Concept Drift Region Identification via Competence-based Discrepancy Distribution Estimation

Fan Dong*†, Jie Lu†, Kan Li*, and Guangquan Zhang†
*School of Computer Science and Technology, Beijing Institute of Technology, Beijing 100081, China
†Centre for Artifical Intelligent, University of Technology Sydney, Ultimo, NSW 2007, Australia
fan.dong@student.uts.edu.au, jie.lu@uts.edu.au, likan@bit.edu.cn, guangquan.zhang@uts.edu.au

***Abstract*—Real-world data analytics often involves cumulative data. While such data contains valuable information, the pattern or concept underlying these data may change over time and is known as concept drift. When learning under concept drift, it is essential to know when, how and where the context has evolved. Most existing drift detection methods focus only on triggering a signal when drift is detected, and little research has endeavored to explain how and where the data changes. To address this issue, we introduce kernel density estimation into competence-based drift detection method, and invent competence-based discrepancy distribution estimation to identify specific regions in the data feature space where drift has occurred. Two experiments demonstrate that our proposed approach, competence-based discrepancy density estimation, can quantitatively highlight drift regions through data feature space, and produce results that are very close to preset drift regions.**

***Keywords*-concept drift; competence model; kernel density estimation;**

I. Introduction

Social media, embedded sensors and other information systems have been widely used in last decade, resulting every day in the generation and cumulation of a tremendous amount of unprocessed raw data. Effective and efficient learning methods are vital-needed for excavating and analyzing valuable information, the pattern or concept underlying these data. The challenge for learning data from real-world applications is that of concept drift, which refers to unpredicted changes that may occur over time in underlying data distribution [1]. It is a very pervasive phenomenon in real-world applications, e.g., the change of user preference in recommender systems, the emergence of new spam in email filter systems, or the evolution of intrusions in telecommunication. Gama et al. [2] give a formal definition of concept drift between time point t_0 and time point t_1 as follows: $\exists \mathbf{X} : P_{t_0}(X, y) \neq P_{t_1}(\mathbf{X}, y)$, where P_{t_0} denotes the joint distribution at time point t_0 between the set of input data $\mathbf{X}$ and the corresponding label y. Concept drift can be categorized into four common types: sudden, incremental, gradual and reoccurring context [3]. When new incoming data start to drift as in the movement of decision boundary, a well-trained static learning model will make more incorrect predictions. The learner will lose accuracy and will become outdated over time.

It is essential when learning under concept drift to know when, how and where the concept has changed. Drift detection methods are techniques that explicitly identify the occurrence of concept drift. Many drift detection methods have been developed over the past decades, and they are typically implemented by monitoring 1) the underlying distribution of data [4]–[6]; 2) the output (error-rate) of the learner [3], [7]–[9]. Most existing drift detection methods focus on triggering a signal when drift is detected. Little research endeavor has been made to explain how and where the data changes. Lu et al. [6], [10] and Dasu et al. [5] mention that their methods can locate the regions in which the data changes; however one issue should be addressed is that, the drift regions they identify are based on how their data model partitions data feature space. If the partitions are too coarse, the result will be imprecise. If the partitions are too fine, the computation costs will be increased.

Another issue should be emphasized is that, the concept drift regions can be utilized as additional information for reacting concept drift. Concept drift reaction refers to the techniques that update the predictive models from evolving data [2]. The most widely used strategy discards the current model when drift occurs and retrains a new predictive model from recent data cached in memory. However, this reaction strategy becomes ineffective when cached data is insufficient. One possible solution is to merge historical data from outside drift regions with recent cached data to construct a training set for building a new predictive model. In many situations, changes occur only in certain regions of the data space [2], thus granular models, such as decision trees (or decision rules), could be updated with local parts of a model, and the drift regions could be used as guide input to adapt these learning models.

Motivated by these issues, we introduce kernel density estimation into competence-based drift detection method, and invent competence-based discrepancy distribution estimation to identify specific regions in the data feature space where drift has occurred. By extracting essential information from competence models and constructing appropriate input of kernel density estimation, our competence-based discrepancy density estimation approach accurately maps the drift-affected discrepancy from one-dimensional competence space to the

multi-dimensional data feature space. Compared with other drift detection methods which have the ability to locate drift regions, our proposed approach demonstrates the following advantages: 1) it can identify critical regions in which drift has occurred in the multi-dimensional data feature space without using a specific space partitioning technique; 2) it can highlight drift regions more accurately than other methods; and 3) it is independent of the learning classifier and can be plugged-in to other drift reaction methods.

This paper is organized as follows. Section II, we investigate prior works related to this study. Section III presents our approach and relevant details. Section IV details two experiment results demonstrating that competence-based discrepancy density estimation is effective and includes a discussion of the limitations. Section V concludes this study and discusses our future work.

II. Related Works

A. Concept drift detection

Dasu et al. [5] presented an information-theoretic-based drift detection method which uses relative entropy, also called Kullback-Leibler divergence, to measure the difference between two sets of data within the given past and recent window. Their approach quantitatively identifies sub-regions of the data that have the greatest changes, revealing where the drift occurs. The principle behind this is that the data feature space is partitioned using *kdq-tree* and apply *Kulldorff's spatial scan statistic* on nodes of *kdq-tree*. However, Lu, Zhang & Lu [6] argued that the partition made by *kdq-tree* does not guarantee that the regions of greatest change will coincide with the true interesting concepts, and the partition may not be easily explained and understood. Their competence-based drift detection method also quantitatively describes when, how and where data change takes place, and demonstrates good performance in different scenarios [6]. The method they used to highlight drift-affected regions is based on competence models, called *Top-P-Competence Areas* (TPCA) [10]. However, TPCA only focuses on competence areas that have a large discrepancy. The regions highlighted by TPCA may miss some true drift regions.

B. Competence-based drift detection method

Competence-based drift detection [6] compares two unknown, multi-dimensional, non-parametric data distribution in the competence space instead of the original data feature space. By modeling data into competence model, multi-dimensional data can be maintained in one-dimensional space. The key idea of competence-based drift detection follows the principle that the probability distribution change of data should reflect its competence. The formal definitions for modeling data to competence are given below:

Definition 1. [11] For a case-base $\mathrm{CB} = \{c_1, c_2, \ldots, c_n\}$, given a case $c \in \mathrm{CB}$:

$$\mathrm{CoverageSet}(c) = \{c' \in \mathrm{CB} : \mathrm{Solve}(c, c')\}$$

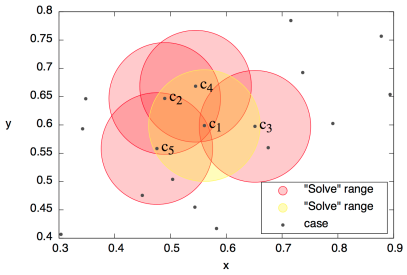


Fig. 1. An example of competence model.

where $\mathrm{Solve}(c, c')$ means that c can be retrieved and adapted to solve c'. The Solve rule can be defined in different ways. To illustrate this, we give an example of a Solve rule that two cases are mutually solve each other if the Euclidean distance of two cases is less than 0.1, as shown in Fig. 1. We use a color-filled circle to highlight the case Solve range. Cases c_1, c_2, c_3, c_4, c_5 are located within the Solve range of c_1, which is highlighted by a yellow circle. According to Definition 1, we have $\mathrm{CoverageSet}(c_1) = \{c_1, c_2, c_3, c_4, c_5\}$.

Definition 2. [11] For a case-base $\mathrm{CB} = \{c_1, c_2, \ldots, c_n\}$, given a case $c \in \mathrm{CB}$:

$$\mathrm{ReachabilitySet}(c) = \{c' \in CB : \mathrm{Solve}(c', c)\}$$

For example, as shown in Fig. 1, case c_1 is located within the Solve range of cases c_1, c_2, c_3, c_4, c_5. Therefore, $\mathrm{ReachabilitySet}(c_1) = \{c_1, c_2, c_3, c_4, c_5\}$.

Definition 3. [12] For a case-base $\mathrm{CB} = \{c_1, c_2, \ldots, c_n\}$, given a case $c \in \mathrm{CB}$, $\mathrm{RelatedSet}(c)$ is defined as:

$$R^{\mathrm{CB}}(c) = \mathrm{CoverageSet}(c) \cup \mathrm{ReachabilitySet}(c)$$

For given case c_1 in Fig. 1, $R^{\mathrm{CB}}(c_1) = \mathrm{CoverageSet}(c_1) \cup \mathrm{ReachabilitySet}(c_1) = \{c_1, c_2, c_3, c_4, c_5\}$.

Definition 4. [6] For a case-base $\mathrm{CB} = \{c_1, c_2, \ldots, c_n\}$, given a case $c \in \mathrm{CB}$, $\mathrm{RelatedClosure}(c)$ with regard to CB is defined as:

$$\Re^{\mathrm{CB}}(c) = \{R^{\mathrm{CB}}(c_i) : \\ \forall c_i \in \mathrm{CB}, \exists R^{\mathrm{CB}}(c_i) \text{ s.t. } c \in R^{\mathrm{CB}}(c_i)\}$$

For a group of cases $S \subseteq \mathrm{CB}$, $\mathrm{RelatedClosure}(S)$ is defined as:

$$\Re^{\mathrm{CB}}(S) = \cup_{c \in S} \Re^{\mathrm{CB}}(c)$$

As shown in Fig. 1, let $\mathrm{CB} = \{c_1, c_2, c_3, c_4, c_5\}$, $R^{\mathrm{CB}}(c_1) = \{c_1, c_2, c_3, c_4, c_5\}$, $R^{\mathrm{CB}}(c_2) = \{c_1, c_2, c_4, c_5\}$, $R^{\mathrm{CB}}(c_3) = \{c_1, c_3\}$, $R^{\mathrm{CB}}(c_4) = \{c_1, c_2, c_4\}$, $R^{\mathrm{CB}}(c_5) = \{c_1, c_2, c_5\}$. The $\mathrm{RelatedClosure}(c_3)$ with regard to CB is the set of all RelatedSets which contain the case c_3. That is, $\Re^{\mathrm{CB}}(c_3) = \{\{c_1, c_2, c_3, c_4, c_5\}, \{c_1, c_3\}\}$. Let $S = \{c_3, c_4\}$, therefore we have $\Re^{\mathrm{CB}}(S) = \{\{c_1, c_2, c_3, c_4, c_5\}, \{c_1, c_2, c_4, c_5\}, \{c_1, c_3\}, \{c_1, c_2, c_4\}\}$.

To detect concept drift, a group of data S_1 representing past context and a group of data S_2 representing current context are

joined together to form one case-base CB. After modeling all the data in CB into competence, two related closures $\Re^{CB}(S_1)$ and $\Re^{CB}(S_2)$ are separately obtained. By using the *density* of $r \in \Re^{\mathrm{CB}}(S)$, each group of data can be represented their distribution in the competence space. The different between the data of the two groups can be measured by *competence-based empirical distance* $d^{\mathrm{CB}}(S_1, S_2)$. If the measurement $d^{\mathrm{CB}}(S_1, S_2)$ is large enough, which means two groups data are significantly different, drift is signaled. To ensure that $d^{\mathrm{CB}}(S_1, S_2)$ is significantly large and provides a statistical guarantee, the two-sample non-parametric permutation test method [13] is applied.

C. Kernel Density Estimation

Kernel Density Estimation (KDE), introduced by Rosenblatt [14] and Parzen [15], is one of the most popular approaches for estimating probability density function with one random variable. However, while applying KDE for data analysis, the bandwidth selection problem has become more important for density estimation than kernel selection [16].

Definition 5. [15] Let $X_1, X_2, \ldots, X_n$ be independent random variables drawn from distribution with unknown density. For a given point x, the kernel density estimator is defined as follows:

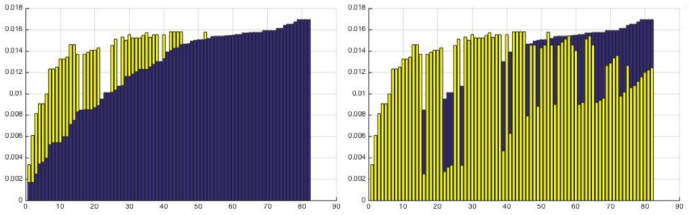
$$\hat{f}(x) = \frac{1}{nh} \sum_{i=1}^{n} K\left(\frac{x - X_i}{h}\right)$$

where $K(\cdot)$ is kernel function, which is a non-negative function that integrates to one and has mean zero, and h is a smoothing parameter called the bandwidth. A range of kernel functions are commonly used, such as normal, uniform, triangular, biweight, triweight, and Epanechnikov.

III. Competence-based discrepancy density estimator

A common method of identifying drift regions is to partition whole space using a data model and then determining the partitions of greatest discrepancy in two groups of data. However, drift regions are irregular in shape and do not fit well with preset space partitions. Therefore, the drift regions highlighted by existing methods cannot match true drift regions.

By utilizing the competence model as a space partition technique, competence-based drift detection method [6] highlight a drift region of the problem space through case-base competence by TPCA [10]. However it ignores the fact that space partitions may overlap one another, and only focuses on competence areas that have a large discrepancy. By introducing KDE, we utilize the overlapping space partitions and invent **Competence-based Discrepancy Density Estimator (CDDE)**, which accurately highlights drift regions in th data feature space with quantitative value. First, we find out the discrepancy distribution of two sliding window data in competence space. We then extract essential information from the discrepancy distribution as the input of KDE, and map the discrepancy distribution in the competence space back to the multi-dimensional data feature space. As mentioned above, when using KDE, the bandwidth of the kernel has a strong influence on the resulting estimate. Therefore, to accurately exhibit a drift region in the data feature space, we carefully select *kernel point* and *kernel bandwidth*, and extend KDE with *discrepancy weight*. The details of CDDE and formal definitions are as follow:



(a) (b)

Fig. 2. An example of competence-based empirical distributions of two sliding window data with concept drift.

Definition 6. [6] Given a case-base CB and the subset of case-base $S \subseteq \mathrm{CB}$. For any $\mathrm{RelatedSet}\, r \in \Re^{\mathrm{CB}}(S)$, denote $\Re = \{r\}$, the *density* of r with regard to S is defined as:

$$w_S(r) = \frac{1}{|S|} \sum_{c \in S} \frac{|\Re \cap \Re^{\mathrm{CB}}(c)|}{|\Re^{\mathrm{CB}}(c)|}$$

where $\Re^{\mathrm{CB}}(S)$ is the $\mathrm{RelatedClosure}$ of S, and $\Re^{\mathrm{CB}}(c)$ is the $\mathrm{RelatedClosure}$ of c.

$\mathrm{RelatedSet}$, as a partition in problem space, is a basic element in competence space. For an intuitive explanation, the *density* of r with regard to S shows the empirical probability of S upon $\mathrm{RelatedSet}\, r$. If we request the *density* of all $\mathrm{RelatedSet}\, r \in \Re^{\mathrm{CB}}(S)$, we will have the empirical distribution of S in the competence space. In Fig. 2, we show the competence-based empirical distribution of two sliding window data in different colors, yellow and blue. The bar located in various positions on the x axis represents different $\mathrm{RelatedSet}$ of competence space. The height of each bar means the *density* of $r \in \Re^{\mathrm{CB}}(S)$. All bars of the same color in the figure demonstrate the empirical distribution of one window of data in the competence space. As shown in the figure, there are significant differences between the empirical distribution of two sliding window data with concept drift.

Definition 7. Given a case-base CB, the subset of case-base $S_1, S_2 \subseteq \mathrm{CB}$ and $\mathrm{RelatedClosure}\, \Re^{\mathrm{CB}}(\mathrm{CB})$. For any $\mathrm{RelatedSet}\, r \in \Re^{\mathrm{CB}}(CB)$, we define the *discrepancy weight* of r between S_1 and S_2 as:

$$dw_{S_1,S_2}(r) = |w_{S_1}(r) - w_{S_2}(r)|$$

where $w_{S_1}(r)$ and $w_{S_2}(r)$ are the *density* of r with regard to S_1 and S_2.

The *discrepancy weight* of r between S_1 and S_2 reveals the difference between two sliding window data S_1 and S_2 in $\mathrm{RelatedSet}\, r$. Like competence-based empirical distribution, we also could acquire the competence-based discrepancy distribution between S_1 and S_2 by calculating the *discrepancy*

weight of all $\mathrm{RelatedSet}\, r \in \Re^{\mathrm{CB}}(CB)$. By mapping this discrepancy distribution from the competence space to the data feature space, we can identify regions where concept drift is believed to be occurring. Since the RelatedSet overlap with one another, the method we use for mapping is KDE.

Definition 8. Given a case-base $\mathrm{CB} = \{c_1, c_2, \ldots, c_n\}$, case $c \in \mathrm{CB}$ is d-dimensional data $\mathbf{x} = (x_1, x_2, \ldots, x_d)$, and $\mathrm{RelatedClosure}\,\Re^{\mathrm{CB}}(\mathrm{CB})$. For any $\mathrm{RelatedSet}\, r \in \Re^{\mathrm{CB}}(CB)$, we define the *kernel point* of r as:

$$\mathbf{X}(r) = \frac{1}{|r|} \sum_{\mathbf{x} \in r} \mathbf{x}$$

where the kernel point $\mathbf{X}(r)$ is the centroid of $\mathrm{RelatedSet}\, r$.

As a kernel point is a basic element in KDE, $\mathrm{RelatedSet}\, r$ can be treated as a kind of "kernel" in competence space. The first step in remapping is to choose an appropriate kernel point to represent $\mathrm{RelatedSet}\, r$ in the data feature space. According to Definition 3, $\mathrm{RelatedSet}\, r$ is a set of cases that have high similarity or are located in a small region. An intuitive and reasonable kernel point is the average of data points in $\mathrm{RelatedSet}\, r$.

Definition 9. Given a case-base $\mathrm{CB} = \{c_1, c_2, \ldots, c_n\}$, case c_i is d-dimensional data $\mathbf{x}_i = (x_i^1, x_i^2, \ldots, x_i^d)$, and $\mathrm{RelatedClosure}\,\Re^{\mathrm{CB}}(\mathrm{CB})$. For any $\mathrm{RelatedSet}\, r \in \Re^{\mathrm{CB}}(CB)$, we define the *kernel bandwidth* of r as:

$$h(r) = 2 \times \max(\{\mathrm{dist}(\mathbf{X}(r), \mathbf{x}) : \mathbf{x} \in r\} \cup \{\theta\})$$
$$\mathbf{H}(r) = \mathrm{diagonal}(h(r), d)$$

where $\mathrm{dist}(\mathbf{X}(r), \mathbf{x})$ means the distance between $\mathbf{X}(r)$ and $\mathbf{x}$ in d-dimensional space, θ is a smooth parameter. $\max(\{\cdot\})$ obtains the maximum value of the given set, and $\mathrm{diagonal}(h(r), d)$ is a $d \times d$ diagonal matrix in which all elements in main diagonal are $h(r)$.

Unlike KDE, which sets a uniform *bandwidth* for every kernel, we carefully select individual *bandwidth* for each kernel. As the way we choose *kernel point*, the *kernel bandwidth* is also related to $\mathrm{RelatedSet}\, r$. The maximum distance between the kernel point and data point in $\mathrm{RelatedSet}\, r$ is chosen as the radius of $\mathrm{RelatedSet}\, r$. Smooth parameter is used to make sure the radius will not be too small, otherwise the result will be dominated by the small *bandwidth*. While the *kernel bandwidth* of r should cover all the data points of $\mathrm{RelatedSet}\, r$, the diameter of $\mathrm{RelatedSet}\, r$ used as *kernel bandwidth* is rational. The data feature space is considered to be d-dimensional, and the *kernel bandwidth* is extended to $d \times d$ matrix to calculate the *discrepancy density*.

Definition 10. For a case-base $\mathrm{CB} = \{c_1, c_2, \ldots, c_n\}$, $\forall c \in \mathrm{CB}$ is d-dimensional data $\mathbf{x} = (x_1, x_2, \ldots, x_d)$, the subset of case-base $S_1, S_2 \subseteq \mathrm{CB}$ and $\mathrm{RelatedClosure}\,\Re^{\mathrm{CB}}(\mathrm{CB}) = \{r_1, r_2, \ldots, r_m\}$. For any $\mathrm{RelatedSet}\, r_i \in \Re^{\mathrm{CB}}(CB), i = 1, 2, \ldots, m$, we define the *discrepancy density estimator* of given data point $\mathbf{x}$ between S_1 and S_2 as:

$$\hat{\mathbb{F}}(\mathbf{x}) = \frac{1}{m \cdot \sum_{j=1}^{m} dw_{S_1,S_2}(r_i)} \sum_{i=1}^{m} \mathbb{K}_i(\mathbf{x}) \cdot dw_{S_1,S_2}(r_i)$$
$$\mathbb{K}_i(\mathbf{x}) = |\mathbf{H}(r_i)|^{-1/2} \cdot K\left(\mathbf{H}(r_i)^{-1/2} \left(\mathbf{X}(r_j) - \mathbf{x}\right)\right)$$

where $\mathbf{X}(r_i)$ is the *kernel point* of r_i, $\mathbf{H}(r_j)$ is the *kernel bandwidth* of r_i, $dw_{S_1,S_2}(r_i)$ is the *discrepancy density* of r_i between S_1 and S_2, and $K(\cdot)$ is the kernel function.

As shown in Definition 10, the *discrepancy density estimator* adopts the idea of KDE with a little variation, which is that, each kernel function is normalized with its corresponding *discrepancy weight*. This variation is used to integrate discrepancy information in the competence space and to show drift-affected discrepancy density in the data feature space.

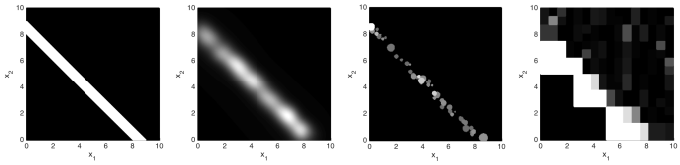
After drift is detected by the competence-based drift detection method, it is easy to deploy the competence-based discrepancy density estimator by utilizing current constructed competence models. There are two stage: 1) preparation; and 2) evaluation. In preparation stage, we first scan all $\mathrm{RelatedSet}$ in $\mathrm{RelatedClosure}\,\Re^{\mathrm{CB}}(\mathrm{CB})$. Each $\mathrm{RelatedSet}$ is transformed to a *kernel point* in KDE, which is achieved by Definition 8. The *bandwidth* of each kernel and its discrepancy weight are calculated by Definitions 9 and 7. In evaluation stage of CDDE, by using the results of the preparation stage, we can obtain the discrepancy density estimate of $\mathbf{x}$ through Definition 10.

IV. Experiments

A. Experiment datasets

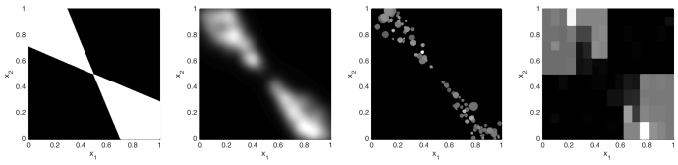
To demonstrate the ability of our proposed to highlight drift regions, we selected two widely-used synthetic datasets, *SEA Concepts* and *Rotating Hyperplane*, to run the experiments. These two datasets can simulate two main types of concept drift: sudden drift and gradual drift. Both of the datasets are configurable for setting up drift regions, which is convenient for evaluation. Most real datasets for the evaluation of learning under concept drift are only used to compare different methods on learning accuracy or error rate. We do not have relevant information about "when", "how" and "where" of drifts have occurred in those real datasets. It is difficult to verify whether our proposed method is effective when it is applied to these real datasets.

SEA Concepts [17]: Data points are uniformly randomly sampled from space $[0, 10]^3$. A data point will be assigned to class 1 if the sum of the first two feature values is not larger than a threshold θ, otherwise it will be assigned to class 0. The third feature is irrelevant as noise. The dataset has four sections with different concepts, each of which has a different threshold θ. The thresholds are set as 8, 9, 7 and 9.5 for each section. Noise is introduced by randomly switching the class of 10% of data. In our experiment settings, we removed the irrelevant third feature and noise. We chose the first two sections of SEA concepts, where the threshold θ varied from 8 to 9. The window size was set to 2000. In summary, two sample sets



(a) true drift regions (b) CDDE (c) TPCA (d) KSSS

Fig. 3. The experiment result of the SEA concepts



(a) true drift regions (b) CDDE (c) TPCA (d) KSSS

Fig. 4. The experiment result of the rotating hyperplane

were set as follows: S_1, representing the past context, was drawn from a distribution with a decision boundary at $x_1 + x_2 = 8$. S_2, representing the current context, was drawn from a distribution with a decision boundary at $x_1 + x_2 = 9$.

Rotating Hyperplane [18]: Data points are uniformly randomly sampled from space $[0, 1]^d$. The drift in this data set is controlled by a hyperplane defined as $\sum_{i=1}^{d} a_i x_i = a_0$, where d is dimension and a_i are randomly initialized weights in the range of $[0, 1]$. The label of each data point is positive if $\sum_{i=1}^{d} a_i x_i < a_0$ or negative if $\sum_{i=1}^{d} a_i x_i \geq a_0$. a_0 is set to $1/2 \sum_{i=1}^{d} a_i$ to guarantee that both parts dividing the hyperplane have similar volume. The concept drift is defined as the weights of dimensions that change over time. The total number of changing dimensions is denoted as K; the magnitude of the changes is denoted as T; the directions of the changes are denoted as $s_i \in \{-1, 1\}, 1 \leq i \leq K$. The concept changes gradually during the arrival of N samples as the weights vary by $s_i \times TN$ after each sample. Also a_0 needs to be recomputed after the weights have been updated. In our experiments setting, we set $d = 2$, $K = 1$, $T = 0.7$, and initially set $a_1 = 0.85$, $a_2 = 0.35$, $s_1 = -1$, $s_2 = 0$. We sampled 4000 data in which the hyperplane gradually rotates from $0.85x_1 + 0.35x_2 = 0.6$ at the beginning to $0.15x_1 + 0.35x_2 = 0.25$ at the end. The first 2000 data are used as S_1, and the remaining 2000 data are used as S_2.

B. Baselines for comparison and measurement for evaluation

To evaluate our proposed drift region highlighting approach, we make a comparison with two other techniques that have the ability to identify drift areas. One is *Top-P-Competence Areas* (TPCA) [10], and the other is *Kulldorff Spatial Scan Statistic* (KSSS) [5]. In TPCA, parameter p indicates the confidence of the identified areas. A smaller p-value results in greater confidence, consequently fewer competence areas will be picked up. In our experiments, we set p to 0.2 since this would highlight the majority drift regions with certain confidence. We use the same method of constructing competence models for fair comparison. The Solve rule used to construct $\mathrm{RelatedSet}$ is a leave-one-out classification. That is, any case c_i is considered to be solved by the retrieved cases that have the same class of c_i bounded by the closest case c_j that has different class of c_i. For KSSS, based on the experiment configuration [5] and our experiment settings, we set *kdq-tree*'s parameter δ to 2^{-10} and τ to 20.

The fit of highlighted drift regions is an important performance measurement for different drift region highlighting techniques. We estimated the mean squared error (MSE) of different techniques to determine which best fit the true drift regions. We defined this estimate of error in terms of the difference between the measurement of estimated drift and the true drift indicator at each grid point, which is uniformly sampled from the whole data feature space, and summed over all grid points:

$$\mathrm{MSE} = \frac{1}{n} \sum_{i=1}^{n} \left(\hat{f}(x) - f(x) \right)^2$$

where n is the number of grid points, x is the grid point, $\hat{f}$ is the true drift indicator at the given grid point, and f is the measurement of estimated drift at the given grid point. For a true drift indicator, we set "1" to grid points located within drift regions, and "0" to grid points located outside drift regions. Because each of the drift region highlighting techniques have different measurement for marking drift regions, we normalized these measurements to $[0, 1]$ before calculating MSE for fair comparison.

C. Results analysis and discussion

Fig. 3(a) and Fig. 4(a) depict the true drift regions between the two window data drawn from SEA and rotating hyperplane, where the bright parts are the true drift regions. The remaining sub-figures of Fig. 3 and Fig. 4 are drawn based on the drift regions highlighted by different techniques. The brightness of these sub-figures is determined by the normalized measurement of each technique. If the normalized measurement is close to 1, the associated region will be brighter, indicating a drift region. The dark region associated with a normalized measurement close to 0 is a non-drift region. From Fig. 3(b) and Fig. 4(b), we can see that the drift regions highlighted by CDDE are very close to true drift regions. We can even tell the rotation direction of the hyperplane in Fig. 4(b), since the regions close to $0.85x_1 + 0.35x_2 = 0.6$ are much brighter than the regions close to $0.15x_1 + 0.35x_2 = 0.25$. TPCA's results are shown in Fig. 3(c) and Fig. 4(c). Almost all the highlighted circles fall in the true drift area, which can be confirmed in later quantitative analysis. One problem that should be addressed is that the highlighted circles are sparse. As a result, some true drift regions are not marked. Even though we can determine a vague outline of drift regions from those scattered circles, the performance of TPCA is not as good as CDDE. Fig. 3(d) and Fig. 4(d) show the result of the KSSS. The result is clearly highly influenced by the space

TABLE I
MEAN SQUARED ERROR (MSE) OF DRIFT REGION HIGHLIGHTING TECHNIQUES ON SEA CONCEPTS

Method	All regions	Drift regions	Non-drift regions
CDDE	**0.0307**	**0.0123**	0.0184
top-p-competence area	0.0631	0.0626	**0.0005**
the Kulldorff statistic	0.1677	0.0286	0.1392

TABLE II
MEAN SQUARED ERROR (MSE) OF DRIFT REGION HIGHLIGHTING TECHNIQUES ON ROTATE HYPERPLANE

Method	All regions	Drift regions	Non-drift regions
CDDE	**0.0931**	**0.0909**	0.0022
top-p-competence area	0.2528	0.2528	**0.0000**
the Kulldorff statistic	0.1486	0.1158	0.0327

partitioning of *kdq-tree*, it also highlights some areas where no drift has occurred.

The three ways are MSE in all regions, only drift regions and only non-drift regions. Table I and Table II give the detail of the MSE results. In both experiments, CDDE achieved the smallest MSE in all regions, demonstrating that it can highlight drift regions more accurately. We can see that TPCA achieved low MSE upon non-drift regions but high MSE in drift regions, which indicates that the drift regions highlighted by TPCA have good precision but low recall. In contrast, KSSS has high MSE in non-drift regions. Although it provides a good measurement of estimated drift, it still highlight some areas in which no drift has occurred.

In this paper, we only present the test results of CDDE on two dimensional feature space, but it is also suitable for handling the problem spaces with higher dimensions. One problem that should be addressed is that since CDDE for highlighting drift areas is based on data samples, more representative samples are needed to improve final result as the problem space gains higher dimensions. The cost of CDDE has two components: the initial construction of the competence models and the discrepancy density estimation. Since we used competence-based drift detection to detect drift first and then applied discrepancy density estimation, it is logical to focus on discrepancy density estimation only. All the components used in the discrepancy density estimation are associated with $\mathrm{RelatedSet}$ in the current CB, so the cost is mainly dependent on the number of $\mathrm{RelatedSet}$ and grows linearly when the number of data samples increases.

V. CONCLUSION AND FUTURE WORKS

In summary, this paper presents an approach for highlighting drift regions in multi-dimensional data feature space without using a space partitioning technique and can provide quantitative discrepancy value, implemented by introducing kernel density estimation into a competence-based drift detection method. Two experiments, containing different types of concept drift, indicate that this approach accurately identifies drift regions. The output of our work, discrepancy density, can be used as additional information for reacting to concept drift or updating an outdated learner. In our future research, we will improve our approach to make it more efficient for dealing with large data samples which have high dimension. Moreover, we will try to utilize our results as an active learning strategy for dealing with the problem of data labels being delayed in real-world applications.

ACKNOWLEDGMENT

The work presented in this paper was supported by the Australian Research Council (ARC) under discovery grant DP150101645.

REFERENCES

[1] G. Widmer and M. Kubat, "Learning in the presence of concept drift and hidden contexts," *Machine Learning*, vol. 23, no. 1, pp. 69–101, 1996.

[2] J. Gama, I. Žliobaitė, A. Bifet, M. Pechenizkiy, and A. Bouchachia, "A survey on concept drift adaptation," *ACM Computer Survey*, vol. 46, no. 4, pp. 1–37, 2014.

[3] J. Gama, P. Medas, G. Castillo, and P. Rodrigues, "Learning with drift detection," in *Advances in Artificial Intelligence–SBIA 2004*. Springer Berlin Heidelberg, 2004, pp. 286–295.

[4] D. Kifer, S. Ben-David, and J. Gehrke, "Detecting change in data streams," in *Proceedings of the Thirtieth International Conference on Very Large Data Bases*, vol. 30. VLDB Endowment, 2004, pp. 180–191.

[5] T. Dasu, S. Krishnan, S. Venkatasubramanian, and K. Yi, "An information-theoretic approach to detecting changes in multi-dimensional data streams," in *Proceedings of the Symposium on the Interface of Statistics, Computing Science, and Applications*. Citeseer, 2006.

[6] N. Lu, G. Zhang, and J. Lu, "Concept drift detection via competence models," *Artificial Intelligence*, vol. 209, pp. 11–28, 2014.

[7] M. Baena-García, J. del Campo-Ávila, R. Fidalgo, A. Bifet, R. Gavaldà, and R. Morales-Bueno, "Early drift detection method," in *Fourth International Workshop on Knowledge Discovery from Data Streams*, vol. 6, 2006, pp. 77–86.

[8] K. Nishida and K. Yamauchi, "Detecting concept drift using statistical testing," in *Discovery Science*, vol. 4755. Springer Berlin Heidelberg, 2007, pp. 264–269.

[9] G. J. Ross, N. M. Adams, D. K. Tasoulis, and D. J. Hand, "Exponentially weighted moving average charts for detecting concept drift," *Pattern Recognition Letters*, vol. 33, no. 2, pp. 191–198, 2012.

[10] N. Lu, J. Lu, G. Zhang, and R. L. de Mantaras, "A concept drift-tolerant case-base editing technique," *Artificial Intelligence*, vol. 230, pp. 108–133, 2016.

[11] B. Smyth and M. Keane, "Remembering to forget: A competence-preserving case deletion policy for case-based reasoning systems," in *Proceedings of the 14th International Joint Conference on Artificial Intelligence*, 1995, pp. 377–382.

[12] B. Smyth and E. McKenna, "Modelling the competence of case-bases," in *Advances in Case-Based Reasoning*. Springer Berlin Heidelberg, 1998, vol. 1488, pp. 208–220.

[13] B. Efron and R. J. Tibshirani, *An Introduction to the Bootstrap*. CRC Press, Boca Raton, 1994.

[14] M. Rosenblatt, "Remarks on some nonparametric estimates of a density function," *The Annals of Mathematical Statistics*, vol. 27, no. 3, pp. 832–837, 1956.

[15] E. Parzen, "On estimation of a probability density function and mode," *The Annals of Mathematical Statistics*, pp. 1065–1076, 1962.

[16] B. W. Silverman, *Density Estimation for Statistics and Data Analysis*. CRC Press, Boca Raton, 1986.

[17] W. N. Street and Y. Kim, "A streaming ensemble algorithm (sea) for large-scale classification," in *Proceedings of the seventh ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 2001, pp. 377–382.

[18] G. Hulten, L. Spencer, and P. Domingos, "Mining time-changing data streams," in *Proceedings of the Seventh ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 2001, pp. 97–106.