Novel Classes Detection
CS 534 Final Project

Members:
Xingyi Li
Jun He
Tanjin Xu
• Problem
  • It is hard to predict the environment, a test instance may come from a class that is different from all training classes
  • How to identify those instances?

• Objective
  • Design a classification algorithm to detect instances from unseen classes
    • Input
      • Training instances from known classes
    • Output for each test instance
      • Label each test instance as novel (from a novel class) or non-novel

Source: Learning with Augmented Classes by Exploiting Unlabeled Data, Da Q et al., AAAI 2014
Overview:

Binary Classifier (SVM, One-class SVM, Twin SVM) → Multi-class Classifier → Novel Class Classifier (output: novel or non-novel)
Multi-Class Classification Strategy:

- For K classes, build K-1 binary classifiers, \( C_i, i = 1, 2, \ldots, K-1 \)
- Each classifier \( C_i \) separates class \( i \) from the rest, for \( i = 1, \ldots, K-1 \)
- If all classifiers say no, then predict the instance as class K
General Novel-class Classification Strategy

Training Classes: \{1, 2, 3\}

Train K = 3 Classifiers instead of K-1:

1
   +1:1
   -1:2,3

2
   +1:2
   -1:1,3

3
   +1:3
   -1:1,2

Results:

Non-novel

Novel

Test instances
SVM

- A powerful paradigm for pattern classification and regression

- Strategies
  - Maximizing the margin between two disjoint half planes
  - Minimization of a convex quadratic function subject to linear inequality constraints
  - Efficient QP algorithms to solve the dual problem
  - Kernel functions can be used in SVM for non-linear classifications

source: wikipedia.org
One-class SVM

• Treat the whole training data as one class, which is called the normal class
• An outlier detector that determines if the test instance is similar to the training set or not.

Source: scikit-learn.org
Twin Support Vector Machines (TWSVM):

- Binary classifier
- Seeks for two non-parallel planes, one for each class – the plane that belongs to a certain class should be closer to that class and be as far as possible from another class at the same time.
- Optimize two sub objective functions (run faster than SVM O$(m^3)$):

\[
(TWSVM1) \quad \min_{w^{(1)}, b^{(1)}, q} \quad \frac{1}{2} (A w^{(1)} + c_1 b^{(1)})^T (A w^{(1)} + c_1 b^{(1)}) + c_1 c_2^T q \\
\text{subject to} \quad -(B w^{(1)} + c_2 b^{(1)}) + q \geq c_2, \quad q \geq 0
\]

\[
(TWSVM2) \quad \min_{w^{(2)}, b^{(2)}, q} \quad \frac{1}{2} (B w^{(2)} + c_2 b^{(2)})^T (B w^{(2)} + c_2 b^{(2)}) + c_2 c_1^T q \\
\text{subject to} \quad (A w^{(2)} + c_1 b^{(2)}) + q \geq c_1, \quad q \geq 0,
\]

- Decision for testing: compare the distance from the test instance to each hyper plane – the shorter one wins
Twin Support Vector Machines (TWSVM):
TWSVM vs. SVM

- Standard SVM seeks for a maximum margin, it does not care about the distribution of the instances for each class, while TWSVM always seek a hyper plane that “close” (not necessary cross) to the centroid of a class.
- TWSVM has statistical advantages: it is sensitive to the distribution of the instances from a particular class. Refer to the picture below, adding more positive training instances into region “A” will move the corresponding hyper plane towards to region “A”.

\[ w^T x + b = 1 \]
TWSVM to Novel Classes Detection
(Gaussian Assumption)

Hyper plane for negative class

Hyper plane for positive class

$\mu_-, \sigma_-$

$\mu_+, \sigma_+$

$D_\rightarrow ? D_+$

Even if $D_+$ is shorter,
How about $P(\,? \in \text{positive class})$

$D_-$ $D_+$

$?$
After one classifier thought a test instance belongs to positive class, calculate the distance from the instance to its hyper plane, and compare with the corresponding Gaussian model, find $P(\text{test instance } \in \text{ positive class})$. If above the threshold - pass, reject otherwise.
Parameter Tuning

• Common purpose: classification accuracy

• Our purpose: novel detection accuracy
Parameter Tuning

• TWSVM has 2 cost parameters: C₁, C₂
• Control C₁ and C₂ to shift the decision boundary.
  • For Linear kernel, fix C₁, decreasing C₂ will move the boundary towards positive class, and vice versa.
Experiment Set Up

- UCI Letter Recognition dataset - 26 classes in total
- MNIST handwriting digits dataset - 10 classes in total

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Training Set</th>
<th>Testing Set</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>#Train Data</td>
<td>#Train Classes</td>
</tr>
<tr>
<td>MNIST</td>
<td>500</td>
<td>5</td>
</tr>
<tr>
<td>UCI LR</td>
<td>500</td>
<td>5</td>
</tr>
</tbody>
</table>

- Three novel classes classifiers to be compared
  - OVR-SVM (based on SVM)
  - MOC-SVM (based on One-class SVM)
  - NCD-TWSVM (based on TWSVM)
Performance Measurement

<table>
<thead>
<tr>
<th>True Novel</th>
<th>True Non-novel</th>
</tr>
</thead>
<tbody>
<tr>
<td>Predict Novel</td>
<td>N1</td>
</tr>
<tr>
<td>Predict Non-novel</td>
<td>N3</td>
</tr>
</tbody>
</table>

- Precision = $\frac{N_1}{N_1 + N_2}$ (0 ≤ Precision ≤ 1)
- Recall = $\frac{N_1}{N_1 + N_3}$ (0 ≤ Recall ≤ 1)
- Good performance means both precision and recall should be high at the same time. For convenience, we introduce F1 score:
  - $F_1 = 2 \times \frac{\text{precision} \times \text{recall}}{\text{precision} + \text{recall}}$ (0 ≤ $F_1$ ≤ 1, the larger, the better)
Experiment Results (Gaussian Kernel)

- Test result: Based on F1 Value, NCD-TWSVM outperforms two other existing algorithms, which are OVR-SVM and MOC-SVM.

Results for UCI Letter Recognition dataset
Experiment Results (Gaussian Kernel)

- Test result: Based on F1 Value, NCD-TWSVM outperforms two other existing algorithms, which are OVR-SVM and MOC-SVM.
Parameter Tuning Experiment

- Fix Gaussian kernel parameter $\sigma$ and $C_1$ (for data set MNIST, seen classes are $\{1, 2, 3, 4, 5\}$)
- Decrease $C_2$ ($C_2 = C_1/2^n$) to evaluate the performance