Contents

1 Introduction

Recent research studies in data science focus on Machine Learning techniques to either predict output for a given set of inputs (testing data) or classifying the inputs on the basis of a labelled set of inputs known beforehand (training data). A number of techniques have been developed which perform the above mentioned tasks with high accuracy.

However, these techniques require a very strong assumption - the training and testing data-set should belong to the same domain and should have similar distribution. This requirement may not be fulfilled every time, which may lead to inaccurate results if these algorithms are applied on such problems. Moreover there are certain situations under which training data is not available or is very expensive to obtain while working on a particular domain. In these circumstances, there is a need for ***knowledge transfer*** from one domain to another.

2 Problem Definition and Objective

We aim to use the concept of Transfer Learning to gain knowledge from the data available in one domain, and apply it to the problem of interest in another domain. For this purpose, we use the Kernelized Bayesian Transfer Learning technique.
The first part of the project aims to implement the Kernelized Bayesian Transfer Learning approach for supervised learning, i.e. only labelled data of all the domains are available.
The second phase of the project aims to solve the similar problem with the difference that we also have unlabelled data points in addition to the labelled data points. This is known as semi-supervised learning.

3 Literature Survey

S No.	Author	Paper Title	Year	Crux	Venue
1.	Mehmet Gonen Adam A. Margolin	*Kernelized Bayesian Transfer Learning* [1]	2014	*This paper provides an insight on Transfer Learning. Firstly it defines Transfer Learning Mechanism and then a Bayesian Model is proposed to learn in One Domain and use this knowledge to the problem of interest in Another Domain. The concept of Kernel trick, priors and posterior probability distribution are used to achieve this.*	Twenty-eighth AAAI Conference on Artificial Intelligence.
2.	Kevin P. Murphy	*Book : Machine Learning – A probabilistic perspective.* [4] *Chapter 3 and 5*	2012	*Chapter 3 – Generative Models for Discrete Data define the concepts of Prior, Likelihood, Posterior and Posterior Predictive Distribution.* *Chapter 5- Bayesian Statistics gives advance details on summarizing posterior distributions and Bayesian Model Selection using Priors and Hierarchical Bayesian Approach.*	Massachusetts Institute of Technology.
3.	Mehmet Gonen	*Bayesian Efficient Multiple Kernel Learning.* [5]	2012	*This Paper provides the conjugate distribution theory for approximation of hper-parameters during training.*	Helsinki Institute of Information Technology HIIT, Dept of Information and Computer Science, Aalto University School of Science
4.	E. Amid, A. Gionis and A. Ukkonen	*A Kernel-learning approach to Semi-Supervised clustering with relative distance comparisons.* [7]	2015	*This Paper provides a Kernel-based approach to minimize the relative distance between data points to obtain clusters in a Semi-Supervised environment.*	Helsinki Institute for Information Technology, Department of Computer Science, Aalto University

4 Transfer Learning

Transfer Learning is an advanced Machine Learning technique which enables us to apply the knowledge learned from the training data in one domain to achieve the desired results in a different domain.

4.1 Traditional ML vs Transfer Learning

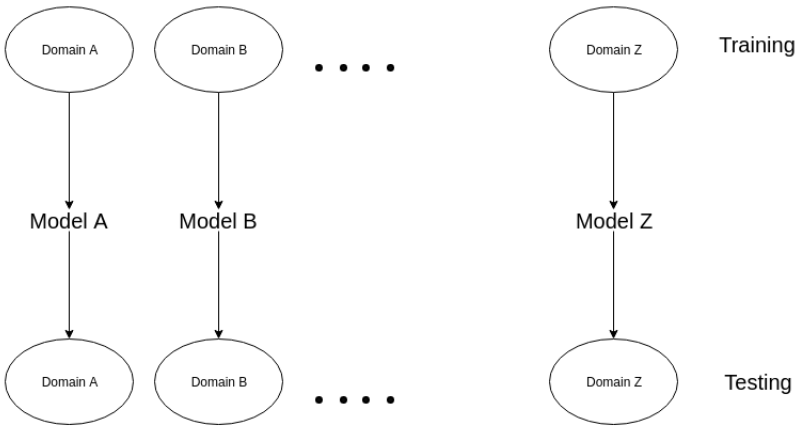

Figure 1: Traditional Machine Learning methods

In the traditional method, we train our model on a particular domain and testing is done for unknown data in the **same** domain.

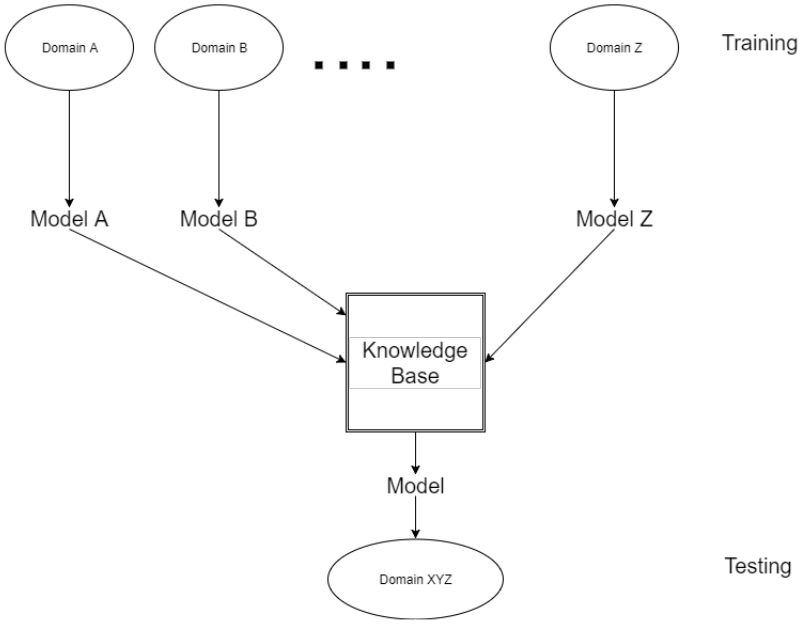

Figure 2: Transfer Learning approach

In the transfer learning approach, the testing is to be performed on a **different** but related domain. The training data-set contains a relatively large number of labelled data points from the source domain and fewer data points from the target domain. The aim is to learn a model which can predict or classify the inputs from the target domain with high accuracy.

4.2 Definition

Formally, Transfer Learning can be defined in terms of Domain and Task.

Domain: "A domain $\mathcal{D}$ consists of a feature space $\mathcal{X}$ and a marginal probability distribution $P(X)$ over the feature space, where $X = x_1, x_2, \cdots, x_n \in \mathcal{X}$." [1]

Task: "Given a domain $\mathcal{D}$, a task $\mathcal{T}$ consists of a label space $\mathcal{Y}$ and a conditional probability distribution $P(Y|X)$ that is typically learned from the training data consisting of pairs (x_i, y_i), where $x_i \in \mathcal{X}$ and $y_i \in \mathcal{Y}$." [2]

Transfer Learning: "Given a source domain $\mathcal{D}_S$, corresponding source task $\mathcal{T}_S$ as well as a target domain $\mathcal{D}_T$ and corresponding target task $\mathcal{T}_T$, the objective of transfer learning is to enable us to learn the target conditional probability $P(Y_T|X_T)$ in $\mathcal{D}_T$ with the information gained from $\mathcal{D}_S$ and $\mathcal{T}_S$ where $\mathcal{D}_S \neq \mathcal{D}_T$ or $\mathcal{T}_S \neq \mathcal{T}_T$."[3]

5 Requirements for Transfer Learning

Transfer learning requires that both the Source and Target domains must be different yet related. Transfer Learning methods involve trying to bring the distributions of both the domains very close to each other. If distributions are already similar, then proceed to the goal otherwise further domain adaptation techniques are required. Both the source domain and the target domain are brought to a common shared subspace which enables to facilitate moving towards the goal.
However if the domains are not well-related i.e. distributions are very much different, then it will lead to Negative Learning.

Negative Learning: Negative learning is said to occur when the learning process which tries to combine the source and target domain into a shared subspace has adverse effects on the accuracy of the model. In such a scenario, if we had not incorporated the source domain and the learning was done on the target domain only, the results would have been better. But due to the inclusion of source domain which is not well-related to the target domain, the accuracy of the model deteriorates.

[1] https://journalofbigdata.springeropen.com/articles/10.1186/s40537-016-0043-6
[2] http://ruder.io/transfer-learning/index.html#adefinitionoftransferlearning
[3] http://ruder.io/transfer-learning/index.html#adefinitionoftransferlearning

6 Transfer Learning Scenario

In Transfer Learning, the source and target domain can vary in the following four ways:

1. The Feature Spaces of both can vary.
2. The Probability Distributions can be different.
3. The Label Spaces of both the domains can be distinct.
4. The Conditional Probability Distributions may not be similar.

7 Types of Domain Adaptations

Type	Training		Testing	
	Labeled	Unlabeled	Labeled	Unlabeled
Unsupervised	Yes	Yes	No	Yes
Semi-supervised	Yes	Yes	Yes	Yes
Supervised	Yes	No	Yes	No

8 Proposed Approach

8.1 Bayesian Approach

The Bayesian approach to Machine learning uses the probabilistic methods to learn from the available data. For the learning purpose, a model is defined which will have some unknown parameters. For the unknown parameters, we assume that they follow some known probability distribution, for example, Normal or Gaussian distribution. On the basis of the training data available, we calculate the posterior probability for the unknown parameters given the available training data. The posterior distribution thus obtained is used to determine the optimal value for the parameters such that it produces the best possible results.

8.2 Kernelized Bayesian Transfer Learning

We are going to define a model which uses the above mentioned approaches. The model does the classification by using a two-step process:

1. ***Subspace Reduction***: The data points from the different tasks have different dimensionalities. So, for establishing some kind of relationship among these data points, they need to be brought into a common subspace. This is done using the kernel-based dimensionality reduction technique. This technique uses the kernel trick to bring the data points into a shared subspace by using a pre-defined kernel function.

2. ***Classification***: After bringing the data points into a common subspace, classification is done using the parameters that were learned during the training.

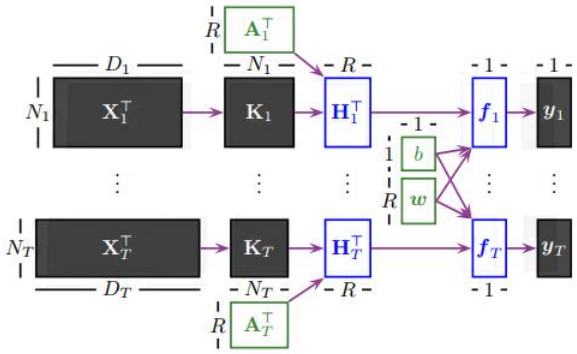

Figure 3: Flowchart for Kernelized Bayesian Transfer Learning
Image taken from [1]

The above figure describes the process used for Kernelized Bayesian Transfer Learning. Let the number of tasks be T, the number of data points for the tasks be $N_1, N_2, \cdots, N_T$ and the dimensions for the tasks be $D_1, D_2, \cdots, D_T$. For the classification purpose, we use the following approach:

The matrices $K_1, K_2, \cdots, K_T$ are the Kernel matrices and the matrices $A_1, A_2, \cdots, A_T$ are the projection matrices. These two matrices are used for the dimensionality reduction part as described in section 8.2 pt 1. After the dimensionality reduction, the shared subspace representations are given by $H_t = A_t^T K_t$

The shared subspace representations are then used to perform classification by calculating the predicted outputs $f_t = H_t^T w + b$, where w and b are the priors used for classification. In this way, the data points from different tasks are used for developing a model which is able to relate those tasks accurately.

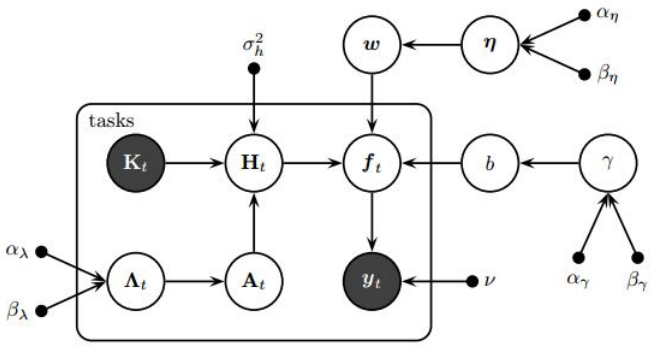

Figure 4: Graphical model for Kernelized Bayesian Transfer Learning
Image taken from [1]

The above figure shows the complete graphical model used for Kernelized Bayesian Transfer Learning. The entities A_t, K_t, H_t, f_t, y_t, w, b are the same as defined in the previous section.
Since Bayesian approach for transfer learning is used, we use priors for learning the optimal values of projection matrices A_t and classification parameters w and b. The priors used for this purpose are Λ, η and γ respectively.
For asserting the known probability distributions onto the priors, the hyper-parameters α and β are used.

To obtain the optimal values for the hyper-parameters and priors, we run a certain number of iterations of the learning phase of the algorithm to find a convergence point for the priors and hyper-parameters to minimize the posterior expectation loss. After getting the optimal values for the hyper-parameters and priors, we obtain the projection matrices A_t and the classification parameters w and b. These learned parameters and matrices are then used to classify and predict the label for the unknown data of the target domain.

8.3 Training Model

Initial Assumptions

· Λ **Priors** The priors for the task specific projection matrices A_t are given by Λ_t. The hyper-parameters which are used to set those priors are given by $\{\alpha_\lambda, \beta_\lambda\}$. λ is assumed to follow a Gaussian Distribution with scale parameter = α_λ and shape parameter = β_λ.

· γ **Priors** The priors for bias parameter b is given by γ. For this prior, the hyper-parameters are $\{\alpha_\gamma, \beta_\gamma\}$. γ is assumed to follow a Gaussian Distribution with scale parameter = α_γ and shape parameter = β_γ.

· η **Priors** The priors for the classification parameter w is given by η. The hyper-parameters for this prior are given by $\{\alpha_\eta, \beta_\eta\}$. η is also assumed to follow a Gaussian Distribution with scale parameter = α_η and shape parameter = β_η.

· **Margin parameter** ν The margin parameter is initially set to integral value 1 and is useful for classification of those data points which lie very near to the margin between the two classes. (Similar to SVM)

Updating during Training Phase

Using the concept of Conjugate Distributions [4], the posterior distributions can be learned as:

· $\alpha(\lambda) = \alpha_\lambda + 1/2$

· $\beta(\lambda) = (1/\beta_\lambda + \langle(a)^2\rangle/2)^{-1}$ where a is the element of Projection Matrix.

· $\alpha(\gamma) = \alpha_\gamma + 1/2$

· $\beta(\gamma) = (1/\beta_\gamma + \langle b^2\rangle/2)^{-1}$

· $\alpha(\eta) = \alpha_\eta + 1/2$

· $\beta(\eta) = (1/\beta_\eta + \langle w^2\rangle/2)^{-1}$

where $\langle A\rangle$ denotes the posterior expectation of A.

Prediction

After the posterior distributions of the priors are updated to reach convergence according to the training data available, we are able to determine the values of the projection matrices A_t and the classification parameters b and w. These learned values are then used to predict the output label to classify the unlabelled data in the target domain.
Given a point $x_{t,*}$ in the domain of target task t. The prediction is done as follows:

· The data point $x_{t,*}$ is brought into the shared subspace using the kernel and projection matrices.

· Using the classification parameters, the value $f_{t,*}$ is calculated as described in section 8.2 pt 2.

· The probability distribution for the prediction of the class label for the data point is given by
$p(y_{t,*} = +1|f_{t,*}) = \mathcal{Z}^{-1}\Phi(\frac{\mu((f_{t,*}) - \nu)}{\Sigma(f_{t,*})})$
where $\mathcal{Z}$ is the normalizing constant for the data point,
and $\Phi()$ is the standardized normal cumulative distribution function.

9 Hardware and Software Requirements

Hardware

- Testing Platform: Laptop HP-Envy-j106tx running under Linux (Ubuntu 16.04LTS) and Windows 10
- OS: Windows 8 or Higher/Ubuntu 14.04LTS or higher
- RAM: 4GB or higher
- Memory: 10GB+ (For MATLAB, Code, Dataset etc.)

Software

- MATLAB R2012a or above.
- Microsoft Office (MS Excel).
- Text Editors like Notepad++, Sublime Text etc.

10 Activity Time Chart

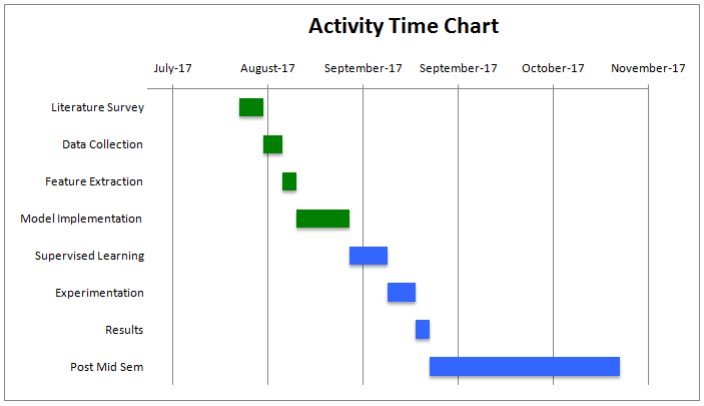

Figure 5: Gantt Chart

11 Work Done Till Mid Semester

Experiment 1

The following experiment was performed on the model defined in the previous sections.

- Dataset Used = Office Dataset[5] containing images from different categories like Backpack, Bottle, Helmet, Mouse, Mug, Printer etc. under 3 domains wiz. Amazon, DSLR and Webcam.
  For Binary Classification under this experiment, we will be using 2 categories only - Backpack and Bottle.
- Source Domain = Amazon Images
- Target Domain = Webcam Images
- Labels : The output labels assigned to the images are $y = +1$ for Backpack and $y = -1$ for Bottle.
- No of training Data Points: 20 Images from each category from Source Domain and 3 Images from each category from Target Domain.
- For the classification purpose, the images were brought to a common shared subspace by using the kernel-based dimensionality reduction techniques as mentioned in the previous section. After the dimensionality reduction, all the images were brought into a subspace with number of dimensions = 20.
- The learning part of the algorithm ran for 200 iterations to learn the optimal values for the classification parameters and projection matrices.
- The testing phase of the model, after performing the steps as mentioned in the previous section, returned the probability of the data point to be labelled as Backpack ($y = +1$). If the value returned from the testing phase is found to be near to 1, then we can safely say that the image on which the testing is performed was of backpack, on the other hand, if the value turns out to be closer to 0, then we can say that the image was of bottle.

Experiment 2

We repeat this experiment with increased number of Data Points in each category of Training Data.

Experiment 3

In this experiment, more categories were added repetitively to the source domain to check:

- The effect of increasing number of categories in the source domain on the accuracy of the model.
- The effect of unrelated categories on the output.
- The impact of Negative Transfer on the expected output.

12 Results

Experiment 1

Source Domain - Amazon Images
Target Domain - Webcam Images
Aim : To Learn from Amazon Images and Classify Webcam Images using the Model described above.
Number of Tasks = 2
Task 1 - Classifying Amazon Images (Source Domain)
Task 2 - Classifying Webcam Images (Target Domain)

Dataset Description : Training

1. Amazon Images
    - Category 1 : Backpacks, 20 Data Points
    - Category 2 : Bottles, 20 Data Points
2. Webcam Images
    - Category 1 : Backpacks, 3 Data Points
    - Category 2 : Bottles, 3 Data Points

The Webcam images were randomly chosen from the dataset.

Dataset Description : Testing

Testing is done only on Target Domain
Testing Data is a mixture of 26 Images of Backpack and 16 Images of Bottles.

Result

On running the algorithm for variable number of iterations and also different subspace dimensionalities, the
Accuracy : **80.95%**

Experiment 2

Source Domain - Amazon Images
Target Domain - Webcam Images
Aim : To Learn from Amazon Images and Classify Webcam Images using the Model described above.
Number of Tasks = 2
Task 1 - Classifying Amazon Images (Source Domain)
Task 2 - Classifying Webcam Images (Target Domain)

Dataset Description : Training

1. Amazon Images
    - Category 1 : Backpacks, 92 Data Points
    - Category 2 : Bottles, 36 Data Points
2. Webcam Images
    - Category 1 : Backpacks, 5 Data Points
    - Category 2 : Bottles, 3 Data Points

The Webcam images were randomly chosen from the dataset.

Dataset Description : Testing

Testing is done only on Target Domain
Testing Data is a mixture of 29 Images of Backpack and 16 Images of Bottles.

Result

On running the algorithm for variable number of iterations and also different subspace dimensionalities, the
Maximum Accuracy : **95%**
Average Accuracy : **93.3%**

Experiment 3

In this experiment, we will increase the Training Data by adding more categories in the Source Domain. These Data Points, as they belong to different categories, can be considered as impurities. Moreover this model is trained for Binary Classification, hence this experiment would result in whether the unknown Data Point is Backpack or not. We are not bothered if it is any other category (say Bottle, Headphone, Table Lamp etc.)
Source Domain - Amazon Images
Target Domain - Webcam Images
Aim : To Learn from Amazon Images and Classify Webcam Images using the Model described above.
Number of Tasks = 2
Task 1 - Classifying Amazon Images (Source Domain)
Task 2 - Classifying Webcam Images (Target Domain)

Dataset Description : Training

1. Amazon Images
    - Category 1 : Backpacks, 92 Data Points

- Category 2 : Bottles, 36 Data Points
- Category 3 : Bookcase, 82 Data Points
- Category 4 : File cabinet, 81 Data Points

2. Webcam Images
    - Category 1 : Backpacks, 5 Data Points

The Webcam images were randomly chosen from the dataset.

Dataset Description : Testing

Testing is done only on Target Domain
Testing Data is a mixture of 29 Images of Backpack and 16 Images of Bottles and 5 Images of Headphones.

Result

On running the algorithm for variable number of iterations and also different subspace dimensionalities, the
Maximum Accuracy : **92%**
Average Accuracy : **90%**

13 Work done till End Semester

The model developed and experimented upon, involved only Labeled Data from both the domains i.e Source and Target. Hence the domain adaptation referred to here is **"Supervised Domain Adaptation"** or **"Supervised Transfer Learning"** and since output label was provided, the results came out to be were quite accurate.
However, an area needs to be further explored which is **"Semi-Supervised Domain Adaptation"** or **"Semi-Supervised Transfer Learning"**. Under this, the aim is to develop a model which could work with a mixture of **Labeled Data** and **Unlabeled Data** both, in the source domain. Thus with this motivation, further process is to develop and experiment with model which has both Labeled and Unlabeled Data in both Source Domain and Labeled Data in the Target Domains.

14 Semi-Supervised Transfer Learning

Till now, our model worked with **Labeled Data** in both source as well as target domain during the training phase. In the training data set, we had to provide

the output labels for all the data points available, which was used to train the classification parameters of our model. However, when a lot of images are available in the source domain, providing the output label for each of the data point may not be possible, or it can be a costly and time-consuming process. Thus, there is a need to develop a model which can handle such training data sets in which output labels are not present for all of the data points available. This is known as **Semi-supervised training**.
We consider the problem in which we have a mixture of both labeled as well as unlabeled images in the source domain and only labeled images in the target domain in the training data set. We aim to learn the parameters of our model using this type of data set.

14.1 Problems with KBTL Model

As mentioned before, the Kernelized Bayesian Transfer Learning model developed thus far requires output labels for all of the input data points available for training. The model cannot work if some of the output labels are missing. This is because the output labels are themselves used to learn the classification parameters in the model. Hence, it is mandatory to provide output labels to perform the classification with acceptable accuracy.

14.2 Approach for Semi-supervised transfer learning

To use the Kernelized Bayesian Transfer Learning model in the modified problem which contains unlabeled data as well in the source domain, we apply a pre-processing step which uses clustering to assign output labels to each unlabeled data point. After clustering, we are able to assign an output label for each of the unlabeled data point, and thus the KBTL model can be used after this step.
Our approach is based on the fact that **clustering** will arrange the similar looking images in the same cluster, and if we have the label for a few of the images in a cluster, we can use that label to assign output labels for the other unlabeled images of the cluster. In this way, we can provide labels for every image. We may get some incorrect labels during the clustering phase, but we can rely on the fact that most of the images which form clusters together will be similar and not many of the labels generated will be incorrect. Hence, even though the accuracy will not be as high as it was when all the images were labeled, the accuracy shall be in an acceptable range. The reduction in the accuracy will be a trade-off for the addition of unlabeled images in the training data set. We aim to perform clustering in such a way that the accuracy remains acceptable. In the following section, we define the approach used for clustering the images during the pre-processing phase.

15 A Kernel-Learning Approach to Semi Supervised Clustering with Relative Distance Comparisons

15.1 Introduction

The aim of this approach [7] is to cluster a given data-set into k-clusters subject to an additional set of constraints based on relative distance comparisons between the data points. The purpose of using additional constraints is to present some background/side information which is specifically not present in the feature vectors directly. This approach uses a *Log Determinant Divergence* for learning the Kernel matrix subject to relative distance constraints. Now using this learned Kernel matrix, clustering can now be obtained using any general algorithm like *kernel k-means*. [4] After clustering, we can now assign the Labels to data points (which is basically the pre-processing task for Semi-Supervised Transfer Learning) and can follow the earlier approach, KBTL (cited in section 8.2) for Binary Classification.

15.2 Kernel Learning

15.2.1 General Terms

Here, we will describe some basic notations and terms used for kernel learning using relative distance approach.

1. $\mathcal{D}$ - set of data items.
2. $\mathcal{X}$ - set of feature vectors in d-dimensional space.
3. $\mathcal{C}$ - relative distance comparisons between data items in $\mathcal{D}$.
4. Given $\mathcal{X}$ and $\mathcal{C}$, begin with initial kernel matrix K_0 and obtain K that captures the distance between data items very accurately.

Given $\mathcal{X}$ and $\mathcal{C}$, begin with initial kernel matrix K_0 and obtain K that captures the distance between data items very accurately.

15.2.2 Distance Constraints

The constraints defined by $\mathcal{C}$ provide relative distance between 3 items in $\mathcal{D}$.
The constraints $\mathcal{C}$ can be partitioned into two sets, i.e. $\mathcal{C}_{eq}$ and $\mathcal{C}_{neq}$.
$\mathcal{C}_{neq}$ contains constraints in which one item is singled out as an outlier.
$\mathcal{C}_{eq}$ contains constraints where distance between all the three items are approximately the same.
Distance measure can be taken as any general distance metric depending on the

[4] Similar to k-means clustering but uses kernel method as a distance measure rather than Euclidean Distance

type of data used. Here, we have a high dimensional data, hence we are using Cosine of angle measure.

15.2.3 Log Determinant Divergence

In the Log Determinant Divergence approach, firstly we assume K_0 to be the initial Kernel matrix which is a Gaussian kernel matrix. The objective is to find Kernel matrix K that is very close to K_0 and satisfies the constraints in $\mathcal{C}$. Hence, we use the concept of log determinant to find the similarity between K and K_0.

$$D_{ld}(K, K_0) = tr(KK_0^{-1}) - log\ det(KK_0^{-1}) - n$$

15.2.4 Algorithm

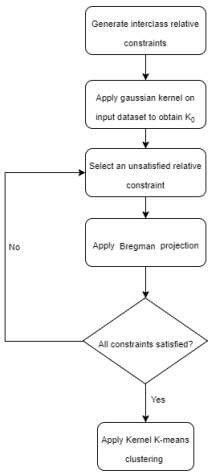

Figure 6: Algorithm for Semi-supervised clustering

Bregman's method is an iterative algorithm to solve certain convex optimization problems, Link - https://en.wikipedia.org/wiki/Bregman_method

16 Experiments

Here we represent some experiments which are step by step incremental procedure which would lead us to the expected result.

16.1 Experiment Pre-requisites

Firstly, lets describe the dataset being used in the experiments.

Dataset Description : Training

1. Source Images

   - Category 1 : Backpacks, 80 Data Points
   - Category 2 : Bike, 80 Data Points
   - Category 3 : Bookcase, 80 Data Points
   - Category 4 : Calculator, 80 Data Points
   - Category 5 : Deskchair, 80 Data Points

   Here **10 data points from each category are LABELED whereas 70 are UNLABELED**.
   Hence the datapoints from the source domain are a mixture of both Labeled and Unlabeled data, and hence Semi-Supervised appraoch needs to be used.

2. Webcam Images

   - Category 1 : Backpacks, 5 Data Points
   - Category 2 : Bottles, 3 Data Points

The Webcam images were randomly chosen from the dataset.

Dataset Description : Testing

Testing is done only on Target Domain
Testing Data is a mixture of 29 Images of Backpack and 16 Images of Bottles.

16.2 Experiment 1

In this Experiment, we will apply the SKLR Approach for clustering the Data Points in the Source Domain so as to obtain Labels for the Unknown Data Points which is the pre processing step for applying the rest of KBTL algorithm as Bayesian approach generally requires the Output labels for its functioning. So, We try to label the 350 unknown data points using this approach first.

16.3 Result 1

After performing Semi-Supervised clustering, we obtain 5 clusters. The cluster description is as follows:

1. Cluster 1 contains 92 Data Points
2. Cluster 2 contains 69 Data Points
3. Cluster 3 contains 74 Data Points
4. Cluster 4 contains 66 Data Points

5. Cluster 5 contains 99 Data Points

After performing the experiment for a number of times, the **accuracy** came out to be around **70.3%**

16.4 Flaw with Result 1

Having a look at the data set which we had been using for all the previous experiments, we see that the feature vectors are high dimensional vectors.
Clustering is a data analysis method in which we discover groups - called clusters - in a data set. Data objects within the same cluster should be very similar to each other, and dissimilar to data points in other clusters. Now in order to measure the closeness, we try to find a distance measure between the data points which is generally either Euclidean Distance or other distance measure like Manhattan Distance. So in that case, clustering data with high dimension can be an issue.
High dimensional data also suffers from the Hubness phenomenon [6] that affects the cluster analysis. Hence as the dimensionality increases, it may not be possible to find tight clusters and thus our accuracy was very less.
Therefore there is a need to reduce the dimensionality of the data so as to find proper clusters. We therefore need to use some dimensionality reduction procedure for reducing the no of dimensions and we thus need to consider **Principal Component Analysis**.

16.5 Experiment 2

Using the same dataset, we first try to reduce the number of dimensions and find tight clusters by iteratively changing the dimensions to obtain the best clusters. We start with no of dimensions as 5 and end at 10.
After getting the result, we can further apply the KBTL as proposed earlier for binary classification in which Backpack is considered as +1 and others as -1.

16.6 Result 2

After iteratively changing the no of dimensions from 5 to 10, we find that accuracy keeps increasing for the **CLUSTERING** Part.

1. No. of Dimensions = 5 : Accuracy 72.5%
2. No. of Dimensions = 10 : Accuracy 83.25%

After getting satisfactory and decent clusters, we are in a position to assign the output labels to the data points and thus apply KBTL for Binary Classification.

For **Classification using KBTL**, the accuracy came out to be as follows:

1. No. of Dimensions = 5 : Accuracy 84.4%
2. No. of Dimensions = 10 : Accuracy 86.67%

16.7 Flaw with Experiment 2

In the above experiment, we see that the number of constraints for each class combination imposed while using SKLR were very limited i.e 2 for each class combination. Hence the total no of inter-class and intra-class combinations constraints as described in the algorithm in section 15.2.2 were 40. Now, we wish to see the impact of number of constraints on clustering. Generally speaking, if we increase the number of constraints then we expect to see tight clusters. Thus this forms the base for the next experiment in which we try to see the influence of number of constraints on the number of clusters and the binary classification thus obtained.

16.8 Experiment 3

As described in the above statements, we now wish to change the number of constraints and see the effect of constraints increment.
When the number of constraints for each class combination is kept 5, then total no of inter-class and intra-class constraints come out to be 100.

16.9 Result 3

After iteratively changing the number of constraints from 2 to 5, we find that accuracy keeps increasing for the **CLUSTERING** Part.

1. No. of Dimensions = 5 : Accuracy 69.5%
2. No. of Dimensions = 10 : Accuracy 82.0%

After getting satisfactory and decent clusters, we are in a position to assign the output labels to the data points and thus apply KBTL for Binary Classification.

For **Classification using KBTL**, the accuracy came out to be as follows:

1. No. of Dimensions = 5 : Accuracy 93.3%
2. No. of Dimensions = 10 : Accuracy 93.3%

17 Conclusions

1. The Kernelized Bayesian Transfer Learning model is able to perform classification of images in Target Domain by learning from the images in Source Domain with good accuracy. Hence this model is quite apt for Transfer Learning.
2. As we increase the number of related data points in the source domain, the accuracy of the model tends to increase. Hence learning for target domain increases with increasing number of training data points in source domain.

3. The addition of impurities as described in the third experiment in section 11 did not have a huge effect on the accuracy. The accuracy decreased by only a small margin. The model did not experience Negative learning due to the addition of more unrelated images. Hence the model is not affected by impurities to a large extent.

These results were obtained when all the images were labeled, i.e. supervised transfer learning was performed.

We further explored the problem of transfer learning when the training was semi-supervised, i.e. the images in the training data set had a mixture of both labeled as well as unlabeled data. We used clustering as the pre-processing step to provide labels to the images for which the labels were missing. After this, the training data set along with the obtained labels were used as input to the KBTL model for classifying images as done previously.

We also used Principle Component Analysis (PCA) [8] to reduce the number of dimensions of the input data set. The data set with reduced number of dimensions was then used for clustering and classification using KBTL.

We performed various experiments on the approach we developed and concluded that

1. The accuracy of clustering decreases as we increase the number of constraints for each class combination. This is because of the tighter clustering due to the addition of more constraints during clustering. As clusters become tighter, the data points which were a little bit different could have been assigned to different clusters, thus resulting in a decrease in clustering accuracy.

2. The results obtained by using the data obtained from clustering as input to KBTL model, were better when the number of constraints were higher. This is due to the fact that as the clustering constraints were increased, the data points which fell into the same cluster were more similar, and thus the labelling would have been more accurate, thus resulting in a higher accuracy of classification.

3. The results of classification were more accurate when we decreased the number of dimensions of the training data set using Principle Component Analysis (PCA). The reduction in the number of dimensions would increase the effectiveness of the distance metric used, thus resulting in better results.

18 Future Scope

1. Firstly the KBTL Model can be extended to perform multi-label classification apart from Binary Classification. This would in turn increase

the overall scope of Transfer Learning to a variety of object classification rather than just 2.

2. Secondly, rather than pre-processing the data to obtain output labels, a model can be developed which could incorporate SKLR and KBTL in one single unit which would thus eliminate the computation complexity and effort.

References

[1] Mehmet Gonen, Adam A. Margolin, *"Kernelized Bayesian Transfer Learning"*, Proceedings of the Twenty-Eighth Conference on Artificial Intelligence, 2014.

[2] Kevin P. Murphy, Machine Learning: A Probabilistic Perspective, Massachusetts Institute of Technology, 2012, p. 67-90, 151-173

[3] Mehmet Gonen, *"Bayesian Efficient Multiple Kernel Learning"*, Helsinki Institute for Information Technology HIIT, Department of Information and Computer Science, Aalto University School of Science, 2012.

[4] Conjugate Prior : *https://en.wikipedia.org/wiki/Conjugate_prior*

[5] Office Dataset : *https://drive.google.com/file/u/1/d/0B4IapRTv9pJ1WTVSd2FIcW4wRTA/view?usp=drive_web*

[6] Tomašev N., Radovanović M., Mladenić D., Ivanović M. *"Hubness-Based Clustering of High-Dimensional Data"*, In: Celebi M. (eds) Partitional Clustering Algorithms. Springer, Cham, 2015

[7] E. Amid, A. Gionis and A. Ukkonen, *"A Kernel-learning approach to semi-supervised clustering with relative distance comparisons"*, Helsinki Institute for Information Technology, Department of Computer Science, Aalto University, 2015.

[8] Lindsay I Smith, *"A tutorial on Principal Components Analysis"*, 2002.