
<tr>
<td>Best TREC runs (Site)</td>
<td>0.4226 (UMass)</td>
<td>N/A</td>
<td>0.3207 (Queen’s College)</td>
<td>N/A</td>
</tr>
</tbody>
</table>

Tested on 4 TREC collections
Ranking SVM (1/2)

- Construct an SVM that not only considers the relevance of documents to the a training query but also the order of each document pair on the ideal ranked list
  - First, construct a vector of features $\psi(d_i, q)$ for each document-query pair
  - Second, capture the relationship between each document pair by introducing a new vector representation $\phi(d_i, d_j, q)$ for each document pair
    \[
    \phi(d_i, d_j, q) = \psi(d_i, q) - \psi(d_j, q)
    \]
  - Third, if $d_i$ is more relevant than $d_j$ given $q$ (denoted $d_i \prec d_j$, i.e., $d_i$ should precede $d_j$ on the ranked list), then associate they with the label $y_{ijq} = +1$; otherwise, $y_{ijq} = -1$

Ranking SVM (2/2)

- Therefore, the above ranking task is formulated as:
  - Find $\mathbf{x}$, $b$, and $\xi_{ijq} \geq 0$ such that
    \[
    \frac{1}{2} \mathbf{w}^T \mathbf{w} + C \sum_{i,j,q} \xi_{i,j,q} \text{ is minimized}
    \]
  - For all $\{\phi(d_i, d_j, q): d_i < d_j\}$, $\mathbf{w}^T \phi(d_i, d_j, q) + b \geq 1 - \xi_{i,j,q}$
    (Note that $y_{ijq}$ are left out here. Why?)