SVM-based Modelling with Pairwise Transformation for Learning to Re-Rank

Jun Kubota, Teruhiko Unoki, and Kenji Umezawa

Photron Limited
{kubota,unoki,ume}@photron.co.jp

Abstract. We present a solution for learning to re-rank questions for Community Question Answering (cQA). The task is to rank the forum questions according to their relevance against the new question. We re-rank questions by using Support Vector Machine (SVM) with pairwise transformation and normalization methods. Our experiments show that the proposed method improves the NDCG@10 for learning to re-rank.

1 Introduction

Community Question Answering (cQA) web sites are used by a lot of people all over the world on a daily basis. One of the biggest problems is that we need to wait until the right answers are posted by other users. If the right answers of similar questions are automatically provided, we would not need to wait for answers. cQA Challenge hosted by ECML/PKDD 2016 is a challenge to explore the possibility of bringing such an automatic system into reality using machine learning techniques.

Several methods using machine learning have been developed and applied to learning to re-rank. Ranking SVM[1][2], RankBoost[3] and RankNet[4] are based on pairwise approach which solve re-ranking problems by predicting the relative ranking of a document pair. Methods based on listwise approach such as Listnet[5] and AdaRank[6] also have been proposed to improve re-ranking prediction results.

In this paper we propose a Support Vector Machine (SVM) based model with pairwise transformation and normalization methods for learning to re-rank. Our experiments show that pairwise transformation, raw feature normalization and inter-query normalization improve the prediction results. By using the proposed model, we won the first prize in both of the development stage and the test stage of cQA Discovery Challenge at ECML/PKDD 2016 conference.

2 Feature Selection

2.1 Features

The training dataset which provided by Discovery Challenge includes 64 features for
each new question and kernel matrices. We call the 64 features ‘raw features’ in this paper. The raw features are divided in three sub-groups as follows.

- 47 features evaluate various similarities between the new and its related forum question
- 4 features are scores related to the quality of the comments from the forum question thread with respect to the new question
- 13 features are scores related to the quality of the comments from the forum question thread with respect to both the new and the forum question

The kernel matrices contain the tree kernel similarities between the syntactic parse trees of the questions. Pre-computed kernel matrices were provided and we used the matrices for building our model.

Given two pairs of new and forum questions \( p_i=(u_i,q_i) \) and \( p_j=(u_j,q_j) \), the kernel matrices are computed by equation (1).

\[
K(p_i, p_j) = K(u_i, u_j) \cdot K(q_i, q_j) + K(u_i, q_j) \cdot K(q_i, u_j)
\]  

(1)

where

- \( K(u_i,u_j) \) : similarities between the new questions \( u_i \) and \( u_j \)
- \( K(q_i,q_j) \) : similarities between the forum questions \( q_i \) and \( q_j \)
- \( K(u_i,q_j) \) : similarities between the new questions \( u_i \) and the forum questions \( q_j \)
- \( K(q_i,u_j) \) : similarities between the forum questions \( q_j \) and the new questions \( u_j \)

2.2 Training/Development Data

The training dataset for the cQA Challenge contains 2669 questions and development dataset contains 500 questions. Each question has id which contains the id of the new question. Also, each question in training dataset is labeled as Perfect match, Relevant, or Irrelevant with respect to a new question.

The kernel matrices contain scores of similarities between all of the questions in training and development dataset and the size is 3169x3169.

2.3 Feature Selection

A feature selection is essential for building a model with generalization ability. As described above, two types of data, the raw features and the kernel matrices, were provided and each has different meaning. Thus we selected features independently from the raw features and the kernel matrices. First, features were selected using Algorithm 1 and then Algorithm 2 was applied to the selected features. In this paper, \text{thr}_{xy} \) represents the threshold in Algorithm 1 and \text{thr}_{xx} \) represents the threshold in Algorithm 2.
Algorithm 1 Feature Selection by coefficient of correlation between features and labels

Input:
\( X \) - features of training data
\( y \) - ground truth labels of training data
\((2:\text{Perfect match}, 1:\text{Relevant}, 0:\text{Irrelevant})\)

output: list of the selected feature indexes
1: for all \( x_i \in X \) do
2: \( c_y \leftarrow \text{coefficient of correlation between } x_i \text{ and } y \)
3: if \( c_y \geq \text{threshold} \) then // thr_xy
4:     list.append(i)
5: end if
6: end for
7: return list

Algorithm 2 Feature Selection by coefficient of correlation between features

Input:
\( X \) - features of training data
\( y \) - ground truth labels of training data
\((2:\text{Perfect match}, 1:\text{Relevant}, 0:\text{Irrelevant})\)

output: list of the selected feature indexes
1: for all \( x_i \in X \) do
2: \( p \leftarrow i \)
3: \( \epsilon_{\text{max}} \leftarrow \text{coefficient of correlation between } x_i \text{ and } y \)
4: for all \( x_j \in X \) do
5:     if \( j \leq i \) then
6:         continue
7:     end if
8:     \( \epsilon_x \leftarrow \text{coefficient of correlation between } x_i \text{ and } x_j \)
9:     if \( \epsilon_x \geq \text{threshold} \) then // thr_xx
10:        \( \epsilon_y \leftarrow \text{coefficient of correlation between } x_j \text{ and } y \)
11:        if \( \epsilon_y > \epsilon_{\text{max}} \) then
12:            \( p \leftarrow j \)
13:            \( \epsilon_{\text{max}} \leftarrow \epsilon_y \)
14:        end if
15:     end if
16: end for
17: list.append(p)
18: end for
19: return list
Prediction Model

3.1 Pointwise Approach

SVM for binary classification is one of the commonly used learning methods for classification. Fig.1 is a conceptual diagram of the method in case the number of the features is two, feature1 and feature2.

One-vs-the-rest strategy is often applied for multi-class classification. The probability of each class membership is calculated and each sample is classified with reference to the highest probability. However, multi-class classification models are non-matching with learning to rank. Thus we tested only SVM for binary classification for cQA Challenge.

One of the problems with pointwise approach is that cannot predict well in case where feature distribution vary across the queries as shown in Fig.2.

3.2 Pairwise Approach

The prediction performance of pairwise approach models are more accurate than pointwise approach models in general. Selected features are transformed to create new features by pairwise transformation described in Algorithm 3.
Algorithm 3 Pairwise Transformation

**Input:**
- $X$ - features of training data
- $y$ - ground truth labels of training data
  - (2:Perfect match, 1:Relevant, 0:Irrelevant)
- $Q$ - list of indexes related to the same query

**output:** pairwise transformed features

1: for all $i \in Q$ and $j \in Q$ do
2:   if $y_i > y_j$ then
3:     $X_{new}$·append($x_i - x_j$)
4:     $y_{new}$·append(1)
5:     $X_{new}$·append($x_j - x_i$)
6:     $y_{new}$·append(-1)
7:   else if $y_j > y_i$ then
8:     $X_{new}$·append($x_j - x_i$)
9:     $y_{new}$·append(1)
10:    $X_{new}$·append($x_i - x_j$)
11:   $y_{new}$·append(-1)
12: end if
13:end for
14:return $X_{new}$, $y_{new}$

After the pairwise transformation, the example data shown in Fig.2 is transformed as Fig.3. Our proposed model is based on SVM (rbf kernel) for binary classification with the pairwise transformation.

![Fig. 3. SVM(linear kernel) for binary classification with pairwise transformation](image)

### 3.3 Listwise Approach

The listwise approach models such as Listnet and AdaNet are reported to have better performance than pointwise and pairwise approach in many case [14]. Due to our limited resources of time and manpower, we just tried using open-source library and tested these method for comparison.
3.4 **Raw Data Normalization**

The features selected from the raw features were normalized feature-by-feature as equation (2).

\[ x_{\text{normalized}} = \frac{x - \text{mean}(x)}{\text{max}(x) - \text{min}(x)} \] (2)

:where
- \( \text{max}(x) \) : maximum value of each feature in the training data and the test data
- \( \text{min}(x) \) : minimum value of each feature in the training data and the test data
- \( \text{mean}(x) \) : mean value of each feature in the training data and the test data

3.5 **Inter-Query Normalization**

The features generated by pairwise transformation were normalized in each query for absorbing difference of distribution for each query.

\[ x_{\text{normalized}} = \frac{x}{\text{max}(\text{norm}(x))} \] (3)

:where
- \( \text{max}(x) \) : maximum value in a query
- \( \text{norm}(x) \) : scalar value which represents a length of a feature vector

3.6 **Proposed Model**

We propose the following model by the result of cross validation for each prediction model. Parameters for each process were decided by the result of grid search analysis using cross validation. The repeat count for the cross validation was 8.

<table>
<thead>
<tr>
<th>item name</th>
<th>setting / value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Raw Features</td>
<td>23 features selected</td>
</tr>
<tr>
<td>Kernel Matrices</td>
<td>53 features selected</td>
</tr>
<tr>
<td>Raw Data Normalization</td>
<td>ON</td>
</tr>
<tr>
<td>Pairwise Transformation</td>
<td>ON</td>
</tr>
<tr>
<td>Inter-query Normalization</td>
<td>ON</td>
</tr>
<tr>
<td>Learning Method</td>
<td>SVM for classification (SVC)</td>
</tr>
<tr>
<td>- kernel</td>
<td>rbf</td>
</tr>
<tr>
<td>- gamma</td>
<td>0.01</td>
</tr>
<tr>
<td>- C</td>
<td>3.0</td>
</tr>
<tr>
<td>- class weight</td>
<td>balanced</td>
</tr>
</tbody>
</table>

'class weight' (class_weight) is one of the options of SVC function included in scikit-learn machine learning package for python. Weight of C for each class is adjusted inversely proportional to class frequencies in the input data by ‘balanced’ setting.
4 Experiments and Results

4.1 Data Split and Cross Validation

We split the training data in 60% for training and 40% for validation. The split was processed based on the query id. As a result, all of the samples related to the same query was always divided to the same side.

We chose the number 8 as the repeat count for cross validation to get stable evaluation value. For the eQA Challenge, the split percentage was fixed independently of the repeat count for cross validation in contrast to k-fold cross validation because the number of the validation data is not enough when k is set to 8. In each cross validation series, a random seed for data split was changed in order to split differently.

4.2 Validation

A prediction model built by using a training data was applied to a validation data. The result was evaluated by measuring MAP, NDCG@3, NDCG@5 and NDCG@10 calculated by the result from the validation data. We referred mainly NDCG@10 for validation as the stable and representative measure. The degree of overfitting was evaluated by calculating the degradation of NDCG@10 as equation (4). We also used NDCG@10 with overfitting correction calculated by equation (5) as a reference.

\[
d_{\text{NDCG}} = \frac{\text{NDCG}_{\text{training}} - \text{NDCG}_{\text{validation}}}{\text{NDCG}_{\text{validation}}} \tag{4}
\]

\[
\text{NDCG}_{\text{corrected}} = \text{NDCG} \cdot (1 - d_{\text{NDCG}}) \tag{5}
\]

4.3 Feature Selection

Feature selection process is controlled by Thresholds of coefficient of correlation thr_xy and thr_xx described in section 2.3. We fixed thr_xx to 0.80 with our time and manpower in mind.

SVM (rbf kernel) for binary classification with pairwise transformation and both of normalization methods were used to determine the best parameters for feature selection. Evaluation was based on NDCG@10. Gamma ranged in value from 0.05 to 0.15 and C ranged in value from 0.3 to 0.5.

First of all, thr_xy was varied from 0.08 to 0.20 and the same thr_xy was used for the raw features and kernel matrices. The result of NDCG@10 and overfitting measure are shown in Fig.4 and Fig.5. Each plot represents the mean of the result of cross validation using a parameter pair (Gamma, C). NDCG@10 became maximum when (thr_xy, thr_xx) equaled to (0.12, 0.80) and the result of overfitting was also better than others. The number of selected features from the raw features was around 20 and the number varied with in a relatively narrow range. Thus it’s fairly enough to determine that (thr_xy, thr_xx) = (0.12, 0.80) is the best parameter pair for selecting features from the raw features. 23 features was selected from the raw features with the parameters.
Fig. 4. feature selection (coarse) : the result of NDCG@10

Fig. 5. feature selection (coarse) : the result of overfitting measure

where rf and tk in the legends in Fig.4 and Fig.5 represent as follows.

rf : the number of features selected from the raw features
tk : the number of features selected from the tree kernel matrices

Secondly, we varied thr_xy for selecting features from the kernel matrices range in value from 0.06 to 0.23 in the condition of using the best parameters for selecting features from the raw features. The result is shown in Fig.6 and Fig.7. The best NDCG@10 was obtained when (thr_xy, thr_xx) equaled to (0.13, 0.80) for selecting features from the kernel matrices. There was no difference in regard to overfitting measure. The number of selected features from the kernel matrices was 58 with the best parameters.
Finally, we decided to use 23 features from the raw features and 58 features from the kernel matrices. At the development stage of the cQA challenge we used 22 features from the raw features and 42 features from the kernel matrices because it was during the analysis.

### 4.4 Data Preprocessing

We propose to use both of raw data normalization and inter-query normalization. The effect of these normalization was evaluated as follows. We tested the normalization methods in combination with two kernels for SVM. As a result, the normalization methods were effective for both of the kernels and ‘rbf’ kernel was better than ‘linear’ kernel with the normalization methods. The number of selected features were the
same as proposed model. A parameter $C$ for SVM with linear/rbf kernel varied in range from 0.3 to 0.5. A parameter gamma for SVM with rbf kernel varied in range from 0.05 to 0.15.

**Fig. 8.** Normalization: the result of NDCG@10

where the legends in Fig.8 and Fig.9 represent as follows.
- linear: SVM with linear kernel
- rbf: SVM with rbf kernel
- FF: no normalization used
- TF: raw data normalization used
- FT: inter-query normalization used
- TT: both of the normalization used

**Fig. 9.** Normalization: the result of NDCG@10 with overfitting correction
4.5 Parameters for SVM

Parameters for SVM are gamma and C. Gamma is a kernel coefficient of rbf kernel. C is a penalty parameter for the error term. The best parameters were determined by grid search with cross validation. The repetition number of the cross validation was 8. The result of coarse grid search is shown in Fig.10 and Fig.11.

The best score was obtained by parameter pair (gamma, C) equaled to (0.01, 3.0) by the coarse grid search. Then, a fine grid search was conducted around the parameter pair. The result of the fine grid search is shown in Fig.12 and Fig.13.
As a result, the best parameter pair was \((\gamma, C) = (0.01, 3.0)\) determined by the result of NDCG@10 with overfitting correction. The result of NDCG@10 without the correction was 0.8446 and overfitting measure was 0.0077.

### 4.6 Result of cQA Challenge

#### Development Stage

We submitted the best result for each model during the development phase. The result was shown in Table 2. ‘Normalization’ means using both of the normalization options (raw data normalization and inter-query normalization). MAP was the score obtained by online submission through the cQA Challenge web page. These results were obtained with different features and parameters because they were submitted in the process of analysis and development.
Table 2. The best online submission result for each model in the development stage

<table>
<thead>
<tr>
<th>MAP</th>
<th>Model</th>
<th>Approach</th>
<th>Raw Features</th>
<th>Kernel Matrices</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.7536</td>
<td>Normalization + Pairwise Transformation + SVM(rbf)</td>
<td>pairwise</td>
<td>22</td>
<td>42</td>
</tr>
<tr>
<td>0.7476</td>
<td>Normalization + Pairwise Transformation + SVM(rbf)</td>
<td>pairwise</td>
<td>19</td>
<td>30</td>
</tr>
<tr>
<td>0.7460</td>
<td>SVM(rbf)</td>
<td>pointwise</td>
<td>19</td>
<td>30</td>
</tr>
<tr>
<td>0.7446</td>
<td>Pairwise Transformation + SVM(rbf)</td>
<td>pairwise</td>
<td>19</td>
<td>30</td>
</tr>
<tr>
<td>0.7392</td>
<td>ECML/PKDD2016 Discovery Challenge Base Line</td>
<td>pairwise</td>
<td>19</td>
<td>30</td>
</tr>
<tr>
<td>0.7387</td>
<td>Pairwise Transformation + SVM(linear)</td>
<td>pairwise</td>
<td>19</td>
<td>30</td>
</tr>
<tr>
<td>0.7169</td>
<td>Listnet</td>
<td>listwise</td>
<td>19</td>
<td>30</td>
</tr>
<tr>
<td>0.7162</td>
<td>SVM(rbf)</td>
<td>pointwise</td>
<td>64</td>
<td>0</td>
</tr>
<tr>
<td>0.7145</td>
<td>AdaRank</td>
<td>listwise</td>
<td>19</td>
<td>30</td>
</tr>
<tr>
<td>0.6922</td>
<td>SVM(rbf)</td>
<td>pointwise</td>
<td>64</td>
<td>3169</td>
</tr>
</tbody>
</table>

Generally models using listwise approach predict better than pairwise approach models. However the result was worse than our model, because we just used RankLib[15] without any modification and optimization. The result scored 0.7446 by ‘Pairwise Transformation + SVM(rbf)’ is supposed to be higher than 0.7460 with appropriate parameters while it would be lower than the result using ‘Normalization’. The model which obtained the best score 0.7536 was the same model as we propose while the number of selected features and the parameters for SVM (gamma=0.1, C=0.38) were different because we haven’t finished analysis for searching the best parameters described in this paper in the development stage. However it comes to be considered that the submitted result is close to the result with the best parameters referring to the following local evaluation result of training dataset calculated by the evaluator provided by cQA Challenge.

Table 3. Evaluation of the submitted model and the proposed model in the development stage

<table>
<thead>
<tr>
<th>local evaluation</th>
<th>online submission</th>
</tr>
</thead>
<tbody>
<tr>
<td>MAP</td>
<td>MRR</td>
</tr>
<tr>
<td>the best submitted model</td>
<td>0.7047</td>
</tr>
<tr>
<td>proposed model</td>
<td>0.7054</td>
</tr>
</tbody>
</table>

Test Stage
The provided test data was processed using the proposed model and the best parameters described in section 3.6. The result of local evaluation was the same as ‘proposed model’ shown in Table.3. The result MAP of online submission was 0.7714.

5 Conclusion
The proposed model is simple and valid way to re-rank questions. Feature selection was one of the important factors for better prediction. Pairwise approach was effective for learning to rank while we need to evaluate listwise approach as a future work. Practically we could build a reasonable model using general machine learning techniques without the knowledge of Natural Language Processing and the meaning of Tree Kernel Matrices. That is one of the great advantages to use machine learning to solve such problems as learning to re-rank.
References

2. T. Joachims. Optimizing Search Engines Using Clickthrough Data, Proceedings of the ACM Conference on Knowledge Discovery and Data Mining (KDD), ACM, 2002
5. Z. Cao, T. Qin, T.-Y. Liu, M.-F Tsai, and H. Li. Learning to Rank: From Pairwise Approach to Listwise Approach. ICML 2007
15. RankLib: https://people.cs.umass.edu/~vdang/ranklib.html