Social Media Comment Management using SMOTE and Random Forest Algorithms

Nuanwan Soonthornphisaj*, Taratep Sira-Aksorn, Pornchanok Suksankawanich

Department of Computer Science, Faculty of Science, Kasetsart University, Bangkok, Thailand

ABSTRACT

Comment posting is one of the main features found in social media. Comment responses enrich the social network function especially for how-to content. In this work, we focus on cooking video clips which are popular among users. Various questions found in comments need to be clustered in order to facilitate the clip owners to effectively provide responses for those viewers. We applied machine learning algorithms to learn and classified comments into predefined classes. Then the density-based clustering algorithm, for density-based spatial clustering of applications with noise, is applied to cluster the content of Comment. The experimental result show that using Random Forest with SMOTE provides the best performance. We got 95% of the average performance measured in term of $F_1$-measure. Furthermore, we implement the incremental learning system via an online application that can automatically retrieve and organize video clip's comment into categories.

1. INTRODUCTION

Food and recipes are one of the preferred content categories found in YouTube channels. Most of user’s comments for how-to clips are in question forms related to Ingredients, Cooking tools and techniques. Even though the platform allows the communication among viewers but the comments has not yet been organized. The clip owner needs to interact with those comments however there are quite a number of comments asking the same questions whilst the owner must take his time to keep answering these comments. If we can manage the comment in such a way that the same category of question is grouped with the response. It will also benefit other viewers to explore frequently asked question based on category.

In this work, we collect posted comments from cooking video clips using YouTube API then we use ensemble learning method to learn from the training set and classify the comments into predefined class. The comment can be categorized into four classes which are: (1) How-to, (2) Ingredients, (3) Cooking tools and (4) Others. Since the dataset is imbalanced therefore, the imbalance class problem is solved by oversampling method.

The rest of paper is organized as follows. Section 2 gives the brief background of algorithms and related works. Section 3 provides the detail of our proposed method. Then all Experimental Results are shown in Section 4. Section 5 shows detail about application and the conclusion is given in the final section.

2. BACKGROUND

2.1. Random Forest

Our research deals with text classification problem where a dataset of comments must be categorized into predefined classes. Feature set in this domain is a word occurrence that lead to dimensionality problem. Therefore an ensemble decision tree learning method so called Random Forest is supposed to be an appropriated method since it produce diversity by training on different subsets of the data and feature sets. In machine learning point of view, diversity is the key to construct good ensembles.

Random Forests is introduced by Breiman [1], where he combines Bagging with random feature selection for decision trees. Each learner of the ensemble is trained on a bootstrap replicate as in Bagging. Decision trees are then grown by selecting the feature to split on at each node from $F$ randomly selected features. After generating large number of trees, the final classification result is obtained from the majority vote. We call these procedures as Random Forests (see Table 1).

Dietterich [2] recommends Random Forests as the method of choice for decision trees, as it compares favorably to AdaBoost [3] and works well even with the presence of noise. His empirical research shows that AdaBoost is not a good choice in the noisy domain since it iteratively increases the weights on the instances most recently misclassified. Instances having incorrect class labels will persist in being misclassified. Then, AdaBoost will concentrate increasing weight on these noisy instances and become warped [1]. The Random Forest procedures do not concentrate weight on any
Table 1  Random Forest algorithm

<table>
<thead>
<tr>
<th>Algorithm: Random Forest</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Input:</strong> A training set $S = {(x_1, y_1), \ldots, (x_n, y_n)}$</td>
</tr>
<tr>
<td>Features set, $F$</td>
</tr>
<tr>
<td>Number of trees in forest, $B$</td>
</tr>
</tbody>
</table>

**Output:** Hypothesis

Begin

$H \leftarrow \emptyset$

$\text{for } i = 1, \ldots, B \text{ do}$

$S' \leftarrow \text{A bootstrap sample from } S$

$h_i \leftarrow \text{Randomized Tree Learn}(S', F)$

$H \leftarrow H \cup \{h_i\}$

endfor

return $H$

end

Function: Randomized Tree Learn($S, F$)

At each node:

- $f$ very small subset of $F$
- Split on best feature in $f$
- return The learned tree

end function

Table 2  Comment Classification algorithm

<table>
<thead>
<tr>
<th>Algorithm: Comment Classification</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Input:</strong> dataset</td>
</tr>
</tbody>
</table>

**Begin**

for each instance in dataset do

- comment = preprocessing()
- vector: = featureExtract(comment)
- add vector and label to vectors

endfor

model: = Random Forest with SMOTE (SMOTE(trainset))

result: = model.predict(testset)

end

2.2. SMOTE

We found that our data set is imbalanced in terms of number of instances in each class see Table 5. The imbalanced data set affects the classification performance. The classification on imbalanced data causes problems because learning algorithm is sensitive to the number of training instances. The classification performance of majority class is higher than that of minority class. The problem is that the majority class is not the target class to be learned by the algorithm. In this research, class Others contains general comments such as admiring comments or comments that need no feedback from the clip owner. This class has higher number of comments than the target classes (How-to, Ingredients, Tools). Therefore we need the algorithm to manipulate the training set by increasing the size of the minority class either duplicates or interpolates minority instances. One of the famous oversampling method is the SMOTE algorithm [5]. SMOTE increases the number of minority class instances using interpolation method. The algorithm starts from searching for the $k$-nearest neighbors for each minority instance, then for each neighbor, it randomly selects a point from the line connecting the neighbor and the instance itself. This algorithm is able to generate synthetic instances rather than duplicate minority class instances; therefore, the overfitting problem can be avoided (see Table 3 for detail).

2.3. Density-Based Spatial Clustering of Applications with Noise

DBSCAN stands for density-based spatial clustering of applications with noise (Table 4). It finds core instance of high density and expands the cluster. The advantage of DBSCAN is that it does not require the number of clusters as a parameter. The algorithm was initially deployed to solve spatial data [6], however it was applied to solve many textual datasets such as Web page [7] and Newsgroup [8].

Table 3  SMOTE algorithm

<table>
<thead>
<tr>
<th>Algorithm: SMOTE</th>
</tr>
</thead>
</table>
| **Input:** Number of minority class samples, $T$
- Amount of SMOTE, $P$
- Number of nearest neighbors, $k$ |

**Output:** $(N/100) \times T$ synthetic minority class samples

// If $N$ is less than 100%, randomize the minority class samples as only a random percent of them will be SMOTE

**Begin**

if $N < 100$ then

- Randomize the $T$ minority class samples
- $T = (N/100) \times T$
- $N = 100$

endif

$N = (\text{int})(N/100)$ // The amount of SMOTE is assumed to be in integer multiples of 100.

$k =$ Number of nearest neighbors

$\text{numattrs} =$ Number of attributes

Sample[ ] [ ]: Array for original minority class samples

newindex: Keeps a count of number of synthetic samples generated, initialized to 0

Synthetic[ ] [ ]: Array for synthetic samples

// Compute $k$-nearest neighbors for each minority instance only

for $i \leftarrow 1$ to $T$

- Compute $k$-nearest neighbors for $i$, and save the indices in the nnarray
- Populate($N, i, \text{nnarray}$)

endfor

end

//Function to generate the synthetic samples//

while $N \neq 0$

- $\text{nn} = \text{Random}()$

  for $\text{attr} \leftarrow 1$ to $\text{numattrs}$

    $\text{dif} = \text{Sample[nnarray[\text{nn}]].[attr]} - \text{Sample[i].[attr]}$

    $\text{gap} = \text{random number between 0 and 1}$

    $\text{Synthetic[newindex].[attr]} = \text{Sample[i].[attr]} + \text{gap} \times \text{dif}$

  endfor

  newindex++

  $N = N - 1$

endwhile

return //End of Populate//
Two parameters is needed in DBSCAN which are $\varepsilon$ and MinPts. For each point in dataset $D$, the $\varepsilon$ neighborhood of point $p$ is shown in Equation (1). The $\varepsilon$ neighborhood of each point in dataset $D$ is processed. While the distance between $p$ and $q$ is calculated, Euclidean distance formula is used as given in Equation (2), where $m$ is the dimension of the points. Then, the distance between any two points is determined and compared whether it is smaller than $\varepsilon$ or not. The MinPts is used to check whether the point $p$ is qualified to be the core point which means that the number of $P'$ neighbors are greater or equal to MinPts [Equation (3)]. All density connected points form a cluster.

$$N_e(p) = \{ q \in D | d(p, q) < \varepsilon \}$$  

(1)

$$d(p, q) = \sqrt{\sum_{i=1}^{m} (p_i - q_i)^2}$$  

(2)

$$|N_e(p)| \geq \text{MinPts}$$  

(3)

Density-based spatial clustering of applications with noise (Table 4) is applied in this work as the second layer process. After the comment classification is done, all comments in each class are further clustered in order to organize similar content into hierarchies for user.

3. FRAMEWORK

The framework of the comment management system (Fig. 2) consists of the feature extraction process, the classification task that applies SMOTE with Random Forest and the comment clustering. After the stopword removal and word stemming are completed, the feature selection is performed by chi-square test ($\chi^2$) [Equation (4)]. The objective of using chi-square is to remove words that are unlikely to be significant. By calculating the chi-square scores for all words, we can rank the words by the chi-square scores, then choose the top ranked words for model training. This yield a dataset containing 800 word features that are further created as a feature vectors in the training set. Note that the higher value of the chi-square score, the more likelihood the feature is correlated with the class, thus it should be included for model training.

$$\chi^2(t,c) = \frac{\left( N \cdot (AD-CB) \right)^2 }{ \left( (A+C) \cdot (B+D) \cdot (A+B) \cdot (C+D) \right) }$$  

(4)

$A =$ Number of comments in class $c$ containing word $t$,
$B =$ Number of comments in other classes (except $c$) containing word $t$,
$C =$ Number of comments in class $c$ that do not contain word $t$,
$D =$ Number of comments in other classes (except $c$) that do not contain word $t$,
$N =$ Total number of comments.

Given the feature vectors attached with the class label, the oversampling is performed on the minority classes using SMOTE. Then the ensemble learning algorithm, Random Forest starts learning from the training set. After the comment classification is finished, all comment in each class are further clustered in order to organize the comment into hierarchical structure (see Fig. 1). Note that comments that are classified as class Others are filtered out.

![Figure 1](https://example.com/hierarchical.png)

**Figure 1** Hierarchical structure obtained from the comment management system

<table>
<thead>
<tr>
<th>Class</th>
<th>Number of instances</th>
</tr>
</thead>
<tbody>
<tr>
<td>How-to</td>
<td>925</td>
</tr>
<tr>
<td>Ingredients</td>
<td>1,169</td>
</tr>
<tr>
<td>Cooking tools</td>
<td>301</td>
</tr>
<tr>
<td>Others</td>
<td>13,882</td>
</tr>
</tbody>
</table>

4. EXPERIMENTAL RESULTS

We retrieve comment posts on cooking video clip via website YouTube. There are 16,277 comments written in Thai language each of which is labeled into four classes. Class How-to consists of content regarding cooking methods or techniques whereas
comments of class Cooking tools mainly concern about cooking utensils. Class Ingredients focus on cooking recipes but class Others are miscellaneous contents such as admiring comment. Since Thai language has no word boundary, therefore Thai word segmentation tool namely Lexto that uses longest matching method is applied in the preprocessing step. The tool use Lexitron as a corpus. Stopword removal and word stemming are done as the preprocessing task.

To explore the learning algorithm’s performance, two machine learning algorithm are evaluated, Random Forest, Naïve Bayes. The imbalanced problem is taken into consideration using synthetic minority class oversampling method namely SMOTE. We measure performance using precision, recall and \( F_1 \) (Equation 5–7), all experiments are performed based on 10-fold cross-validation.

\[
\text{Precision} = \frac{\text{True positive}}{\text{True positive} + \text{False positive}} \quad (5)
\]
\[
\text{Recall} = \frac{\text{True positive}}{\text{True positive} + \text{False negative}} \quad (6)
\]
\[
F_1 = \frac{2 \cdot \text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}} \quad (7)
\]

The experimental result show that applying SMOTE with Random Forest provides the best performance compared to other algorithms (see Fig. 3 and Table 6). We found that \( F_1 \) score obtained from SMOTE + RF is 0.87 for Ingredients and Cooking tools class. For How-to class, \( F_1 \) score obtained from SMOTE + RF is 0.72 which is higher than that of SMOTE + NB and NB algorithm. We found that for imbalanced dataset using synthetic oversampling method with ensemble can enhance the classification performance. SMOTE + NB cannot improve the precision value for every class.

We found that NB performs better than SMOTE + NB since NB considers all features to calculate the conditional probability for each given class label. Although SMOTE algorithm increases the number of instances in minority classes but their feature values are synthesis to form the new instances. All of these feature values \( (a) \) are needed for conditional probability calculation that might affect classification of NB. Note that NB classify the instance using Equation (8).

\[
V_{NB} = \arg \max P(C_j) \cdot \prod_{k=1}^{n} P(a_k|c_j) 
\]

We know that Random Forest algorithm creates multiple learning models by randomly selects feature subset to create different decision trees and combines the final decision using majority vote.
The advantage of ensembles with respect to single models show the robustness since there is no side effect on synthetic instances created by SMOTE.

5. APPLICATION

Our application has three main features as follows:

1. Clip video feeder obtains user query to retrieve clip video and comments via YouTube API (see Figs. 4 and 5).
2. Comment management system classifies all comments into four classes (see Fig. 6).
3. The incremental learning facilitate administrator to update model’s knowledge by adding new training set or relabel the misclassification instances (see Fig. 7).

Figure 7 shows the output of incremental learning feature that allows user to update new instances into the training set. User can do class labelling via the system and see the performance of the new model (see Fig. 8).
6. CONCLUSION

In this work, we have studied and applied machine learning algorithms for the comment post in social media. The imbalanced problem is solved using SMOTE algorithm. There is no effect from minority class instances oversampling since the ensemble create multiple learning model using different feature sets.

The framework is implemented as the comment management system based on cooking video clips. The system provides options for the user to update the learning model in incremental style.

REFERENCES