Fuzzy Competence Model Drift Detection for Data-driven Decision Support Systems

Fan Dong[a,b], Guangquan Zhang[b], Jie Lu[b], Kan Li[a,*]

[a]*School of Computer Science & Technology, Beijing Institute of Technology, Beijing 100081, China*
[b]*Decision Systems & e-Service Intelligence (DeSI) Lab, Centre for Artificial Intelligence (CAI), University of Technology Sydney, Ultimo, NSW 2007, Australia*

Abstract

This paper focuses on concept drift in business intelligence and data-driven decision support systems (DSSs). The assumption of a fixed distribution in the data renders conventional static DSSs inaccurate and unable to make correct decisions when concept drift occurs. However, it is important to know when, how, and where concept drift occurs so a DSS can adjust its decision processing knowledge to adapt to an ever-changing environment at the appropriate time. This paper presents a data distribution-based concept drift detection method called fuzzy competence model drift detection (FCM-DD). By introducing fuzzy sets theory and replacing crisp boundaries with fuzzy ones, we have improved the competence model to provide a better, more refined empirical distribution of the data stream. FCM-DD requires no prior knowledge of the underlying distribution and provides statistical guarantee of the reliability of the detected drift, based on the theory of bootstrapping. A series of experiments show that our proposed FCM-DD method can detect drift more accurately, has good sensitivity, and is robust.

Keywords: Concept drift, data-driven decision making, fuzzy sets theory, competence model

*Corresponding author

Email addresses: fan.dong@student.uts.edu.au (Fan Dong), guangquan.zhang@uts.edu.au (Guangquan Zhang), jie.lu@uts.edu.au (Jie Lu), likan@bit.edu.cn (Kan Li)

1. Introduction

Various information systems have been widely used in previous decades, resulting in the generation of a tremendous amount of unprocessed streaming data that is waiting to be excavated and analyzed. Concept drift is a big challenge that must be faced for current business intelligence and data-driven decision support systems (DSSs) when processing streaming data. It refers to unpredictable changes that may occur over time in the underlying data distribution [1]. Current machine learning and DSS research still assumes that historical data and new data are drawn from a fixed yet unknown distribution [2], and they are not adaptive; they do not have the ability to detect or even react to concept drifts. When concept drift occurs, the patterns excavated from past data may be outdated and may not be suitable to apply to the new incoming data; hence, a static DSS will make poor decisions leading to a decrease in prediction accuracy [3]. The phenomenon of concept drift is very pervasive in many decision-related, real-world applications, such as user interest change in recommender systems, the emergence of new types of spam in email filtering systems, or the evolution of fraud methods in electronic transactions, to name a few [4]. Therefore, adaptive DSSs that can handle concept drift are extremely important and urgently needed.

In the research area of machine learning, the problem of concept drift means that the statistical properties of the target variable change over time in uncertain ways. Concept drift can formally be described as follows: we denote the data feature vector as $\mathbf{x}$ and the corresponding data class label as y; therefore, streaming data will be an infinite sequence of $(\mathbf{x}, y)$ with time stamp t. The joint distribution of $(\mathbf{x}, y)$ at time stamp t is denoted as $P_t(\mathbf{x}, y)$. A concept drift between time stamp t_0 and time stamp t_1 can be written as $\exists \mathbf{x} : P_{t_0}(\mathbf{x}, y) \neq P_{t_1}(\mathbf{x}, y)$, where t_0 refers to the time stamp before the concept drifts, and t_1 refers to the time stamp after the concept drift [5]. Concept drift can be categorized into three common types: sudden drift, gradual drift, and re-occurring drift [6, 7]. Minku et al. [8] presented another drift category based on multiple criteria: drift speed, severity, predictability, frequency, and recurrence. Since the joint distribution of $(\mathbf{x}, y)$ may evolve with time, well-trained static learning models may lose accuracy and become outdated. At present, approaches for coping with concept drift can be categorized into two learning modes: incremental adaptation and drift detection with retraining [5]. Incremental adaptation approaches [2, 9, 10, 11, 12] continuously adjust the current model using the most recently available data to track concept

drift. They might also use a forgetting mechanism, such as instance selection or instance weighting [13, 14], to discard outdated patterns. Drift detection retraining approaches [15, 16] usually adopt a lazy learning strategy whereby drift detection techniques monitor whether concept drift is occurring. When a drift is detected, the current outdated learning model is discarded, and a new model is retrained with recent data representing the new concept.

However, due to the lack of explicit drift detection techniques, incremental adaptations may encounter a significant performance decrease preventing timely adaptation after a drift has occurred [17]. As well-built DSSs need to quickly react to environmental changes to make correct decisions, adaptive DSSs are better suited to cope with drift detection techniques. Most popular drift detection techniques are usually achieved by a statistical test that monitors the output (error) of the learner [18, 19, 20, 21], the parameters of the learner [22], or the raw data distribution [23, 24, 25, 26, 17]. The drift detection methods that monitor the output and the parameters of the learner can trigger a signal when a concept drift occurs; however, they cannot provide further information about the concept drift. While, by monitoring variations in the data distribution, data distribution-based drift detection methods can reveal information that describes how and where the concept drift has occurred. Information that describes when, how, and where concept drift has occurred benefits DSSs when adapting to changing environments by adding, modifying, and deleting their problem process knowledge accordingly.

Motivated by these issues, we propose a novel method called fuzzy competence model drift detection (FCM-DD). FCM-DD can indirectly measure the change of data distribution via its fuzzy competence model. It requires no prior knowledge of the raw data distribution. In addition to triggering a signal when concept drift occurs, our proposed method can also describe the degree of concept drift and identify where it occurs within the raw data distribution. By introducing fuzzy sets theory, we can improve the competence model partition from crisp boundaries to fuzzy boundaries to estimate a more refined empirical distribution. This makes our proposed drift detection more sensitive and robust, and it achieves better results as shown in our series of experiments.

The rest of this paper is organized as follows. Section 2 discusses related works. Section 3 presents our proposed FCM-DD method. Section 4 describes and discusses the results of the experimental evaluation. Section 5 concludes this study with suggestions for future work.

2. Related work

The problem of data distribution-based concept drift detection is briefly described in Section 2.1. Section 2.2 presents a general framework of data distribution-based concept drift detection methods, and a review of the established literature on data distribution-based drift detection methods. The limitations of the current research are also identified.

2.1. Problem description of data distribution-based concept drift detection

A generic data distribution-based concept drift detection can be formulated as follows. Let $\mathbf{x}_1, \mathbf{x}_2, \ldots$ be infinite streaming data, where each data point $\mathbf{x}_i = (x_1, x_2, \ldots, x_d)$ belongs to a d-dimensional feature space $\mathbb{R}^d$. Concept drift detection works by making a comparison between two sets of data points in different time windows; one time window, W_t represents the previous environment, while $W_{t'}$ represents the current environment. The window $W_t = \{\mathbf{x}_{t-n+1}, \ldots, \mathbf{x}_t\}$ contains a successive sequence of data points ending at $\mathbf{x}_t$ of size n. We assume the data points within a window are independent random samples drawn from two unknown, multi-dimensional, non-parametric distributions. Let F_t and $F_{t'}$ be unknown distributions of each time window W_t and $W_{t'}$, respectively. The goal of concept drift detection is to use a statistical test to verify whether F_t and $F_{t'}$ are identical. The null hypothesis $H_0 : F_t = F_{t'}$ means no concept drift; thus, the alternative hypothesis $H_1 : F_t \neq F_{t'}$ means concept drift occurs between time window W_t and $W_{t'}$. The significance level α is the probability of making an error that F_t and $F_{t'}$ are different when, in fact, they are identical.

2.2. Data distribution-based concept drift detection methods

Data distribution-based drift detection methods have the advantage of sensitive detection and valuable output knowledge (when, how and where concept drift occurs) since they use raw data points directly instead of using indirect abstract information (the output or the parameters of the learner). According to the problem description in Section 2.1, a general data distribution-based concept drift detection framework consists of three modules, as shown in Figure 1: 1) data modeling; 2) divergence measurement; and 3) statistical significance test.

In general, data from real-world applications rarely hold any specific distribution, and it is unrealistic to obtain an unknown non-parametric distribution F_t directly from the raw data points W_t. Most drift detection

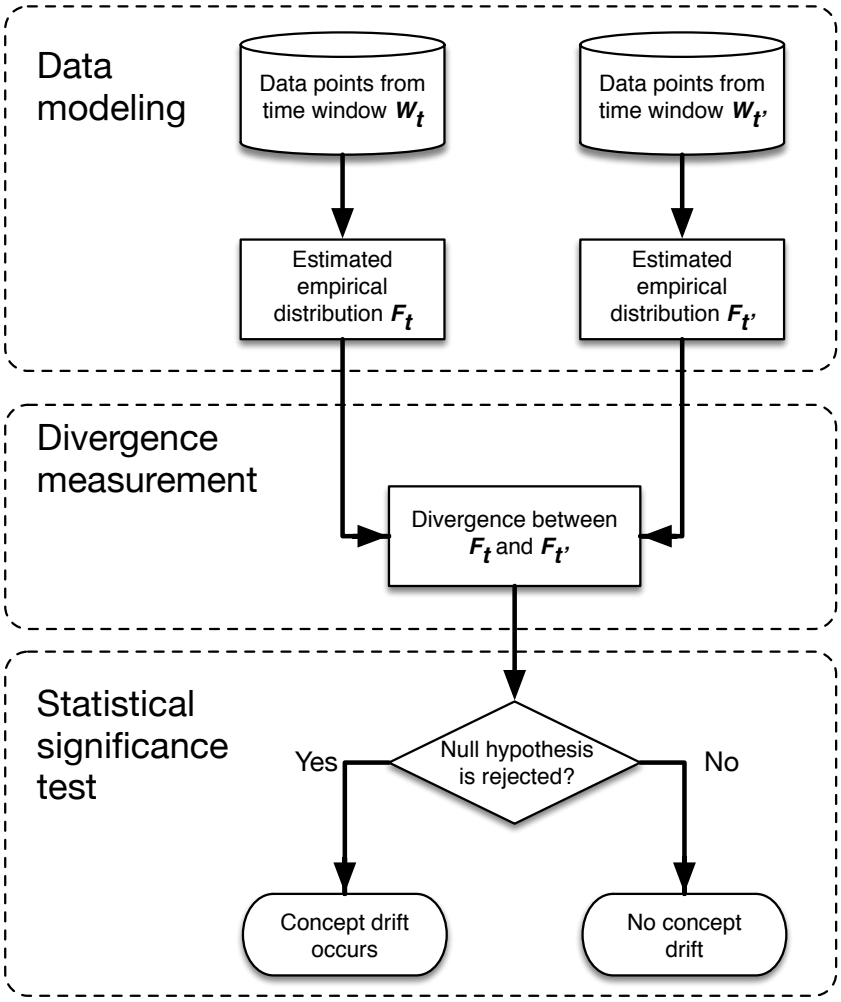

Figure 1: A general data distribution-based concept drift detection framework

methods use data modeling to estimate non-parametric empirical distribution $\hat{F}_t$. Gama et al. [24] used a decision tree to model raw data points; the class distribution they monitored is reflected on the tree's leaves. Dasu et al. [25] claimed to use a *kdq*-tree as a space partitioning scheme, where the empirical distribution is derived from the counts of leaf nodes. Sebastiao and Gama [26] used a histogram constructed from data points to represent empirical probability distribution. Lu et al. [17] modeled the raw data points

through a competence model which is used in case-based reasoning systems. The empirical distribution is reflected upon its competence closure.

A divergence measurement is a quantified value to describe the distance between two probability distributions. In the drift detection methods area, it is used to quantify the similarity of estimated empirical distributions $\hat{F}_t$ and $\hat{F}_{t'}$. Usually, a lower distance value indicates that two probability distributions have more similarity. Kifer et al. [23] employed a notation of $\mathcal{A}$-distance to measure the difference between the two distributions, which, in fact, is a relaxation of the total variation distance. Lu et al. [17] constructed a competence-based empirical distance to show the difference between two data windows, which related the total variation distance to the 1-norm of the difference between the two probability distributions. Other change detection methods use information-theoretic distance. For example, Kullback-Leibler divergence, also called relative entropy, was used in two studies [25, 26]. Vorburger and Bernstein [27] presented an entropy-based metric to measure the distribution inequality between two data windows, where an entropy measure of 1 indicates that the distributions are equal, and a measure of 0 means they are totally different.

A statistical significance test is applied to determine whether the divergence measurement is statistically significant. One solution is to use the Chernoff bound [23], which has several reported advantages, such as being able to control the rate of false alarm (false positive) and missed detection (false negative). The bootstrap method [28] is another approach to determining the significance of a test statistic. It has the advantage of being applied to a problem that has an unknown, complicated, and non-parametric distribution, which suits concept drift detection. Dasu et al. [25] applied a percentile bootstrap method [28] on one estimated empirical distribution $\hat{F}_t$ to construct a critical region. If the divergence measurement of $\hat{F}_t$ and $\hat{F}_{t'}$ falls into this region, they considered that H_0 to be invalidated; thus, concept drift occurred. Lu et al. [17] adopted a two-sample non-parametric permutation test method [28] to approximate the achieved significance level, or the p-value of the test. If the probability of observing a divergence measurement is less than the user-defined significance level, then a concept drift has occurred.

All mentioned drift detection methods have different advantages and limitations. Kifer et al.'s [23] work still contains some technical challenges when applying it to higher-dimensional data environments. Dasu et al.s [25] work has the ability to locate the drift regions with the greatest differences though

Kulldorffs spatial scan statistic, while the space partition made by *kdq*-tree does not guarantee that the regions of greatest change will coincide with true and interesting concepts; therefore, the partition may not be easily explained and understood. Nevertheless, Lu et al.s [17] work achieved better results because of share distribution contributions, although the way they used crisp boundaries to model data and estimate rough empirical distribution could be improved.

3. Fuzzy competence model drift detection

A novel solution for concept drift detection, called fuzzy competence model drift detection (FCM-DD), is proposed in this paper. An overview and the top-algorithm of FCM-DD are presented in Section 3.1. FCM-DD analyses and compares the two data distributions thought empirical distribution constructed by fuzzy competence model instead of the original data feature space, which is presented in Section 3.2. The fuzzy competence-based empirical distance is the divergence measurement between the two data samples, and this is discussed in Section 3.3. Lastly, we apply a permutation test for the divergence measurement to provide statistical guarantee of the detected concept drift in Section 3.4.

3.1. Overview

This section begins with an overview of our drift detection method and a presentation of the main mechanism of the method without mentioning specific details.

Recall the concept drift detection problem description in Section 2.1. The purpose of drift detection is to monitor whether the data distribution of new incoming data differs from the previous environment. In general, we use a time window $W_t = \{\mathbf{x}_{t-n+1}, \ldots, \mathbf{x}_t\}$ that consists of data points $\mathbf{x}$ of size n representing the previous environment, and another time window $W_{t'}$ to representing the current environment, where the time stamp $t' > t$. If the underlying data distributions of W_t and $W_{t'}$ have a statistically significant difference, a concept drift has occurred.

Concept drift may occur in different data scales. Therefore, in practice, we could apply our drift detection with different-sized windows in parallel to detect any potential drifts. The size of the window could be user defined with meaningful parameters, such as weekly or monthly periods, and each

occurrence of concept drift detected in the different windows could be processed independently. The following example using a fixed window size n makes this context clear.

The strategies of sliding window have the advantage of being able to detect different types of drift. There are two kinds of window strategies [25]:

1. *adjacent windows*: where two adjacent windows, W_{t-n} and W_t are moved together when new data arrives; t denotes the current time stamp. This strategy is specifically designed to detect sudden drift, i.e., drift that occurs over a short period of time.
2. *fix-slide windows*: where the first reference window W_n is fixed at time stamp n without movement and the second window W_t slides when new data arrives. After a concept drift is discovered, the reference window is fixed to the current time stamp, and the process is repeated, with the next incoming window becoming the second window. This strategy is designed to detect gradual drift, i.e., smaller changes that accumulate over time.

Data points in the time windows W_t and $W_{t'}$ are subject to the unknown non-parametric distributions F_t and $F_{t'}$ accordingly. Since it is unrealistic to obtain the two distributions of F_t and $F_{t'}$, we use a fuzzy competence model to perform a space partition and obtain the estimated empirical distributions $\hat{F}_t$ and $\hat{F}_{t'}$. They are discussed in Section 3.2. We then compute the fuzzy competence-based empirical distance $\hat{d} = \text{dist}_{\text{FCM}}(\hat{F}_t, \hat{F}_{t'})$ between $\hat{F}_t$ and $\hat{F}_{t'}$. $\hat{d}$ is the divergence measurement between the two windows of data. Section 3.3 provides more details on how to compute the fuzzy competence-based empirical distance $\hat{d}$.

The next step is to determine whether this measurement is statistically significant. Our hypothesis test asserts null hypothesis of $H_0 : \hat{F}_t = \hat{F}_{t'}$. We wish to determine the probability of observing the value $\hat{d}$ if H_0 is accepted. A permutation test is repeated K times to approximate the achieved significance level (ASL) of the null hypothesis and obtain the threshold of the distance measurement d_{ASL}. If $\hat{d} > d_{\text{ASL}}$, H_0 is rejected with a false positive rate at the desired significance level α. This is described in more details in Section 3.4.

The top-algorithm of FCM-DD described above is summarized in Algorithm 1.

Algorithm 1 The top-algorithm of FCM-DD for a fixed window size

Require:

data streams $\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_i, \ldots$
window size n
repeat time of permutation test K
desired significance level α

$t \leftarrow n$
$t' \leftarrow 2n$
construct data windows $W_t = \{\mathbf{x}_1, \ldots, \mathbf{x}_n\}$ and $W_{t'} = \{\mathbf{x}_{n+1}, \ldots, \mathbf{x}_{2n}\}$
while not reach the end of data steam **do**
    estimate $\hat{F}_t$ of W_t by fuzzy competence model (Algorithm 2)
    estimate $\hat{F}_{t'}$ of $W_{t'}$ by fuzzy competence model (Algorithm 2)
    $\hat{d} \leftarrow \mathrm{dist}_{\mathrm{FCM}}(\hat{F}_t, \hat{F}_{t'})$
    $d_{\mathrm{ASL}} \leftarrow K$ times of permutation test with α (Algorithm 3)
    if $\hat{d} > d_{\mathrm{ASL}}$ **then**
        report a concept drift
    end if
    move two windows W_t and $W_{t'}$ with desired strategy.
end while

3.2. Fuzzy competence model

Data in real-world applications are usually multi-dimensional and subject to an unknown non-parametric distribution. Therefore, obtaining the true distribution of data is unrealistic. One solution is to use a space partition technique that constructs an empirical probability distribution, which is a maximum likelihood estimation of the true distribution. This section explains how a fuzzy competence model is used as a space partition technique to estimate an empirical distribution.

Competence is a measurement of how well a case-based reasoning (CBR) system fulfills its goal. Competence is usually taken to be the proportion of problems at hand that can be solved successfully [29]. Since it measures the solving capabilities of a CBR system as a proportion of problem spaces, the probability distribution change of its data should also be a reflection of its competence [17]. Inspired by this, Lu et al. [17] measured the distribution of change of data with regard to its competencies.

The formal definitions for modeling data in terms of competence are given below:

Definition 1. [30] For a set of data $X = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n\}$, any $\mathbf{x}_i \in X$:

$$\text{CoverageSet}(\mathbf{x}_i) = \{\mathbf{x}_j \in X : \mathit{Solve}(\mathbf{x}_i, \mathbf{x}_j)\} \tag{1}$$

where $\mathit{Solve}(\mathbf{x}_i, \mathbf{x}_j)$ indicates a *Solve* rule that $\mathbf{x}_i$ can be retrieved and adapted to solve $\mathbf{x}_j$.

A *Solve* rule can be defined in different ways. For example, data $\mathbf{x}_i$ can solve data $\mathbf{x}_j$ if their distance is less or their similarity is greater than a threshold d_ε; or in a classification task, the data $\mathbf{x}_i$ and $\mathbf{x}_j$ have the same label and they are k-nearest neighbors.

Definition 2. [30] For a set of data $X = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n\}$, any $\mathbf{x}_i \in X$:

$$\text{ReachabilitySet}(\mathbf{x}_i) = \{\mathbf{x}_j \in X : \mathit{Solve}(\mathbf{x}_j, \mathbf{x}_i)\} \tag{2}$$

Definition 3. [31] For a set of data $X = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n\}$, any $\mathbf{x}_i \in X$:

$$\text{RelatedSet}(\mathbf{x}_i) = \text{CoverageSet}(\mathbf{x}_i) \cup \text{ReachabilitySet}(\mathbf{x}_i) \tag{3}$$

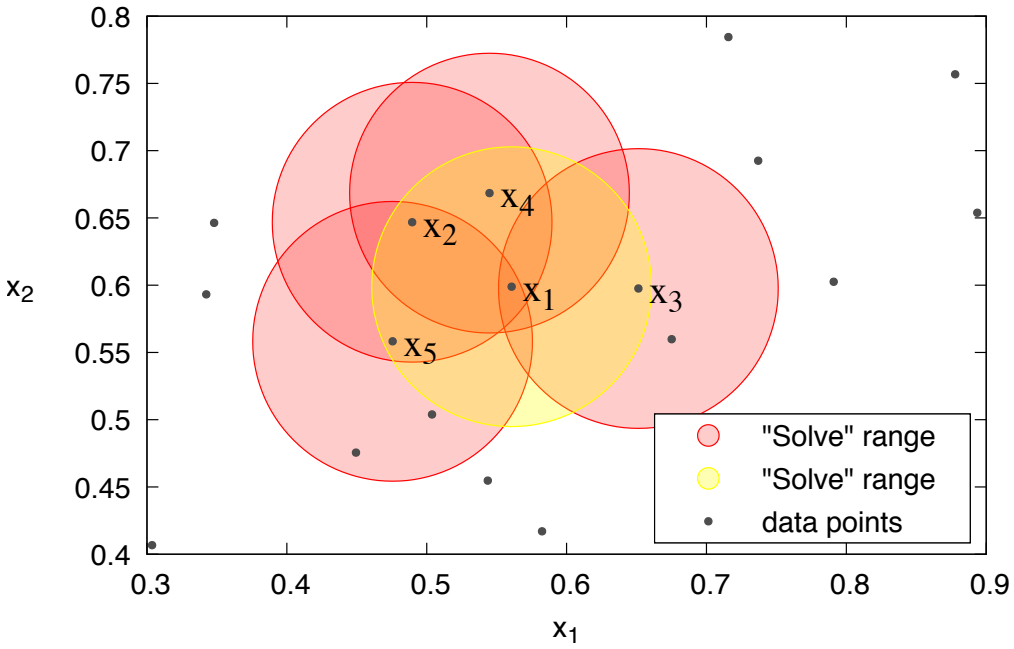

Figure 2: An example of competence model.

To illustrate the Definitions 1, 2, and 3, we give an example to show the CoverageSet, ReachabilitySet, and RelatedSet. The *Solve* rule is defined as follows: two data can mutually solve each other if the Euclidean distance between them is less than $d_\varepsilon = 0.1$. As shown in Figure 2, color-filled circles highlight the *Solve* ranges of the data. Data $\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3, \mathbf{x}_4, \mathbf{x}_5$ are located within the *Solve* range of $\mathbf{x}_1$, which is highlighted by a yellow circle whose center is $\mathbf{x}_1$. According to Definition 1, we have a CoverageSet$(\mathbf{x}_1) = \{\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3, \mathbf{x}_4, \mathbf{x}_5\}$. In addition, we have a ReachabilitySet$(\mathbf{x}_1) = \{\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3, \mathbf{x}_4, \mathbf{x}_5\}$ and a RelatedSet$(\mathbf{x}_1) = \{\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3, \mathbf{x}_4, \mathbf{x}_5\}$. The essential of a RelatedSet$(\mathbf{x}_i)$ is a sphere defined in Euclidean distance space of X. The center of the sphere is $\mathbf{x}_i$, and the radius of the sphere is d_ε.

In [17], the empirical distribution is estimated using RelatedSets of competence model, which is a kind of space partition techniques. Thus, there are n overlapped spheres defined in the original data feature space. Each data $\mathbf{x}_i$ is the center of the sphere, and the radius d_ε of the sphere is defined by the *Solve* rule. A competence-based empirical weight can then be used to represent the estimated distribution of data over the RelatedSets

of competence model. There are two advantages of partitioning the space with this competence model: 1) It compresses the original multi-dimensional data feature space into a 1-dimensional discrete data distance space without losing information about the relationship between the two data. 2) Rather than splitting the space simply by cutting its edges, the overlapping spheres generate a tolerable sample bias [17]. However, the crisp boundaries used to build the RelatedSet will result in a roughly estimated distribution. Therefore, we introduce fuzzy sets theory [32] to provide a more refined empirical distribution.

Definition 4. For a set of data $X = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n\}$, any $\mathbf{x}_i \in X$, a fuzzy RelatedSet FR_i is defined as:

$$\mathrm{FR}_i = \{\mathbf{x}_j, \mu_{\mathrm{FR}_i}(\mathbf{x}_j) \mid \mathbf{x}_j \in X\} \tag{4}$$

$\mu_{\mathrm{FR}_i}(\mathbf{x}_j)$ is membership function with the form as follows:

$$\mu_{\mathrm{FR}_i}(\mathbf{x}_j) = \exp(\frac{-\|\mathbf{x}_i - \mathbf{x}_j\|^2}{2d_\varepsilon^2}) \tag{5}$$

where $\|\cdot\|$ stands for Euclidean distance and d_ε is the threshold defined by *Solve* rule.

By using fuzzy competence model, the original data feature space is mapped into a discrete space with n alphabets. Each alphabet in that discrete space is associated with a fuzzy RelatedSet. The next step is to represent the empirical distribution of data points upon that discrete space.

Definition 5. For a set of data $X = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n\}$, any $\mathbf{x}_j \in X$, the fuzzy competence-based density vector $\mathbf{w}(\mathbf{x}_j)$ is defined as:

$$\mathbf{w}(\mathbf{x}_j) = (\frac{\mu_{\mathrm{FR}_1}(\mathbf{x}_j)}{\sum_{i=1}^n \mu_{\mathrm{FR}_i}(\mathbf{x}_j)}, \frac{\mu_{\mathrm{FR}_2}(\mathbf{x}_j)}{\sum_{i=1}^n \mu_{\mathrm{FR}_i}(\mathbf{x}_j)}, \ldots, \frac{\mu_{\mathrm{FR}_n}(\mathbf{x}_j)}{\sum_{i=1}^n \mu_{\mathrm{FR}_i}(\mathbf{x}_j)}) \tag{6}$$

The fuzzy competence-based density vector of $\mathbf{x}_j$ is a vector with n-dimensional; each dimension equals a normalized membership value of $\mathbf{x}_j$ in fuzzy RelatedSet FR_i. In addition, given a $\mathbf{w}(\mathbf{x}_j) = (w_1, w_2, \ldots, w_n)$, the sum of each dimension $\sum_{i=1}^n w_i$ equals to 1. The empirical distribution of data points depends on these fuzzy competence-based density vectors.

Definition 6. For a set of data $X = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n\}$, given a subset $S \subseteq X$, the fuzzy competence-based empirical vector $\mathcal{P}(S)$ is defined as:

$$\mathcal{P}(S) = \frac{1}{|S|} \sum_{\mathbf{x}_i \in S} \mathbf{w}(\mathbf{x}_i) \tag{7}$$

The fuzzy competence-based empirical vector of a set of data points is calculated by adding the fuzzy competence-based density vector of each data point and then dividing the number of data points. The sum of each dimension equals 1. This vector can be treated as the discrete distribution of the data points on fuzzy competence model.

Returning to our drift detection problem, the procedure to obtain the fuzzy competence-based empirical distributions of the two data time windows W_t and $W_{t'}$ are summarized in Algorithm 2:

Algorithm 2 Fuzzy competence-based empirical distribution estimation

Require:

two data time windows W_t and $W_{t'}$
threshold d_ε defined by *Solve* rule

1: union to data time windows to $W = W_t \cup W_{t'}$
2: construct all fuzzy RelatedSets of W by Definition 4.
3: for all $\mathbf{x}_i \in W$, calculate each $\mathbf{w}(\mathbf{x}_i)$ by Definition 5
4: $\hat{F}_t \leftarrow \mathcal{P}(W_t)$ by Definition 6
5: $\hat{F}_{t'} \leftarrow \mathcal{P}(W_{t'})$ by Definition 6

3.3. Fuzzy competence-based empirical distance

In the previous subsection, we obtained the estimated empirical discrete distributions $\hat{F}_t$ and $\hat{F}_{t'}$ using fuzzy competence model. Therefore, the divergence measurement of $\hat{F}_t$ and $\hat{F}_{t'}$ can be determined by measuring the distance between two discrete distributions. In statistics and information theory, f-divergence is the largest and most frequently used form of divergences. An f-divergence is a function $D_f(P\|Q)$ that measures the difference between two probability distributions P and Q. Many common divergences, such as total variation distance, Kullback-Leibler divergence and Hellinger distance are special cases of an f-divergence [33]. In our drift detection method, we chose total variation distance as our divergence measurement,

since it is easy, symmetric, and satisfies the triangle inequality. The fuzzy competence-based empirical distance is defined as follows:

Definition 7. For a set of data $X = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n\}$, given two subsets $S_1, S_2 \subset X$, $\mathcal{P}(S_1) = (p_1, \ldots, p_n)$ and $\mathcal{P}(S_2) = (q_1, \ldots, q_n)$ are their respective fuzzy competence-based empirical vectors, the fuzzy competence-based empirical distance $\mathrm{dist}_{\mathrm{FCM}}(\mathcal{P}(S_1), \mathcal{P}(S_2))$ is defined as:

$$\mathrm{dist}_{\mathrm{FCM}}(\mathcal{P}(S_1), \mathcal{P}(S_2)) = \frac{1}{2}\|\mathcal{P}(S_1) - \mathcal{P}(S_2)\|_1 = \frac{1}{2}\sum_{i=1}^{n}|p_i - q_i| \qquad (8)$$

Note that, the fuzzy competence-based empirical distance $\mathrm{dist}_{\mathrm{FCM}}$ compares the distance between two datasets through their fuzzy competencies, rather than their real distributions. It is also a measurement that describes how the concept drift has occurred. If $\mathrm{dist}_{\mathrm{FCM}}(\mathcal{P}(S_1), \mathcal{P}(S_2))$ equals 0, it means that two datasets S_1 and S_2 are identical. While if $\mathrm{dist}_{\mathrm{FCM}}(\mathcal{P}(S_1), \mathcal{P}(S_2))$ equals 1, the two datasets are totally different. In addition, by identifying fuzzy RelatedSet FR_i with greater difference $|p_i - q_i|$, we can locate the area where concept drift occurs.

3.4. Hypothesis test for divergence measurement

A divergence measurement that determines the difference between two datasets is only one important achievement in our drift detection method. Another achievement is providing a statistical guarantee of the detected change. Given a divergence measurement $\hat{d}$ between two data empirical distributions $\hat{F}_t$ and $\hat{F}_{t'}$, this issues can be resolved by deriving the probability that $\hat{d}$ can be obtained under the null hypothesis H_0: $\hat{F}_t = \hat{F}_{t'}$. The fuzzy competence-based empirical distance, viewed as a test statistic, has an unknown distribution. Therefore, permutation test [28] is a good way to provide a statistical guarantee of concept drift detection. It has several advantages in several aspects: 1) it is easy to implement; 2) it is free of mathematical assumptions; and 3) it is suitable when the theoretical distribution of the test statistic is unknown.

In our proposed drift detection methods, given two data time windows W_t and $W_{t'}$ and their estimated empirical distributions $\hat{F}_t$ and $\hat{F}_{t'}$, the hypothesis test determines whether $\hat{F}_t$ and $\hat{F}_{t'}$ are identical. Once an observation $\hat{d} = \mathrm{dist}_{\mathrm{FCM}}(\hat{F}_t, \hat{F}_{t'})$ is made, the *achieved significance level* (ASL) of the

observation is the likelihood that $\hat{d}$ appears naturally under H_0:

$$\text{ASL} = P_{H_0}(\hat{d}^* \geq \hat{d}) \tag{9}$$

where $\hat{d}^*$ is a random variable measuring d that assuming the null hypothesis H_0 is true.

The permutation test is, therefore, a clear way to estimate ASL for the null hypothesis and works as follows: We combine all $2n$ data points of W_t and $W_{t'}$, then take a sample of size n data without replacement to represent as S_1^i; the remaining n data are used for S_2^i. We compute the test statistic $d_i = \text{dist}_{\text{FCM}}(\mathcal{P}(S_1^i), \mathcal{P}(S_2^i))$ for each permutation and repeated a large number of times K. Finally, the permutation test is estimated through the Monte Carlo approach [34].

Once the desired ASL α has been established, we let d_{ASL} be the $(1-\alpha)$-percentile of all d_i, where $i = 1, \ldots, K$. The probability of $\hat{d}$ falling into interval (d_{ASL}, ∞) is α. Therefore, if $\hat{d} > d_{\text{ASL}}$, the divergence measurement is statistically significant, H_0 is rejected, and a concept drift has occurred.

The algorithm for obtaining d_{ASL} is summarized in Algorithm 3.

Algorithm 3 The algorithm for obtaining d_{ASL}

Require:

two data time windows W_t and $W_{t'}$
repeat time of permutation test K
desired significance level α

union to data time windows to $W = W_t \cup W_{t'}$
for $i = 1, \ldots, K$ **do**
    $S_1^i \leftarrow$ Sample n data from W without replacement
    $S_2^i \leftarrow W - S_1^i$
    $d_i \leftarrow \text{dist}_{\text{FCM}}(\mathcal{P}(S_1^i), \mathcal{P}(S_2^i))$
end for
$d_{\text{ASL}} \leftarrow (1-\alpha)$-percentile of all d_i

4. Experimental evaluation

To demonstrate the universality of the proposed FCM-DD, our evaluation consisted of two sections and seven experiments on various kinds of datasets.

The datasets used in each experiment were synthesized using standard distributions. The advantage of using these simulated datasets is that the drift patterns can be controlled by carefully choosing different parameters and verifying how the concept drift detection methods reacts to different types of changes.

4.1. Evaluating the accuracy of the fuzzy competence-based empirical distance

In Section 3.3, we introduced fuzzy competence-based empirical distance dist_{FCM} to measure how different distributions are placed from on another. This distance equals 0 if two distributions are identical, and equals 1 if they are totally different. We established three experiments to see how dist_{FCM} varies as the degree of drift changes. Data points in each experiment were generated independently from 1D normal distribution with preset parameters. We compared our results with competence-based empirical distance dist_{CM} [17]. For this series of experiments, we did not use permutation test to detect drift, since our goal was to demonstrate the behavior of fuzzy competence-based empirical distance on changing data.

We considered that two data points were able to mutually solve each other if their Euclidean distance is smaller than a threshold d_ε. Therefore, the radius of RelatedSet is d_ε. In order to eliminate the impact of randomness, all results are calculated as the mean of 100 independent tests.

Experiment 1. Varying the mean. In this experiment, we compared data samples drawn from 11 normal distributions. The ith normal distribution has a fixed standard deviation $\sigma = 0.2$, but with a different mean value $\mu = 0.2 + 0.06 * (i - 1)$, where $i = 1, 2, \ldots, 11$. We did a total $t = 21$ test. For $t = 2 \times i - 1$, we compared two data samples both drawn from the ith distribution; when $t = 2 \times i$, we compared two data samples drawn from the ith distribution and $i + 1$st distribution. Figure 3 shows how the fuzzy competence-based empirical distance reacts as the data distribution changes. dist_{FCM} still has the same peak-valley pattern as dist_{CM} after introducing fuzzy sets theory. Since the peak-valley margin only depends on the means between two adjacent distributions, we found similar height on all peaks for each series. We listed the peak-valley margin of each series for each time stamp, which is shown in Table 1. For a fixed d_ε, dist_{FCM} always has a greater peak-valley margin than dist_{CM}, which means dist_{FCM} is more sensitive than dist_{CM} to the changes.

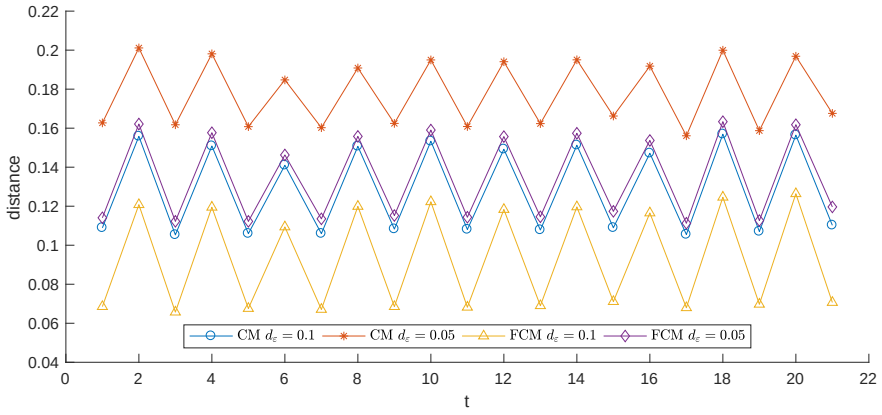

Figure 3: Fuzzy competence-based empirical distance between normal distribution with varying the mean

Table 1: Peak-valley margin of dist_{FCM} and dist_{CM} in Figure 3

t	1,2	3,4	5,6	7,8	9,10	11,12	13,14	15,16	17,18	19,20
dist_{FCM} $d_\varepsilon = 0.05$	**0.0481**	**0.0454**	**0.0340**	**0.0423**	**0.0438**	**0.0413**	**0.0428**	**0.0363**	**0.0521**	**0.0491**
dist_{CM} $d_\varepsilon = 0.05$	0.0384	0.0364	0.0240	0.0306	0.0325	0.0332	0.0327	0.0256	0.0439	0.0381
dist_{FCM} $d_\varepsilon = 0.10$	**0.0523**	**0.0538**	**0.0418**	**0.0528**	**0.0538**	**0.0501**	**0.0507**	**0.0454**	**0.0567**	**0.0567**
dist_{CM} $d_\varepsilon = 0.10$	0.0469	0.0455	0.0350	0.0447	0.0450	0.0411	0.0434	0.0382	0.0514	0.0494

Experiment 2. Varying the standard deviation. In this experiment, we fixed the mean of normal distribution at $\mu = 0.5$, but changed the standard deviation $\sigma = 0.1 + 0.02 \times (i - 1)$ for the ith distribution. Again, we still did a total of $t = 21$ tests. For $t = 2 \times i - 1$, we compared two data samples both drawn from the ith distribution; when $t = 2 \times i$, we compared two data samples drawn from the ith distribution and the $i+1$st distribution. Since the change of standard deviation is hard to reflect on a small sample size, we tested 1000 samples. As shown in Figure 4, dist_{FCM} still has the same peak-valley pattern as dist_{CM}. The peak-valley margin shrinks as σ increases; because the distribution becomes less concentrated so the relative distance is smaller. The specific peak-valley margin for each series on different time stamps is shown in Table 2. For a fixed d_ε, dist_{FCM} has a greater average peak-valley margin than dist_{CM} in larger σ, which means dist_{FCM} is more sensitive than dist_{CM} to small changes. More evidence will be shown in Section 4.2.

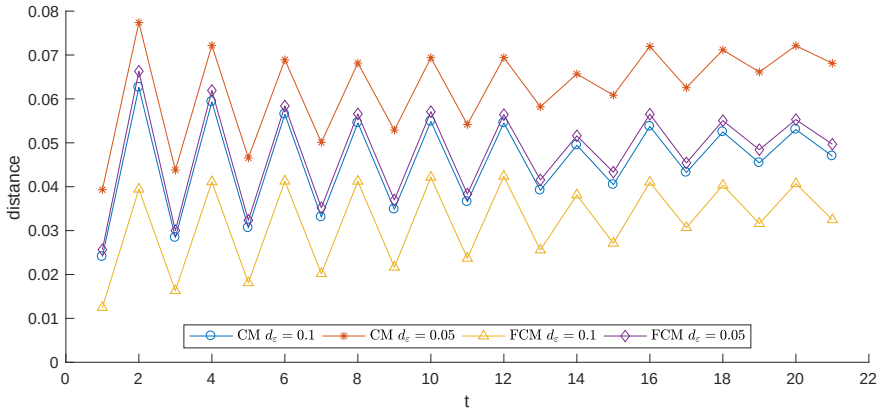

Figure 4: Fuzzy competence-based empirical distance between normal distribution with varying standard deviation

Table 2: Peak-valley margin of dist_{FCM} and dist_{CM} in Figure 4

t	1,2	3,4	5,6	7,8	9,10	11,12	13,14	15,16	17,18	19,20
dist_{FCM} $d_\varepsilon = 0.05$	**0.0407**	**0.0319**	**0.0260**	**0.0214**	**0.0201**	**0.0181**	**0.0100**	**0.0132**	**0.0096**	**0.0069**
dist_{CM} $d_\varepsilon = 0.05$	0.0381	0.0284	0.0223	0.0181	0.0165	0.0152	0.0075	0.0111	0.0086	0.0060
dist_{FCM} $d_\varepsilon = 0.10$	0.0270	0.0248	0.0231	0.0211	**0.0205**	**0.0187**	**0.0126**	**0.0139**	**0.0096**	**0.0091**
dist_{CM} $d_\varepsilon = 0.10$	**0.0386**	**0.0309**	**0.0259**	**0.0215**	0.0200	0.0180	0.0103	0.0133	0.0092	0.0076

Experiment 3. Varying the sample size. In this experiment, we compared distances between two data samples drawn from normal distribution $\mathcal{N}(0.2, 0.2)$ and $\mathcal{N}(0.44, 0.2)$, respectively. We changed the sample size of each test from 100 to 1000. As shown in Figure 5, the trend of series of dist_{FCM} and dist_{CM} are similar and remain steady as the sample size increases. The standard deviation of each series was calculated and is shown in Table 3. With a fixed d_ε, dist_{FCM} has a lower standard deviation than dist_{CM}, which means dist_{FCM} is more robust than dist_{CM} for small samples.

4.2. Evaluating the fuzzy competence model drift detection

In the previous subsection, we demonstrated how fuzzy competence-based empirical distance reacts as the underlying distribution changes. In order to determine if a given measurement is sufficiently statistically significant enough to identify a concept drift, we integrated the permutation test to

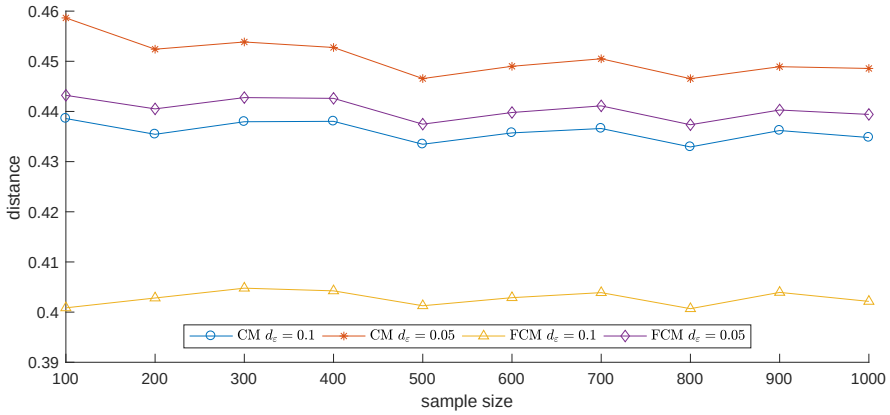

Figure 5: Fuzzy competence-based empirical distance between normal distribution with different sample size

Table 3: Stand deviation of dist_{FCM} and dist_{CM} in Figure 5

	$d_\varepsilon = 0.05$	$d_\varepsilon = 0.10$
dist_{FCM}	**0.0021**	**0.0015**
dist_{CM}	0.0037	0.0015

estimate the d_{ASL} as described in Section 3.4. We report that a concept drift has occurred when $\hat{d} > d_{\text{ASL}}$. The significant level is defined by α, and K indicates the times of repeat permutation test. In the following experiments, we compared our drift detection results with a *Kullback-Leibler divergence*-based drift detector (KLD) [25], a *maximum mean discrepancy*-based kernel two-sample test (MMD) [35], and a competence model drift detector (CM-DD) [17]. As we mentioned in Section 4.1, our proposed method (FCM-DD) is more sensitive and robust than CM-DD. To provide a fair comparison, we used the same experimental setup, including data source, data processing strategy, evaluation criteria and parameter settings.

Data source. For each test, we simulated streaming data consisting of $N = 5,000,000$ data points. The streaming data are divided into 100 groups of $50,000$ data points. Data points within the ith group are drawn from a distribution F_i with parameters $p_1^i, \ldots, p_k^i$. For each group, we simulated a drift by varying a particular set of parameters $(p_{r_1}^i, p_{r_2}^i, \ldots)$ of F_i. Therefore,

we obtained a streaming data with a total of 99 drifts. We denoted a step size Δ to control the degree of drift. When we reached each new group i, for each changing parameter $p^i_{r_j}$, we uniformly chose a random number c in $[-\Delta, -\Delta/2] \cup [\Delta, \Delta/2]$ and added it to the parameter $p^i_{r_j} = p^{i-1}_{r_j} + c$. If the new $p^i_{r_j}$ fell out of a predefined allowable interval $[p^i_{\min}, p^i_{\max}]$, we chose another random number and repeated it. We prepared four basic types of streaming data:

1. $M(\Delta)$ stream: Data samples are drawn from a 2-dimensional normal distribution. Drift is simulated by fixing the standard deviation $\sigma_1 = \sigma_2 = 0.2$ and correlation $\rho = 0.5$ but varying the mean μ_1 and μ_2. The mean μ_1 and μ_2 start at 0.5, and walk randomly in interval $[0.2, 0.8]$ with step Δ as described.
2. $C(\Delta)$ stream: Data samples are drawn from a 2-dimensional normal distribution. Drift is simulated by fixing the mean $\mu_1 = \mu_2 = 0.5$ and $\sigma_1 = \sigma_2 = 0.2$ and but varying the correlation ρ. The correlation ρ starts at 0, and walks randomly in interval $[-1, 1]$ with step Δ as described.
3. $P(\Delta)$ stream: Data samples are drawn from a 2-dimensional discrete Poisson distribution. Drift is simulated by generating data according to $(X, Y) \sim \text{Poisson}(500(1-\rho), 500(1-\rho), 500\rho)$, where ρ starts at 0.5, and the walks randomly in interval $[0, 1]$ with step Δ as described. As we keep the marginal distribution of X and Y the same (with $\lambda = 500$), the drift is reflected on correlation.
4. $\Delta D\_C(0.20)$ stream: Data samples in this stream are generated by extending the $C(0.20)$ stream to a Δ-dimensional stream. The first 2-dimensional data are generated by following the $C(0.20)$ stream rule. We expand it by adding $(\Delta - 2)$-dimensional normal distribution data that do not change. Data in stationary distribution are drawn from $\mathcal{N}(0.5, 0.2)$ without any correlations.

One issue that should be noted is we assume that two window data samples used for drift detection should correctly represent previous and current environments. Data samples that contain missing value or errors should be preprocessed and cleaned. However, data preprocessing is not our main focus.

Data processing strategy. We adopted fix-slide window models: we fixed the first reference time window starting at W_n and the second sliding time window starting at W_{2n}, where n is the window size. We fixed the

first reference window and moved slide window if there was no concept drift reported; otherwise we moved both windows together. Keeping a fixed reference window allows us to capture gradual drift as well as sudden drift. If we both windows are moving together, the change is completely missed after passing the drift point. If the first reference window is fixed, we have a chance of detecting drift at a later time.

Evaluation criteria. We reported four statistics in our drift detection experiments:

1. *True (true positive)*: the number of drifts correctly detected in time, that is, a drift signal alerted before the sliding window has seen the 3rd window of the new data concept group.
2. *Late*: the number of drifts detected late, that is, a drift signal alerted after the sliding window has seen more than two windows of new data concept groups.
3. *False (false positive)*: the number of drifts detected when there is no drift.
4. *Miss (false negative)*: the number of drifts not detected.

Note that there is an inherent trade-off of between power (1 - false positive rate) and sensitive (1 - false negative rate), which will be shown in the following experiments.

Parameter settings. Without stating special experimental settings, we used the same parameters for comparing all methods: window size $n = 10,000$, the times of repeat permutation test or bootstrap samples $K = 500$, desired significance level $\alpha = 0.01$. The parameter d_ε, which is used to construct the competence model and the fuzzy competence model, is chosen empirically according to each experiment with fair comparison as describe in [17]. The parameters used in KLD and MMD have the same settings as mentioned in their respective manuscripts. More specifically, for KLD: the minimum side length of a cell $\delta = 2^{-10}$, the maximum number of data point in a cell $\tau = 100$, and the persistence factor $\gamma = 0.05$. For MMD, a Gaussian RBF kernel was used, and the kernel size σ was set to the median distance heuristic.

Experiment 4. Varying the data sources. In this experiment, we evaluated the performance of our proposed drift detection method on the different streaming data sources, with results shown in Table 4. As we expected, the

performance of the drift detection methods increased as the degree of drift increased. All four methods can detect all drift with small *False* counts within the streaming data: $M(0.05)$, $C(0.20)$, and $P(0.20)$. However, FCM-DD was more powerful than other three methods in showing smaller changes within the streaming data, such as $M(0.02)$, $C(0.15)$, $C(0.10)$, and $P(0.10)$. With equal or fewer *False* counts to CM-DD, FCM-DD is able to detect more drift accurately. Therefore, this proves that FCM-DD is more sensitive than CM-DD to various types of concept drift.

Table 4: Drift detection results on different streaming data sources.

Data stream	Method	d_ε	True	Late	False	Miss
$M(0.05)$	KLD	n/a	98	0	10	1
	MMD	n/a	99	0	6	0
	CM-DD	0.05	99	0	3	0
	FCM-DD	0.05	99	0	9	0
$M(0.02)$	KLD	n/a	71	12	5	16
	MMD	n/a	99	0	7	0
	CM-DD	0.05	92	6	13	1
	FCM-DD	0.05	99	0	7	0
$C(0.20)$	KLD	n/a	96	1	9	2
	MMD	n/a	99	0	11	0
	CM-DD	0.05	96	2	6	1
	FCM-DD	0.05	99	0	11	0
$C(0.15)$	KLD	n/a	85	8	9	6
	MMD	n/a	98	1	3	0
	CM-DD	0.05	91	2	4	6
	FCM-DD	0.05	99	0	4	0

Continued on next page

Table 4 – continued from the previous page

Data stream	Method	d_ε	True	Late	False	Miss
$C(0.10)$	KLD	n/a	31	19	2	49
	MMD	n/a	50	11	2	38
	CM-DD	0.05	41	12	2	46
	FCM-DD	0.05	59	11	5	29
$P(0.20)$	KLD	n/a	96	1	10	2
	MMD	n/a	99	0	9	0
	CM-DD	10	99	0	5	0
	FCM-DD	10	99	0	8	0
$P(0.10)$	KLD	n/a	65	10	4	24
	MMD	n/a	76	5	5	18
	CM-DD	10	84	5	5	10
	FCM-DD	10	89	3	4	7

Experiment 5. Varying the window size. In this experiment, we varied the size of the window to see how it affected the performance of our proposed drift detection method. We chose $C(\Delta)$ data streams since changing correlation ρ is harder to detect, while we can gauge the robustness of each drift detection method. The results are shown in Table 5. As we expected, the performance of drift detection methods works better with larger window sizes because more data points provide a better approximation of the true data distribution. The *True* detect counts of FCM-DD dropped less than CM-DD but had similar *False* counts. For example, in data stream $C(0.15)$, the *True* detect counts of FCM-DD dropped from 99 to 84 as window size varied from $10,000$ to $5,000$, while the *True* detect counts of CM-DD dropped from 91 to 69. Thus, FCM-DD improves its robustness to small sample size through introducing fuzzy sets theory. Even when compared to KLD and MMD, FCM-DD still had higher *True* detect counts and less *Miss* counts.

Table 5: Drift detection results on different window size.

Data stream	Window size	Method	True	Late	False	Miss
$C(0.20)$	10000	KLD	96	1	9	2
		MMD	99	0	11	0
		CM-DD	96	2	6	1
		FCM-DD	99	0	11	0
$C(0.20)$	5000	KLD	89	5	13	5
		MMD	92	5	17	2
		CM-DD	88	10	7	1
		FCM-DD	97	2	10	2
$C(0.15)$	10000	KLD	85	8	9	6
		MMD	98	1	3	0
		CM-DD	91	2	4	6
		FCM-DD	99	0	4	0
$C(0.15)$	5000	KLD	80	7	18	12
		MMD	72	15	6	12
		CM-DD	69	16	14	14
		FCM-DD	84	8	12	7
$C(0.10)$	10000	KLD	31	19	2	49
		MMD	50	11	2	38
		CM-DD	41	12	2	46
		FCM-DD	59	11	5	29
$C(0.10)$	5000	KLD	26	12	4	61
		MMD	35	21	9	43
		CM-DD	35	22	10	42
		FCM-DD	38	21	7	40

Experiment 6. Varying the number of permutation test and significance level. In this experiment, we varied the number of permutation test and significance level to see how they affect the performance. We still ran our experiments with $C(\Delta)$ data streams. As we mentioned above, there is a trade-off between power and sensitivity. We adjusted $K = 100, \alpha = 0.02$ to make drift detection not only more powerful but also more sensitive. As shown in Table 6, the *False* counts increased as K decreased and α increased. In fact, the false positive rate depends on the number of permutation test and significant level due to the nature of the permutation test. We found that FCM-DD has fewer *Miss* counts than CM-DD, but has similar *False* counts. This is further evidence that shows FCM-DD is more sensitive to true drift than CM-DD. However, KLD and MMD are more powerful with lower significance level in data stream $C(0.1)$, but FCM-DD still outperformed them with higher *True* detect counts and the same level of *False* counts.

Table 6: Drift detection results on different number of permutation test and significance level.

Data stream	K	α	Method	True	Late	False	Miss
$C(0.20)$	500	0.01	KLD	96	1	9	2
			MMD	99	0	11	0
			CM-DD	96	2	6	1
			FCM-DD	99	0	11	0
$C(0.20)$	100	0.02	KLD	94	1	19	4
			MMD	99	0	17	0
			CM-DD	96	2	15	1
			FCM-DD	99	0	19	0
$C(0.15)$	500	0.01	KLD	85	8	9	6
			MMD	98	1	3	0
			CM-DD	91	2	4	6
			FCM-DD	99	0	4	0

Continued on next page

Table 6 – continued from the previous page

Data stream	K	α	Method	True	Late	False	Miss
$C(0.15)$	100	0.02	KLD	81	7	14	11
			MMD	96	3	6	0
			CM-DD	93	3	19	3
			FCM-DD	99	0	13	0
$C(0.10)$	500	0.01	KLD	31	19	2	49
			MMD	50	11	2	38
			CM-DD	41	12	2	46
			FCM-DD	59	11	5	29
$C(0.10)$	100	0.02	KLD	34	29	3	36
			MMD	64	8	7	27
			CM-DD	52	17	8	30
			FCM-DD	73	12	8	14

Experiment 7. Higher Dimensions. We also compared FCM-DD and CM-DD on higher dimensions of data stream to test the scalability. The data streams that we used were $4D\_C0.20$, $6D\_C0.20$, and $10D\_C0.20$. The parameter d_ε used in fuzzy membership function is not as linear as d_ε in constructing a competence model. Thus, we set d_ε by preliminary computational testing, and the specific values are listed in Table 7. As we expected, it became harder to detect real drift when more stationary dimensions were added. The performance of MMD decreased considerably when handling higher data dimensions with smaller changes. Even though KLD and CM-DD showed better performance than MMD in higher data dimensions, FCM-DD still outperformed the other methods with more *True* counts and fewer *Miss* counts. This provides further proof that our detection method is more sensitive to smaller changes.

Table 7: Drift detection results on Δ-dimensional data streams.

Data stream	d_ε	Method	True	Late	False	Miss
$4D\_C(0.20)$	n/a	KLD	84	8	6	7
	n/a	MMD	93	4	3	2
	0.150	CM-DD	91	4	8	4
	0.120	FCM-DD	98	1	6	0
$6D\_C(0.20)$	n/a	KLD	81	6	5	12
	n/a	MMD	79	6	6	14
	0.300	CM-DD	94	3	8	2
	0.135	FCM-DD	96	1	7	1
$10D\_C(0.20)$	n/a	KLD	74	12	8	13
	n/a	MMD	50	12	3	37
	0.500	CM-DD	82	7	4	10
	0.160	FCM-DD	85	8	4	6

The results of Experiments 4, 5, and 7 are summarized below. The average counts of these experimental results are shown in Table 8. FCM-DD was much more powerful than other drift detection methods in various situations since it had a higher *True* detect rate and the same level of *False* detect rate. In this series of experiments FCM-DD outperformed CM-DD in three areas: 1) detection rate, 2) sensitivity to smaller degrees of drift and a relaxation of the statistical guarantee; and 3) robustness when used with small sample sizes.

Table 8: Average drift detection results

Method	True	Late	False	Miss
KLD	75.08	7.77	7.92	16.15
MMD	80.08	6.15	6.69	12.77
CM-DD	81.62	6.85	6.85	10.54
FCM-DD	87.77	4.23	7.35	6.92

Experiment 8. Efficiency. This experiment assessed the computational effort of all four drift detection methods over different data dimensions and window sizes. The nature of data distribution-based drift detection methods comprises three modules: data modeling, divergence measurement, and the statistical significance test. Since all four drift detection methods employ similarity permutation/bootstrap test, the computation time of one round of drift detection was compared with a permutation/bootstrap test repeated $K = 20$ times. All four drift detection methods were implemented by MATLAB, and the programs were run on a PC with a 2.2GHz Intel Core i7 processor and 16GB of memory.

As shown in Figure 6 and Table 9, the computation time of all methods were slightly affected by the number of data dimensions d, and were directly affected by the window size (n). The computation time of MMD, CM-DD and FCM-DD exhibited quadratic growth when the window size increased. In fact, CM-DD and FCM-DD need to maintain the relationship between any two data points for modeling; thus, the computational complexities of CM-DD and FCM-DD are $O(n^2)$. Also, the computational complexity of MMD is $O(n^2)$ [35]. Since the computation complexity of KLD has been proven to be $O(nd \log \frac{1}{\delta})$ [25], the computation time of KLD exhibits linear growth when the window size increases.

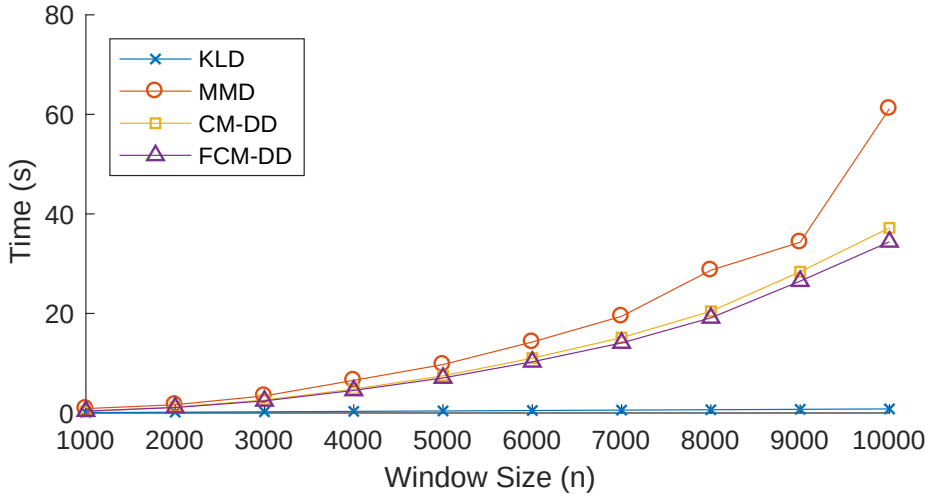

Figure 6: Computation time of 2D data with different window size

Table 9: Computation time (in second) with different data dimensions and window size

Method	Dimensions	Window Size (n)				
		2000	4000	6000	8000	10000
KLD	2	0.1628	0.3253	0.4963	0.6500	0.8146
	4	0.1604	0.3237	0.4949	0.6533	0.8521
	10	0.1787	0.3426	0.5209	0.7082	0.8605
MMD	2	1.6577	6.5533	14.2638	28.6822	61.1449
	4	1.6700	6.6027	14.3088	28.6444	61.0773
	10	1.6627	6.5741	14.2506	28.8001	60.9580
CM-DD	2	1.1819	4.8053	11.0129	20.4323	37.1936
	4	1.2114	4.9838	11.4777	21.2458	38.2489
	10	1.2083	4.9793	11.4711	21.1521	38.4966
FCM-DD	2	1.0967	4.5338	10.2883	19.1183	34.3934
	4	1.0885	4.5133	10.2956	19.1238	34.3761
	10	1.0963	4.5553	10.3733	19.2427	34.5998

5. Conclusions and further studies

In this paper, we presented and evaluated a drift detection-based model, called FCM-DD. The model was designed to help DSSs recognize different types of concept drift in non-stationary streaming data. FCM-DD requires no prior knowledge of the underlying data and can output when, how, and where concept drift occurs. How concept drift occurs is returned as fuzzy competence-based empirical distance and where it occurs is reflected in the partitions generated by the fuzzy competence model with great differences. Our model provides a better estimation of the empirical distribution of data, the use of fuzzy sets theory makes our drift detection method more sensitive and reliable. Empirical experiments demonstrate three advantages of our proposed FCM-DD over state-of-the-art competence model drift detection [17]: 1) a higher detection rate; 2) more sensitive to smaller degrees of drift and a relaxation of the statistical guarantee; and 3) robustness on small sample sizes.

We find that estimating empirical data distribution using fuzzy competence models is very effective, but efficiency is reduced. Our next attempt will aim to provide a high-performance drift detection method with less computation cost by integrating tree data modeling techniques. We also should reveal the relations between the parameters used to build fuzzy competence model and the data dimensions. Finally, this paper is part of a greater body of work that is responding to concept drift problems in DSSs. Effective reaction strategies for adaptive DSSs are needed to improve final decision outcomes and accuracy.

Acknowledgments

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

Reference

[1] G. Widmer, M. Kubat, Learning in the presence of concept drift and hidden contexts, Machine learning 23 (1) (1996) 69–101.

[2] R. Elwell, R. Polikar, Incremental learning of concept drift in nonstationary environments, IEEE Transactions on Neural Networks 22 (10) (2011) 1517–1531.

[3] S. Piramuthu, M. J. Shaw, Learning-enhanced adaptive dss: a design science perspective, Information Technology and Management 10 (1) (2009) 41–54.

[4] J. B. Gomes, E. Menasalvas, P. A. Sousa, Learning recurring concepts from data streams with a context-aware ensemble, in: Proceedings of the 2011 ACM symposium on applied computing, ACM, 2011, pp. 994–999.

[5] J. Gama, I. Žliobaitė, A. Bifet, M. Pechenizkiy, A. Bouchachia, A survey on concept drift adaptation, ACM Computing Surveys (CSUR) 46 (4) (2014) 44.

[6] I. Žliobaitė, Adaptive training set formation, Ph.D. thesis, Vilnius University (2010).

[7] C. Alippi, G. Boracchi, M. Roveri, Just-in-time classifiers for recurrent concepts, IEEE transactions on neural networks and learning systems 24 (4) (2013) 620–634.

[8] L. L. Minku, A. P. White, X. Yao, The impact of diversity on online ensemble learning in the presence of concept drift, IEEE Transactions on Knowledge and Data Engineering 22 (5) (2010) 730–742.

[9] B. Krawczyk, M. Woźniak, One-class classifiers with incremental learning and forgetting for data streams with concept drift, Soft Computing 19 (12) (2015) 3387–3400.

[10] B. Mirza, Z. Lin, N. Liu, Ensemble of subset online sequential extreme learning machine for class imbalance and concept drift, Neurocomputing 149 (2015) 316–329.

[11] H. Yang, S. Fong, Countering the concept-drift problems in big data by an incrementally optimized stream mining model, Journal of Systems and Software 102 (2015) 158–166.

[12] A. Liu, G. Zhang, J. Lu, A novel weighting method for online ensemble learning with the presence of concept drift, in: Proceedings of the 11th Internationaal FLINS Conference, Decision Making and Soft Computing, World Scientific Publishing Co. Pte. Ltd., 2014.

[13] R. Klinkenberg, Learning drifting concepts: Example selection vs. example weighting, Intelligent Data Analysis 8 (3) (2004) 281–300.

[14] I. Žliobaitė, Combining similarity in time and space for training set formation under concept drift, Intelligent Data Analysis 15 (4) (2011) 589–611.

[15] N. Lu, J. Lu, G. Zhang, R. L. de Mantaras, A concept drift-tolerant case-base editing technique, Artificial Intelligence 230 (2016) 108–133.

[16] J. Shao, Z. Ahmadi, S. Kramer, Prototype-based learning on concept-drifting data streams, in: Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining, ACM, 2014, pp. 412–421.

[17] N. Lu, G. Zhang, J. Lu, Concept drift detection via competence models, Artificial intelligence 209 (2014) 11–28.

[18] J. Gama, P. Medas, G. Castillo, P. Rodrigues, Learning with drift detection, in: Brazilian Symposium on Artificial Intelligence, Springer, 2004, pp. 286–295.

[19] M. Baena-Garcıa, J. del Campo-Ávila, R. Fidalgo, A. Bifet, R. Gavalda, R. Morales-Bueno, Early drift detection method, in: Fourth international workshop on knowledge discovery from data streams, Vol. 6, 2006, pp. 77–86.

[20] K. Nishida, K. Yamauchi, Detecting concept drift using statistical testing, in: International conference on discovery science, Springer, 2007, pp. 264–269.

[21] G. J. Ross, N. M. Adams, D. K. Tasoulis, D. J. Hand, Exponentially weighted moving average charts for detecting concept drift, Pattern Recognition Letters 33 (2) (2012) 191–198.

[22] B. Su, Y.-D. Shen, W. Xu, Modeling concept drift from the perspective of classifiers, in: Cybernetics and Intelligent Systems, 2008 IEEE Conference on, IEEE, 2008, pp. 1055–1060.

[23] D. Kifer, S. Ben-David, J. Gehrke, Detecting change in data streams, in: Proceedings of the Thirtieth international conference on Very large data bases-Volume 30, VLDB Endowment, 2004, pp. 180–191.

[24] J. Gama, R. Fernandes, R. Rocha, Decision trees for mining data streams, Intelligent Data Analysis 10 (1) (2006) 23–45.

[25] T. Dasu, S. Krishnan, S. Venkatasubramanian, K. Yi, An information-theoretic approach to detecting changes in multi-dimensional data streams, in: In Proc. Symp. on the Interface of Statistics, Computing Science, and Applications, Citeseer, 2006.

[26] R. Sebastião, J. Gama, Change detection in learning histograms from data streams, in: Portuguese Conference on Artificial Intelligence, Springer, 2007, pp. 112–123.

[27] P. Vorburger, A. Bernstein, Entropy-based concept shift detection, in: Data Mining, 2006. ICDM'06. Sixth International Conference on, IEEE, 2006, pp. 1113–1118.

[28] B. Efron, R. J. Tibshirani, An introduction to the bootstrap, CRC press, 1994.

[29] S. Massie, S. Craw, N. Wiratunga, What is cbr competence, BCS-SGAI Expert Update 8 (1) (2005) 7–10.

[30] B. Smyth, M. Keane, Remembering to forget: a competence preserving deletion policy for case-based reasoning system, in: Proc. 14th Int. Joint Conf. Artificial Intelligence (Morgan-Kaufmann, 1995), pp. 377–382.

[31] B. Smyth, E. McKenna, Modelling the competence of case-bases, in: European Workshop on Advances in Case-Based Reasoning, Springer, 1998, pp. 208–220.

[32] L. A. Zadeh, Fuzzy sets, Information and control 8 (3) (1965) 338–353.

[33] F. Liese, I. Vajda, On divergences and informations in statistics and information theory, IEEE Transactions on Information Theory 52 (10) (2006) 4394–4412.

[34] E. Koehler, E. Brown, S. J.-P. Haneuse, On the assessment of monte carlo error in simulation-based statistical analyses, The American Statistician 63 (2) (2009) 155–162.

[35] A. Gretton, K. M. Borgwardt, M. J. Rasch, B. Schölkopf, A. Smola, A kernel two-sample test, Journal of Machine Learning Research 13 (Mar) (2012) 723–773.