Transfer Learning to Adapt One Class SVM Detection to Additional Features

Yongjian Xue and Pierre Beauseroy
Institut Charles Delaunay/LM2S, UMR CNRS 6281, Université de Champagne, Université de Technologie de Troyes, 12, rue Marie Curie CS 42060 - 10004, Troyes Cedex, France

Keywords: Transfer Learning, Multi-task Learning, Outliers Detection, One Class Classification.

Abstract: In this paper, we use the multi-task learning idea to solve a problem of detection with one class SVM when new sensors are added to the system. The main idea is to adapt the detection system to the upgraded sensor system. To solve that problem, the kernel matrix of multi-task learning model can be divided into two parts, one part is based on the former features and the other part is based on the new features. Typical estimation methods can be used to fill the corresponding new features in the old detection system, and a variable kernel is used for the new features in order to balance the importance of the new features with the number of observed samples. Experimental results show that it can keep the false alarm rate relatively stable and decrease the miss alarm rate rapidly as the number of samples increases in the target task.

1 INTRODUCTION

In real applications, many machine learning models may not work very well due to the ideal assumption that the training data and the future data are subject to the same distribution or that they are observed in the same feature space, which may not hold with recent system that can evolve based on sensor upgrade or use of logical software based on sensors. Transfer learning approach arose accordingly to solve that problem, and it has received significant attention in recent years, which is widely studied in both supervised learning and unsupervised learning area (Pan and Yang, 2010). In this paper, we focus on using the multi-task learning approach to solve the transfer learning problem to one class classification or outliers detection problem, where the detection model may experience a change due to practical reasons.

For detection, two kinds of one class support vector machines are mainly used. One is proposed by (Tax and Duin, 1999), which aims to find a hypersphere with minimal volume to enclose the data samples in feature space, the amount of data within the hypersphere is tuned by a parameter $C$ (noted as $C$-OCSVM). Another one is introduced by (Schölkopf et al., 2001), which finds an optimal hyperplane in feature space to separate a selected proportion of the data samples from the origin, and the selection parameter is $\nu$ which gives an upper bound on the fraction of outliers in the training data (noted as $\nu$-OCSVM). It is proved that these two approaches lead to the same solution according to (Chang and Lin, 2001), if a relationship between parameters $\nu$ and $C$ is fulfilled and under build condition over the choice of the kernel.

From data driven side, we can divide the issues for such detection system into two categories. One is the transfer learning problem when the feature space remains the same meaning that the number of features is not changed but are drawn from a different data distributions. For example, the introduction of a detection task for a new version of a system, or the update of a detection after system maintenances with sensor update. Another issue is the transfer learning problem in different feature space, where we have different number of features for the target task. For example, in the application of fault detection for an engine system, there are a few sensors which have already worked on an engine diagnosis system for much time and every sensor gets a few data. Now due to technical or some other practical needs, such as improving detection performances, new sensors are added to this system. As far as we know, this problem has never been tackled in the detection context using one class SVM.

Instead of training a new detection system from scratch, multi-task learning seems to be an ideal mean to adapt the former detection to an updated system, since it uses the assumption which is satisfied in
our context that related tasks share some common structure or similar model parameters (Evgeniou and Pontil, 2004), assuming one task is the former system and the second one is the updated system. And the idea is also used to solve one class classification problem by (Yang et al., 2010; He et al., 2014), but both of them are subject to the situation that the related tasks are in the same feature space. In (Xue and Beausery, 2016), a new multi-task learning model is proposed to solve the detection problem when additional new feature is added, where it gives a good transition from the old detection system to the new modified one. However, in some cases the kernel matrix in that model is not positive semi-definite which means that some approximation in a semi-definite subspace must be considered to determine the detection.

In this paper, a new approach is proposed to avoid that issue. As is shown in section 2.2, we can divide the kernel matrix into two part, one part is based on the old features and the second part is based on the new added feature. After typical estimation method is conducted to fill the corresponding new feature in the old detection system in order to get a positive semi-definite matrix, a specific variable kernel is used in the second kernel matrix (which is base on the new feature) to control the impact of the new feature over the detection according to the amount of collected new data.

The paper is organised as follows. In section 2, we propose the approach to use multi-task learning idea to solve one class SVM problems with the same features and with additional new features respectively. Then we prove the effectiveness of the proposed approach by experimental results in section 3. Finally, we give conclusions and future work in section 4.

2 MULTI-TASK LEARNING FOR ONE CLASS SVM

For the one class transfer learning classification problem, two kinds of situation might happen depending whether the source task and the target task share the same feature space (homogenous case) or not (heterogenous case). To study the heterogenous case, we consider the situation of adding new feature one by one in target task to simulate the modification or evolution of an existing detection system.

2.1 Homogeneous Case

Consider the case of source task (with data set \( X_1 \in \mathcal{R}^n \)) and target task (with data set \( X_2 \in \mathcal{R}^n \)) in the same space. For source task, a good detection model can be trained based on a large number of samples \( n_1 \). After the maintenance or modification of the system, we have just a limited number of samples \( n_2 \) during a period of time. Intuitively, we may either try to solve the problem by considering independent separated tasks or treat them together as one single task. Inspired by references (Evgeniou and Pontil, 2004) and (He et al., 2014), a multi-task learning method which tries to balance between the two extreme cases was proposed by (Xue and Beausery, 2016). The decision function for each task \( t \in \{1, 2\} \) (where \( t = 1 \) corresponds to the source task and \( t = 2 \) corresponds to the target task) is defined as:

\[
 f_t(x) = \text{sign}(\langle w_t, \phi(x) \rangle - 1),
\]

where \( w_t \) is the normal vector to the decision hyperplane and \( \phi(x) \) is the non-linear feature mapping. In the chosen multi-task learning approach, the needed vector of each task \( w_t \) could be divided into two part, one part is the common mean vector \( w_0 \) shared among all the learning tasks and the other part is the specific vector \( v_t \) for a specific task.

\[
 w_t = \mu w_0 + (1 - \mu) v_t,
\]

where \( \mu \in [0, 1] \). When \( \mu = 0 \), then \( w_t = v_t \), which corresponds to two separated task, while \( \mu = 1 \), implies that \( w_t = w_0 \), which corresponds to one single global task. Based on this setting, the primal one class problem could be formulated as:

\[
 \begin{align*}
 \min_{w_0, v_t, \xi_t} & \frac{1}{2} \mu \| w_0 \|^2 + \frac{1}{2} (1 - \mu) \sum_{t=1}^2 \| v_t \|^2 + C \sum_{t=1}^2 \sum_{i=1}^{n_t} \xi_{it} \\
 \text{s.t.} & \quad \langle w_0 + (1 - \mu) v_t, \phi(x_{it}) \rangle \geq 1 - \xi_{it}, \quad \xi_{it} \geq 0,
\end{align*}
\]

(3)

where \( t \in \{1, 2\} \), \( x_{it} \) is the ith sample from task \( t \), \( \xi_{it} \) is the corresponding slack variable and \( C \) is penalty parameter.

Based on the Lagrangian, the dual form could be given as:

\[
 \begin{align*}
 \max_{\alpha} & \quad \frac{1}{2} \alpha^T K \alpha + \alpha^T 1 \\
 \text{s.t.} & \quad 0 \leq \alpha \leq C I, \\
\end{align*}
\]

(4)

where \( \alpha = [\alpha_{11}, \ldots, \alpha_{n1}, \alpha_{12}, \ldots, \alpha_{n2}] \) and

\[
 K = \begin{bmatrix} K_{ss} & \mu K_{st} \\ \mu K_{ts} & K_{tt} \end{bmatrix}
\]

(5)

is a modified Gram matrix, \( K_{ss} = \langle \phi(X_1), \phi(X_1) \rangle \), \( K_{tt} = \langle \phi(X_2), \phi(X_2) \rangle \), \( K_{st} = \langle \phi(X_1), \phi(X_2) \rangle \), \( K_{ts} = \langle \phi(X_2), \phi(X_1) \rangle \), which means that we can solve the problem by classical one-class SVM with a specific kernel (we use Gaussian kernel in this paper).

Accordingly, the decision function for the target task could be defined as:

\[
 f_2(x) = \text{sign}(\mu \langle \phi(X_1), \phi(x) \rangle - \langle \phi(X_2), \phi(x) \rangle - 1).
\]

(6)
2.2 Heterogenous Case

Due to practical reasons, when new feature is added to the old detection system, if we continue to use the old detection system we will not be able to take advantage of the new information to improve the detection performances. If we wait until we gather enough new data to train a new detector which means that on one hand we have to delay the benefit of the update of the system, and on the other hand we have to go through all the hyper parameter optimisation process which may be time consuming. On the contrary, the multi-task learning model should be able to take into consideration the information brought by the new feature. We introduce a former method (MTL_I) and a new one (MTL_{II}) to tackle that problem. For both we consider \( X_1 \in \mathcal{R}^p \) be the data set of the old detection system, and \( X_2 \in \mathcal{R}^{p+1} \) be the data set since new feature is added.

2.2.1 MTL_I

Notice that for the formulation of multi-task learning (4), if we want to compute the modified Gram matrix (5), problem happens with block matrix \( K_{st} \) because of the different features for the source task and the target task. In the work of (Xue and Beauseroy, 2016), named as MTL_I, the new feature is ignored for computing matrix \( K_{st} \). To some extend, it gives a balance from the old detection system to the new one by tuning the parameter \( \mu \) with a proposed criteria. However, by using this method, the modified kernel matrix is not always positive semi-definite which means that a global optimisation solution can not be guaranteed with standard approach.

2.2.2 MTL_{II}

To fill the corresponding new feature, some estimation methods like the nearest neighbour, the imputation etc., can be used. Accordingly, we get \( \tilde{X}_1 = \{ x | x^{(1)}, \ldots, x^{(p)}, \tilde{x}^{(p+1)} \} \), where \( \tilde{x}^{(p+1)} \) is the new feature in the old detection system estimated by using information from \( X_2 \). The drawback of this method is that when the number of samples \( X_2 \) for target task is small, it is hard to give a good estimation to the new feature in \( X_1 \).

Once we get \( \tilde{X}_1 \in \mathcal{R}^{p+1} \) and \( X_2 \in \mathcal{R}^{p+1} \), as we use Gaussian kernel, then the kernel matrix in (5) can be decomposed into two part:

\[
K^\mu = \begin{bmatrix}
K_{ss} & \mu K_{st} \\
\mu K_{ts}^T & K_{tt}
\end{bmatrix}_{\mathcal{R}^{p+1}}
\]

where \( \circ \) is element-wise product and \( A_0 \) is kernel matrix based on \( \mathcal{R}^p \) with the first \( p \)th features for \( X_1 \) and \( X_2 \). \( A_1 \) is kernel matrix based on \( \mathcal{R}^1 \) space with the \( p+1 \)th estimated feature \( \tilde{x}^{(p+1)} \) from \( X_1 \) and \( x^{(p+1)} \) from \( X_2 \). Notice that \( K^\mu \) is a positive semi-definite matrix when \( \mu \in [0, 1] \), even if different kernel parameters are adopted for computing \( A_0 \) and \( A_1 \).

We use the Gaussian kernel that is defined as:

\[
k(x_i, x_j) = \exp\left(\frac{||x_i - x_j||^2}{-2\sigma^2}\right),
\]

where \( \sigma \) is the kernel parameter. Notice that when \( \sigma \to +\infty \) then \( k(x_i, x_j) \to 1 \). So we propose to use the former \( \sigma_0 \) for \( \mathcal{R}^p \) subspace and to choose a varying \( \sigma(n) \) for the new feature, where \( n \) is the number of samples. As a first intuition, we want \( \sigma(n_2) \) to be large when \( n_2 \) is small and to be close to \( \sigma_0 \) when \( n_2 \) is large.

By doing this, the entries of matrix \( A_1 \) will tend to be 1 when \( n_2 \) is small, which means that it does not have very important influence to the total kernel matrix when the estimation of the new feature \( \tilde{x}^{(p+1)} \) in \( X_1 \) is not very dependable. As \( n_2 \) becomes larger, more information is brought in from the new feature and a better estimation of \( \tilde{x}^{(p+1)} \) will be obtained, more consideration should be taken for matrix \( A_1 \), so \( \sigma \) decreases and it converges to the same value as \( \sigma_0 \) when \( n_2 \) is large enough.

In kernel density estimation, the optimal window width for a standard distribution is given by (Silverman, 1986):

\[
h_{opt} = \left(\frac{4}{d+2}\right)^{\frac{1}{d+4}} \frac{1}{n} \frac{1}{\bar{f}^{\frac{1}{d+4}}},
\]

where \( d \) is the number of dimensions and \( n \) is the number of samples.

Upon above, the kernel parameter function for \( A_1 \) could be defined as:

\[
\sigma(n) = c_2 \exp\left(\frac{c_1}{\sqrt{n}} h_{opt}\right),
\]

where the exponent function \( \exp\left(\frac{c_1}{\sqrt{n}} h_{opt}\right) \) decreases from a large value when \( n \) is small to a small value close to 1 when \( n \) is large, which means that we multiply \( h_{opt} \) by a large number at the beginning and we almost keep \( h_{opt} \) when \( n \) is large enough. The constant \( c_1 \) is used to control the value that we want to multiply \( h_{opt} \) when \( n \) is small and \( c_2 \) is a scale factor that makes \( \sigma(n) \) converge to \( \sigma_0 \) when \( n \) is large. A few groups of \( \sigma(n) \) are shown in figure 2. We name this multi-task learning method as MTL_{II} in this paper.
3 EXPERIMENTS

In this section, experiments are conducted on artificial data set. We compare the proposed method $MTL_{II}$ with the former one $MTL_{I}$, as well as the other possible solutions: the old detection system $T_1$ based on the old features, the new detection system $T_2$ based on data when new feature is added, and the union detection system $T_{big}$ which is based on the estimated data $\tilde{X}_1$ and the new obtained data $X_2$.

3.1 Setup

Let $y_1, y_2, y_3, y_4 \sim N(0, 1)$, three features are defined as:

\[
x^{(1)} = y_1, \quad x^{(2)} = 3 \cos \left( \frac{1}{2} y_1 + \frac{1}{2} y_2 + \frac{1}{4} y_3 \right) + N(0, 0.05), \quad x^{(3)} = y_4,
\]

where $N(0, 0.05)$ is Gaussian noisy. We use $X_1 = \{ x | x^{(1)}, x^{(2)} \}$ as the data set for the old detection system (source task), and $X_2 = \{ x | x^{(1)}, x^{(2)}, x^{(3)} \}$ as the data set for the new detection system (target task). The number of training samples is $n_1 = 200$, and we increase $n_2$ from 5 to 400 to simulate the change of the new detection system. A 3 dimensional view of the data set is shown in figure 1.

![3D view of the data set.](image)

To test the performance of the detection system, 20,000 positive samples are generated from $X_2$ to test the false alarm rate. Besides that, we use 20,000 uniform distribution data which cover the whole test data set to test the performance of miss alarm rate. Specifically, let $u^{(1)}, u^{(2)}, u^{(3)} \sim U(-4, 4)$, three groups of negative samples are defined as:

1. Uniform distribution for all the features $X_{negI} = \{ x | u^{(1)}, u^{(2)}, u^{(3)} \}$.
2. Uniform distribution only for the third dimension $X_{negII} = \{ x | x^{(1)}, x^{(2)}, u^{(3)} \}$ to simulate the outliers coming from the new added feature.
3. Uniform distribution only for the first two dimensions $X_{negIII} = \{ x | u^{(1)}, u^{(2)}, x^{(3)} \}$ to simulate the outliers coming from the old features.

We choose kernel parameter $\sigma_0 = 1.75$ and $\nu = 0.1$ for $\nu$-OCSVM (it exits a corresponding $C$ for C-OCSVM) which make the proportion of outliers around 0.1 for the old detection system at the beginning. A list of the comparison of different methods is shown in table 1. Where $\tilde{X}_1 = \{ x | x^{(1)}, x^{(2)}, \tilde{x}^{(3)} \}$, $\tilde{x}^{(3)}$ is the estimated feature (we use nearest neighbour method to fill this new feature) and $X_2 \setminus x^{(3)}$ denotes that $X_2$ without the new feature. For $T_1$, $T_2$ and $T_{big}$, the same kernel parameter $\sigma_0$ is used, for $MTL_{I}$ the setting is same as in (Xue and Beauseroy, 2016) and for $MTL_{II}$, $\sigma_0$ is used for the first two features and a variation of $\sigma(n)$ according to (10) is used for the third feature. The choice of $\mu$ for $MTL_{II}$ is conducted by the criteria proposed in (Xue and Beauseroy, 2017). All the results are averaged by 10 times.

<table>
<thead>
<tr>
<th>Compare methods</th>
<th>Train data sets</th>
</tr>
</thead>
<tbody>
<tr>
<td>$T_1$</td>
<td>$X_1, X_2 \setminus x^{(3)}$</td>
</tr>
<tr>
<td>$T_2$</td>
<td>$X_2$</td>
</tr>
<tr>
<td>$T_{big}$</td>
<td>$\tilde{X}_1, X_2$</td>
</tr>
<tr>
<td>$MTL_{I}$</td>
<td>$X_1, X_2$</td>
</tr>
<tr>
<td>$MTL_{II}$</td>
<td>$\tilde{X}_1, X_2$</td>
</tr>
</tbody>
</table>

3.2 Performance with Different Kernel Parameters

Three groups of kernel parameters $\sigma_1, \sigma_2, \sigma_3$ are generated to test the performance of $MTL_{II}$. As shown in figure 2, we choose $c_1 = 1, 3, 6$ and then choose

![Different kernel functions.](image)
corresponding $c_2$ in (10) which makes $\sigma(400) = \sigma_0$ (where $\sigma_0 = 1.75$ is the kernel parameter for the old detection system).

Results of $MTL_{II}$ are shown in figure 3 with different $\sigma$ for computing $A_1$ in (7). If we use constant $\sigma_0$, the false alarm rate is very high when $n_2$ is small because of the bad estimation while lack of samples from $X_2$. Both the false alarm rate and the miss alarm rate will become more stable as $n_2$ increases due to better estimation for $\sigma(3)$. However, with the variation of kernel parameters $\sigma_1, \sigma_2, \sigma_3$, when $n_2$ is small, the larger $\sigma$ is, the closer of $A_1$ is to a matrix with $1$ elements (that means we are using a kernel matrix which is very close to the matrix just based on the old features), so we increase less for the false alarm rate ($MTL_{II}(\sigma_3) < MTL_{II}(\sigma_2) < MTL_{II}(\sigma_1) < MTL_{II}(\sigma_0)$).

As for the miss alarm rate on $X_{\text{neg}}$ (figure 3(b)) to simulate the outliers coming from for all features, the method with variation kernel parameters increases a bit at the beginning and it decreases rapidly to the same value as we use fixed one. The same trend happens for data set $X_{\text{negIII}}$ (figure 3(c)) to simulate the outliers coming from the new features except at the beginning, where the miss alarm rate is relatively high, but as we increase $n_2$, we decrease $\sigma$ and the miss alarm rate decreases rapidly to the same value with fixed $\sigma_0$. This kind of trend makes meaningful sense because when new feature is added, while $n_2$ is small, if outliers are all from the new feature, we can not decide them all as negative samples, instead we would rather keep a relative stable false alarm rate while reduce the miss alarm rate rapidly as $n_2$ increases which means that we take the new feature’s information into consideration gradually. For the miss alarm rate on $X_{\text{negIII}}$ (figure 3(d)), all methods keep almost stable which means that we do not increase the miss alarm rate if the outliers come from the old features. From the above analysis, $MTL_{II}(\sigma_1)$ produces a relatively good detection model when new feature is added, where $\sigma_3$ is relatively large at the beginning and it converges to $\sigma_0$ at the end.

---

Figure 3: Results of different kernel parameters for $MTL_{II}$ (a) false alarm rate, (b) miss alarm rate on $X_{\text{neg}}$ (uniform data for all features), (c) miss alarm rate on $X_{\text{negIII}}$ (uniform data only for new feature), (d) miss alarm rate on $X_{\text{negIII}}$ (uniform data only for old features).
Figure 4: Compare results of different methods: (a) false alarm rate, (b) miss alarm rate on \( X_{\text{negI}} \) (uniform data for all features), (c) miss alarm rate on \( X_{\text{negII}} \) (uniform data only for new feature), (d) miss alarm rate on \( X_{\text{negIII}} \) (uniform data only for old features).

### 3.3 Experimental Results

We use \( \text{MTL}_{\text{II}}(\sigma_3) \) to compare with the other possible methods listed in table 1, results are reported in figure 4. Besides that, in order to study the problem that might happen is the adaptation for the old feature space (that means the data distribution for the old features may experience a change due to system maintenance or update), we give a rotation of \( \frac{\pi}{6} \) to the first two features in \( X_2 \) to study the model’s performance on this situation, and the results are shown in figure 5.

For the method \( T_1 \), which is trained on the old features of \( X_1 \) and \( X_2 \), the false alarm rate is almost constant around 0.1, but the miss alarm rate is the highest one among all the other methods because it does not take into consideration of the new feature.

For \( T_2 \) which is based only on \( X_2 \) since the new feature is added, it gives very high false alarm rate when \( n_2 \) is small, which means that it does not make full use of the information from the former detection system at the beginning, as \( n_2 \) increases large enough (here \( n_2 > 150 \)), it produces more stable false alarm rate and miss alarm rate.

If we combine the estimated data set \( \tilde{X}_1 \) and \( X_2 \) to train a detection model, named as \( T_{\text{big}} \), the false alarm rate is lower than that of \( T_2 \), and the miss alarm rate will end up with the same as \( T_2 \). However, with a rotation of the first two features in \( X_2 \), it will increase the chance of miss alarm at the end (which is shown in figure 5(b), 5(c) and 5(d)), because \( T_{\text{big}} \) tends to in-close all the train data set together. That means \( T_{\text{big}} \) is not practical when data distribution of the old features experiences a change in the new detection system.

For multi-task learning method, both \( \text{MTL}_{\text{II}} \) and \( \text{MTL}_{\text{II}} \) gives a transition from the old detection system \( T_1 \) (which is just based on the old features) to the new modified system \( T_2 \) (which is based on the new data set \( X_2 \) since new feature is added) as \( n_2 \) increases. The false alarm rate of \( \text{MTL}_{\text{II}} \) is a bit lower than that of \( \text{MTL}_{\text{II}} \), and both of them are relatively stable compared to \( T_2 \) and \( T_{\text{big}} \). But for miss alarm rate, only \( \text{MTL}_{\text{II}} \) converges to that of \( T_2 \) while \( \text{MTL}_{\text{II}} \) does not.
as \( n_2 \) increases. And the general miss alarm rate of \( MTL_{II} \) is much lower than that of \( MTL_I \), this difference is much larger when there is a rotation to the first two features in \( X_2 \) (figure 5). Therefore, \( MTL_{II} \) gives a better transition from the old detection system to the new one than \( MTL_I \), it can keep the false alarm rate relatively stable while decrease the miss alarm rate rapidly to a stable value.

4 CONCLUSIONS

In this paper, a modified approach of multi-task learning method \( MTL_{II} \) is proposed to solve the problem of transfer learning to one class SVM, where additional new features are added in the target task.

The idea is to decompose the kernel matrix in multi-task learning model into two parts, one part is the kernel matrix based on the old features and the other part is the kernel matrix based on the new added features. Typical methods can be used to estimate the corresponding new features in the source data set in order to compute the kernel matrix based on the new features. Then a variable kernel is used to balance the importance of the new features with the number of new samples and at last it converges to the same value as used in the old detection system. Experimental results show that the proposed method outperforms the former proposed method \( MTL_I \) and the other possible approaches.

Future work may consider online implementation of the proposed approach.

REFERENCES


Xue, Y. and Beuseroy, P. (2017). Transfer learning for one class svm adaptation to limited data distribution change. *Pattern recognition letters, accepted*.