Quantifying the Limited and Gradual Concept Drift Assumption

Joseph Sarnelle, Anthony Sanchez, Robert Capo, Joshua Haas, and Robi Polikar

***Abstract*—Nonstationary environments, where underlying distributions change over time, are becoming increasingly common in real-world applications. A specific example of such an environment is concept drift, where the joint probability distributions of observed data drift over time. Such environments call for a model that can update its parameters to adapt to the changing environment. An extreme case of this scenario, referred to as extreme verification latency, is where labeled data are only available at initialization, with unlabeled data becoming available in a streaming fashion thereafter. In such a scenario, the classifier must update its hypothesis based on only unlabeled data drawn from the drifting distributions. In our prior work, we described a framework, called COMPOSE, that works well in this type of environment, provided that the data distributions experience limited (or gradual) drift. Limited drift assumption is common in many concept drift algorithms yet — surprisingly — there is little or no formal definition of this assumption. In this contribution, we describe a mechanism to formally quantify limited drift. We define two metrics, one that represents the normalized class separation drift, and the other that uses the ratio of between-class separations and within class drift through time. We test these metrics on both synthetic and real world problems, and argue that the latter can be more suitably used.**

I. INTRODUCTION

Concept drift is a challenging classification problem for nonstationary data, for which the joint probability distribution of observations and their corresponding grouping variables (i.e., labels) change with time. In the most general sense, concept drift can be described as

$$p_t(X, L) \neq p_{t-1}(X, L) \tag{1}$$

where X is an N x d observation matrix with N observations in d dimensions, L is a length N vector containing labels corresponding to each observation in X, and $p_t(X, L)$ is the joint probability distribution of the data and their labels. The problem becomes even more difficult if L is completely unknown at all times beyond initialization $t > t_0$. This type of problem presents itself in many situations, where some data are manually annotated at a specific point in time t_0, and only unlabeled data are available thereafter. This scenario is known as an initially labeled nonstationary environment (ILNSE) [1], [2]. The relevance of the labels received at t_0 degrades as the data distributions continue to drift away from their initial positions. Our previous efforts [1], [2] have explored various methods for classification in these environments and defined a framework, called COMPOSE, for doing so.

The COMPacted Object Sample Extraction (COMPOSE) framework is a wrapper approach for learning in an ILNSE. The framework makes minimal assumptions about the nature of the drift aside from the constraint that it is limited in nature (i.e. abrupt changes do not occur). Limited drift is an intuitive constraint for an algorithm designed to operate in an ILNSE. While there are several algorithms on concept drift that make the gradual drift assumption [3]–[6], a quantification or a formal definition of what gradual (or abrupt, for that matter) drift has not yet been defined. Our goal is to quantify drift in a way that allows us to determine its impact on COMPOSE's classification performance in particular, and on concept drift algorithms in general.

COMPOSE is a three-step approach that runs every time a new batch of unlabeled data is received. The first step employs a semi-supervised learning (SSL) algorithm using currently available labeled data to classify the new unlabeled data. Currently available labeled data are the initial data at timestep $t = t_0$, or the labels generated by COMPOSE using core supports (described below) on all future timesteps, $t > t_0$. Many SSL algorithms have been developed (see [7] for examples), but we chose the label propagation algorithm [8] for our experiments. The second step involves class-specific density estimation followed by compaction to determine those central regions of the distribution that are most heavily represented in the current data. In COMPOSE, these areas are referred to as core support regions. Any of the density estimation procedures can be used for this step, such as α-shapes, Gaussian mixture models, k-nearest neighbor based estimation, etc. Each approach has its own strengths and weaknesses, as compared in [9]. In this work, we use Gaussian mixture models due to their substantially reduced computational complexity over other methods. The third step is core support extraction (CSE), which takes a sample of instances from the core support region of each class. These sampled instances were originally unlabeled, but have just been labeled by the semi-supervised learning algorithm. These instances, called core supports, then serve as the "current" labeled data to be used in the next time step to help the SSL algorithm label the new set of unlabeled data. The algorithm then iteratively continues to label unlabeled data, determine core support region as the most dense region of the current distribution, and extract samples from this region as core supports to be used as labeled data for the next iteration. A block diagram of COMPOSE that illustrates the approach is shown in Figure 1. The algorithmic and implementation details can be found in [1]. Steps shown in blue are the three main

Joseph Sarnelle, Anthony Sanchez, Robert Capo, Joshua Haas, and Robi Polikar are with the Department of Electrical & Computer Engineering at Rowan University, Glassboro, NJ, USA. (emails: sarnel62@students.rowan.edu, sanche08@students.rowan.edu, robcapo@gmail.com, haasj74@students.rowan.edu, polikar@rowan.edu)

This work is supported by the NSF under Grant No: ECCS-1310496.

978-1-4799-1959-8/15/$31.00

components of COMPOSE while the steps shown in red are data processing tasks.

If the environment can provide labeled data - perhaps at least occasionally - such information can also be used by COMPOSE, where the new labeled data are used as the current core supports. An active learning step can optionally be added as well [10]. Doing so trades the constraint of limited drift for the ability to request the labels of a small number of carefully selected instances at any given time. In this work, we assume that no future labeled data are available, and that all subsequent data are unlabeled.

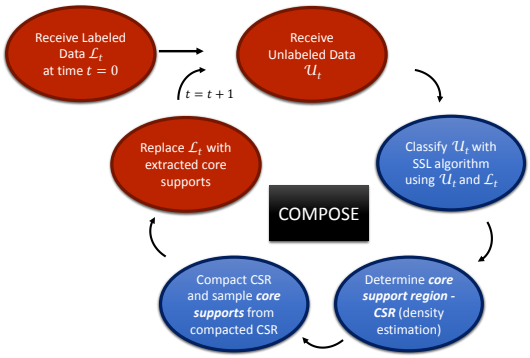

Fig. 1. Flowchart of the COMPOSE algorithm

COMPOSE requires limited (gradual) drift so that it can track the unlabeled data by following the core support regions. Limited, however, drift is a general assumption with a heuristic description that lacks a formal mathematical definition. How slow the drift needs to be to qualify as "limited" or "gradual"? An obvious approach to determine the drift rate from one timestep to the next is to measure distances or divergences between distributions. We explored three metrics, the Kullback-Leibler (KL) divergence [11], the Hellinger distance [12], and the average pairwise distance between observations to serve as a basis in determining a suitable metric for the drift rate.

Determining and quantifying the drift rate is not just to satisfy a theoretical interest, but it also has some practical reasons as well. Depending on the SSL algorithm and CSE method chosen, COMPOSE can be computationally expensive, particularly in high dimensional feature spaces. Due to the computational time and resources needed to run the algorithm, it may be prudent to determine ahead of time whether the limited drift assumption of the algorithm is satisfied, and hence the algorithm is suitable for the data at hand to begin with. The data must not violate the limited drift assumption if the algorithm is to perform well. We show in this work that under certain cases, it is possible to define a metric and determine whether the data satisfy the limited drift assumption with the help of this metric.

Section II explains the theory behind the three metrics we use to measure the distance between distributions. Section III explores methods for using these distance metrics to determine the drift between two timesteps. Section IV presents the experimental setups for our metrics and discusses the results from these experiments. Section V explains the conclusions and avenues for future work based on our results.

II. Divergence & Distance Measures

Without loss of any generality, we assume a two class problem, where X_t is the set of observations belonging to class 1 at time t and Y_t is the set of observations belonging to class 2 at time t. Our discussion below naturally extends to data with multiple classes. Any references to the distributions X and Y at time $t-1$ are the corresponding core supports for classes X and Y, respectively. We define $D(X, Y)$ as the distance (or divergence) between distributions X and Y, regardless of the specific distance or divergence metric used.

It is important for distance metrics to be independent of distribution volume. An increase or decrease in cardinality is not indicative of an increase or decrease in distance between the underlying class distributions.

A. Kullback-Leibler Divergence

Kullback-Leibler (KL) divergence, also known as relative entropy, is a measure of the information loss between two probability distributions. The KL divergence of distribution Y from X is written as $\mathrm{KL}(X \parallel Y)$ and is a measure of the information lost when Y is used to approximate X. Although KL divergence is not a true distance metric (due to $\mathrm{KL}(X \parallel Y) \neq \mathrm{KL}(Y \parallel X)$ i.e. it is not symmetric), it is one of the most commonly used measures to determine how "far" two distributions are from each other. The general equation to calculate the KL divergence between two probability distributions X and Y is given as

$$\mathrm{KL}(X \parallel Y) = \sum_{x \in (X \cup Y)} \ln\left(\frac{P(X(x))}{P(Y(x))}\right) P(X(x)) \quad (2)$$

Here $P(X(x))$ and $P(Y(x))$ represent the probability density functions (PDFs) of the distributions X and Y, respectively. When X and Y are well separated, the ratio of $P(X(x))$ and $P(Y(y))$ is a large positive number for values of x that are in the core support of $P(X(x))$, contributing large positive numbers to the sum in Equation (2), whereas the values of x that are in the core support region of P(Y(x)) contribute small negative numbers, resulting in a positive large number for the divergence between X and Y.

Conversely, when the distributions are not well separated, $P(Y(x))$ and $P(Y(x))$ have similar values for the same x. In the extreme case, where the distributions are equivalent, the natural logarithm evaluates to $ln(1) = 0$ for all x, making the sum zero. Therefore, the KL divergence is an appropriate divergence metric to measure the distance between two distributions.

B. Hellinger Distance

Hellinger distance is a quantification of the similarity between two probability distributions. Unlike the KL divergence, the Hellinger distance is symmetric, and therefore is a proper

distance metric. The Hellinger distance between two distributions, X and Y, has a general form given by Equation 3 below.

$$\mathrm{H}(X,Y) = \frac{1}{\sqrt{2}}\sqrt{\sum_{x\in X\cup Y}\left(\sqrt{P(X(x))} - \sqrt{P(Y(x))}\right)^2} \quad (3)$$

When X and Y are completely separated, the Hellinger distance is close to 1. When X and Y are completely overlapping, the distance evaluates to a small number greater than 0, and if they are identical, the Hellinger distance is 0.

C. Average Pairwise Distance

A commonly — if perhaps rather naively — used non-statistical distance metric to measure the distance between two distributions is the average pairwise distance between all observations in both distributions. This metric simply iterates through each observation of both distributions, computes the distances, and takes the average across all observations. The average pairwise distance is given by

$$D_{avg}(X,Y) = \frac{1}{|X||Y|}\sum_{x\in X}\sum_{y\in Y}(x-y)^2 \quad (4)$$

We use Euclidean pairwise distances, but any function that measures the distance between two points can be used. The average pairwise distance is a simple and effective way to measure the distance between two distributions when the probability density functions (PDFs) generating their data are unknown. This distance metric relies only on the closeness assumption (that instances near each other in the feature space are related), which is the main assumption of the label propagation algorithm as well.

Figure 2 shows a comparison of the three basic metrics discussed above. These metrics were taken between two distributions that initially started on top of each other and gradually drifted apart over time. We can see that each metric has a different characteristic. As the distance between cluster centers increases, the KL divergence increases exponentially, the Hellinger distances increase logarithmically, and the pairwise distance increases linearly. Due to its linearity and ease of computation, we employ the average pairwise distance in the experiments described below.

III. Calculating Drift

The ability to measure distance between two distributions is obviously important when calculating drift. Two types of distances are relevant: between class separation, and within class drift. The between class separation is the distance between distributions X and Y at any timestep. The within class drift is the distance that a class drifts between time $t-1$ to time t. We investigate two new metrics for quantifying the amount of drift in an ILNSE.

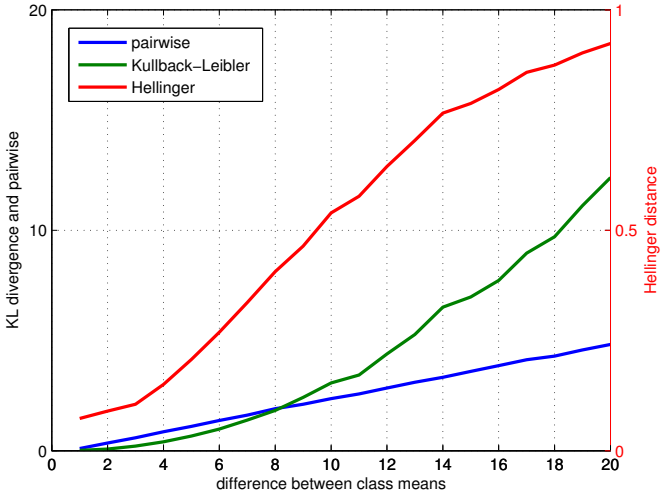

Fig. 2. Comparison of the metrics. The red axis (on the right) represents the Hellinger distance, the black axis represents the value of the KL divergence and pairwise distance .

A. Normalized Class Separation (NCS) Based Drift: $\Delta(t)$

We define $\Delta(t)$ as the drift that has occurred between two consecutive timesteps, $t-1$ and t. Rather than considering only within class drift, we use between class separation as a normalization factor, as distributions can be tracked even when they drift faster if they are well separated. Hence, we define the normalized class separation based drift as

$$\Delta(t) = \frac{\max\left(D(X_t, X_{t-1}), D(Y_t, Y_{t-1})\right)}{D(X_{t-1}, Y_{t-1})} \quad (5)$$

where D is a method for calculating distance or divergence such as the average pairwise distance or KL divergence. We note that class labels must be known for all data at timesteps $t-1$ and t a priori. In a controlled synthetic data experiment or in cases where there are labeled data available, this information is easily obtained, and the amount of drift in an environment can then be used to benchmark performance of different algorithms. In the practical case, labeled data are not always available in large quantities, but there may exist an initial, manually annotated dataset that represents a larger unannotated dataset. Drift calculations on the initial dataset can be used to infer whether the drift rate is gradual enough for the chosen concept drift algorithm, provided that the drift profile does not change dramatically in the future.

It is important to note that $\Delta(t)$ only measures the drift within an environment, with no consideration given to the direction of the drift. The direction can impact the difficulty of the problem. For example, when distributions are drifting away from each other, the classification problem is much easier to solve than when they are drifting towards each other, even if the distance the distributions have drifted is large.

B. Drift Classification Risk

As stated previously, the direction of the drift is not taken into consideration in $\Delta(t)$. A scenario where distributions are

drifting away from each other is easily tracked by COMPOSE, regardless of how much each distribution drifts. Conversely, using $\Delta(t)$ to model the drift in this scenario results in a value indicating - incorrectly - that COMPOSE will perform poorly. In order to properly quantify the drift in more challenging environments, we define a ratio that takes into account the divergence of the unlabeled data with its own corresponding core supports, as well as the nearest core supports from any other class. When there is little drift within a class between timesteps, the distance (or divergence) is small and the ratio is also small, a scenario that can easily be handled by COMPOSE. As the unlabeled data drifts away from its corresponding core support towards the core support of another class, the divergence between the unlabeled data and both core supports approach the same value. In this scenario, the ratio yields a value of 1, indicating an uncertainty of around 50% for unlabeled data between core supports from two classes. A ratio much greater than 1 represents a scenario that is unfavorable for COMPOSE, where the unlabeled data are misclassified, because the new unlabeled data are closer to the core supports of a different class than the class they belong. We call this ratio the *drift classification risk* (DCR), a measure of drift rate that is also suitable for multiple classes, and is defined as

$$DCR = \max_j \left(\frac{D(d_{j,t+1}, d_{j,t})}{\min_{i \neq j} D(d_{j,t+1}, d_{i,t})} \right) \tag{6}$$

This measure calculates the ratio of the distance from any unlabeled data of a given class to its own core supports and the same unlabeled data to the nearest core supports of any other (nearest) class. Here $D(d_{j,t+1}, d_{j,t})$ corresponds to the distance between the unlabeled data distribution of class j at time $t+1$ and the core supports from class j at the previous time t. The distance (average pairwise, Hellinger, etc.) or divergence (KL), between the unlabeled data distribution of class j at time $t+1$ and the core supports from class i at the previous time t is referred to as $D(d_{j,t+1}, d_{i,t})$. Unlike COMPOSE, which requires labeled only at initialization, the DCR requires class labels for all data at each timestep it is calculated at, making it difficult to use in a scenario where no labeled data is available beyond the first timestep; however, it is common for a dataset to have labeled data for several timesteps, or have a larger set of initial data that has the drift already present. Such data can be used to calculate the DCR and provide a quantified feedback to the user as to whether the scenario qualifies as limited drift.

IV. Experimental Setup & Results

To test the utility of both drift rate quantification metrics, we implemented two categories of experimentation. The first round of testing includes experiments using synthetic datasets, which serve as a proof of concept that these metrics are indeed capable of appropriately quantifying drift. The second round of testing extends the usefulness of these metrics from the synthetic domain to real world datasets. Results on real world datasets prove the practical utility of these metrics for pre-classification assessment on the labeled data provided.

A. Synthetic Data

To test the $\Delta(t)$ drift metric, we started with a two class, unimodal Gaussian case. We initialize two distributions at time $t-1$, horizontally separated by h units of distance and vertically separated by v units of distance, as shown in Figure 3. We then generate two new distributions at time t that have drifted by d, a user controlled parameter. In order to determine the exact conditions under which COMPOSE fails, we extract core supports from the distributions at time $t-1$, and use them to classify the distributions at time t, while gradually increasing the drift distance d from 0 to h.

Obviously, there are many possibilities for the direction of drift, and as previously mentioned, drift direction has a significant impact on the difficulty of the problem. In the most challenging scenario, the distributions drift directly toward each other. In the easiest scenario, the distributions drift directly away from each other. In our first experiment, we adjust the values of v and h to change the amount of separation, and force each distribution to drift horizontally toward each other. The goal of the experiment is to determine the value at which COMPOSEs performance drops below 50%, the minimum performance one can expect from a classifier in a binary class problem.

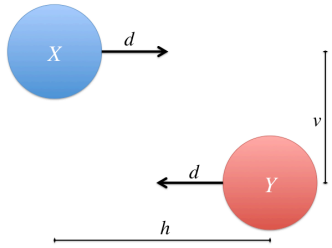

Fig. 3. Experiment 1 setup

The results of the first experiment are shown in Figure 4, where the blue and red curve is the value of $\Delta(t)$ and COMPOSE's performance, respectively. This experiment was run over 10 iterations with different realizations of the same distributions in order to show the 95% confidence intervals, indicated as shaded regions in Figure 4. The most challenging situation (a) is shown where v = 0, because the distributions are drifting directly toward each other. We can see that the performance drops earliest in this case, when $\Delta(t)$ = 0.5 (indicated by the vertical line). We marked the point at which $\Delta(t)$ = 0.5 in the rest of the plots for easy visual comparison. Figure 4 (b) and (c) show that if h is kept constant, increasing v makes the problem easier, as expected. The performance remains higher, and $\Delta(t)$ stays under 0.5 for larger values of d. Figure 4 (d) and (e) show that $\Delta(t)$ works if the angle of approach remains constant. When the ratio of $\frac{v}{h}$ remains constant, the performance drop occurs at the same value of d. $\Delta(t)$ also remains constant, indicating that it is robust to changes in the spread of data. Contrarily, there exists a case, discussed in the following experiment, where exactly the same

amount of drift occurs, but in the opposite direction, which yields the same $\Delta(t)$ profile, despite no negative impact on the performance of COMPOSE. Therefore, $\Delta(t)$ serves as a metric to quantify how much any one class has drifted in an environment, but it is not reliable for determining the difficulty a classifier will experience during classification.

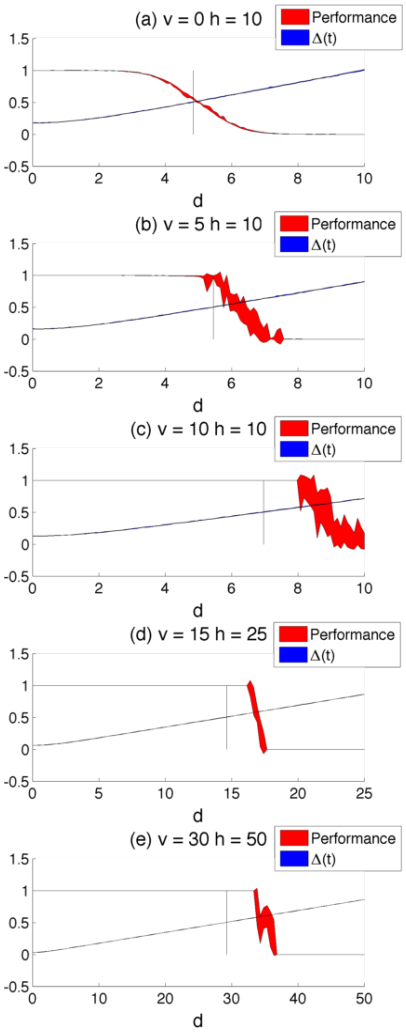

Fig. 4. Experiment 1 results, the left axis represents the $\Delta(t)$ value and COMPOSE's performance (performance normalized between 0 and 1), both are shown with a 95% confidence interval. d represents the distance between class means.

To determine if the DCR is a useful metric for quantifying drift in an ILNSE, we explore a special case of the two class, unimodal Gaussian case shown in Figure 3. This experiment lasts for only two timesteps, as the goal is to simulate varying severity in the abruptness of drift between receiving new batches of data. Two distributions are initialized at a set horizontal distance apart that does not change throughout the experiment, and with no vertical separation (i.e., v=0) between the y-coordinates of their means. Core supports are extracted from this data and serve as labeled data to be used by the SSL algorithm during classification. At the next timestep, unlabeled data are generated for each class. The unlabeled data for one class remains relatively close to its core support, and are easily classified. We are interested in the unlabeled data for the other class as it drifts at incremental distances away from its own core support and towards the core support of the other class. We calculate the DCR of this unlabeled data as well as the performance of COMPOSE on this class to establish a relationship between the two. This process constitutes one trial, and is repeated with increasing levels of drift between the unlabeled data of one class and its corresponding core support. The entire experiment is repeated 10 times to establish a 95% confidence interval. A visual representation of this setup is shown in Figure 5.

Fig. 5. Experiment 2 setup

The results from this second experiment are shown in Figure 6. This figure represents the performance of COMPOSE when the level of drift changes as indicated by the DCR values. It can be seen that COMPOSE experiences no difficulties in classifying the unlabeled data when the DCR is significantly less than 1, but as the value approaches 1, the performance begins to decrease and as the DCR continues to increase beyond 1, COMPOSE's performance on the unlabeled data decreases until it reaches 0% accuracy, or a total loss of support for class $X(t)$, represented by red in Figure 6.

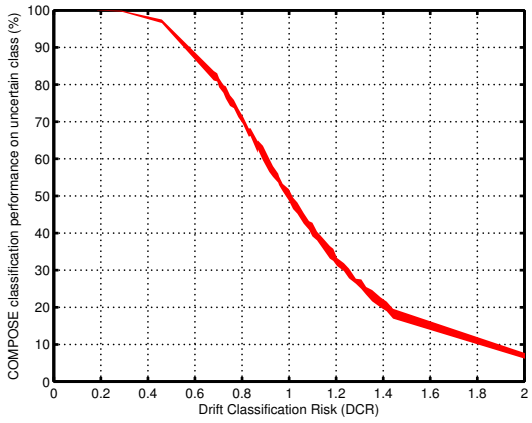

Fig. 6. Experiment 2 results calculating DCR when unlabeled data drifts in increasing magnitudes from its corresponding core supports to that of another class. The x-axis shows these DCR values and the corresponding performance of COMPOSE (shown on the y-axis) on the unlabeled data. The width of the red curve indicates the 95% confidence interval over 10 trials.

To demonstrate the inability of $\Delta(t)$ to reliably determine when a classifier will perform poorly, we recreate the same experiment from Figure 5 and calculate $\Delta(t)$ and the DCR. We then repeat the experiment with the direction of the drift rotated 180 degrees, so that the classes are no longer drifting toward one another. In Figure 7, $\Delta(t)$ is plotted on the right axis and the DCR is plotted on the left axis for easy

comparison. We observe that the value of $\Delta(t)$ remains the same when the magnitude of the drift is constant regardless of the direction (i.e., a class is drifting towards or away from another). The DCR value approaches 1 only when a class is drifting towards the core support of another class, but remains far below 1 when drifting away from it.

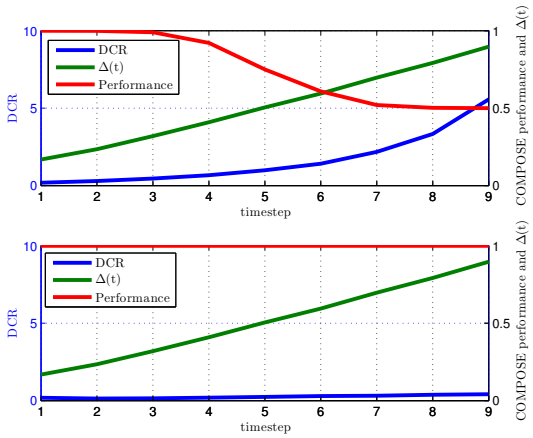

Fig. 7. Experiment 2 results comparing $\Delta(t)$ and DCR. The blue axis represents the DCR values versus the timesteps represented on the horizontal axis. The black axis (on the right) represents the $\Delta(t)$ value and COMPOSE's performance (performance normalized between 0 and 1). The top plot shows COMPOSE's performance, DCR, and $\Delta(t)$ values as unlababled data drifts towards another class. The bottom plot shows COMPOSE's performance, DCR, and $\Delta(t)$ values as unlababled data drifts away from another class.

DCR is defined in such a way that it assumes the value of 1 when the unlabeled data is just as close to its own core support as the core support of another class, representing a scenario with the most uncertainty as to which group of instances belong to which class. This situations also corresponds to the case where the label propagation algorithm used by COMPOSE is likely to encounter the most difficulty in correctly determining the classification of the unlabeled data. To test the robustness of the characteristics of the DCR, such as recovery from a drift with a DCR value of 1, we create an experiment shown in Figure 8 where we reverse the direction of the drift at this DCR value to determine if any overlapping core supports left behind in the previous timestep allow COMPOSE to recover from near failure.

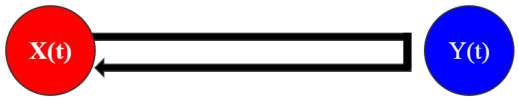

Fig. 8. Experiment 3 setup

The results of this experiment are shown in Table I, which demonstrate that COMPOSE can indeed recover from abrupt drift quantified by a DCR close to 1 as long as the data distribution does not sustain this level of drift for an extended period of time. This is due to the label propagation process. When unlabeled data lie nearly equidistantly between two core supports, labels have similar probabilities of propagating to either class of the unlabeled data. The result is a division of the unlabeled data between both classes. Figure 9 shows this scenario.

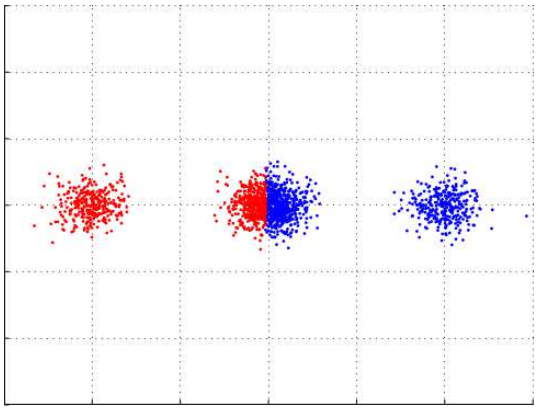

Fig. 9. Split unlabeled data at DCR=1

Results in Table I show a degrading classification performance as the DCR values approaches 1. The classification performance is 56.85% at a peak DCR value of 1.19. Upon the direction reversal, classification performance immediately begins to increase as the unlabeled is now drifting away from the core support of the other class, with COMPOSE eventually recovering completely. DCR values much greater than 1 have proven to be irrecoverable in other tests and would require our active learning component of COMPOSE [10] to be initiated for success.

TABLE I
EXPERIMENT 3 RESULTS

DCR Value	Performance (%)
0.20	100
0.33	98.65
0.60	87.05
0.85	74.25
1.19	56.85
0.86	75.70
0.59	86.56
0.44	94.30
0.36	98.05
0.22	100

B. Real World Data

We have also tested COMPOSE and the DCR metric using the electricity dataset obtained from pricing information in the Australian New South Wales (NSW) Electricity Market and its neighboring state Victoria [13]. Seven features (week, period, NSW price, NSW demand, Victoria price, Victoria demand, transfer from NSW to Victoria) are used to determine whether the price of electricity in NSW will be higher than the previous 24 hour average. The original dataset contains 45,312 instances recorded every 30 minutes, with many instances missing data. Tests were performed with the 27,549 instances that are not missing any data, in 11,445 of which the price went up and

in 16,104 of which the price went down, indicating a class imbalance around 59%. Data were grouped into 55 batches of 500 instances each (corresponding to about 10 days per batch) and remaining data that did not fit in the batches were dropped to insure a consistent test-bed across all timesteps. All features were whitened before classification.

It turns out, COMPOSE does not perform as well as we had hoped on this dataset. However, running the algorithm to find out about this is an inefficient way to do so. Having the DCR metric allows us to realize why the algorithm was not performing well: the dataset does not meet the "gradual drift" assumption. Computing the DCR provides us with an alternative to test the algorithm ahead of time — a prescreening if you will — without actually running the algorithm. In this particular scenario, COMPOSE takes 12.53 seconds to run while the DCR calculations only take 0.49 seconds to compute. As dimensionality of the data increases, the computational complexity of COMPOSE also increases, but the complexity for the DCR calculations remain unchanged. The time and resources saved by knowing whether the algorithm is to be run on an unfavorable environment is a desirable advantage. Results from this experiment are shown in Figure 10, which prove that the DCR metric is an appropriate metric for determining the drift rate, and in turn whether the given environment can qualify as limited or gradual drift. The large green spike in the DCR around timestep 20 indicates a very abrupt drift in the data. Before this drift, COMPOSE maintains an average classification performance of around 65%, after initially being given only 20% of the data at the first timestep and no labeled data thereafter. After the drift, average classification performance drops to 45%. With no new labeled information, COMPOSE is unable to recover after this drift.

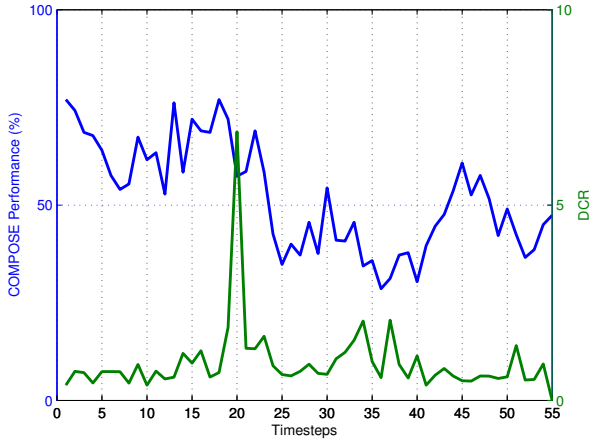

Fig. 10. Real world data analysis. The blue axis represents COMPOSE's performance, the green axis (on the right) represents the DCR value.

V. Conclusions and Future Work

The ability to quantify drift is extremely important to concept drift algorithms, such as COMPOSE. It can be used to determine whether the algorithm is suitable for the dataset. It can also be used to generate parameters that adjust to the drift rate. Our first experiment evaluates the normalized class-separation based drift metric, $\Delta(t)$, and shows that as long as $\Delta(t) \leq 0.5$, COMPOSE should perform well in an environment where classes drift directly towards each other. As previously stated, however, this metric only takes the magnitude of the drift, but not the direction of the drift into consideration. This is a crucial component for concept drift algorithms, as the direction of the drift can impact performance.

To remedy this concern, the drift classification risk was developed. As expected, COMPOSE operates favorably when this metric has a value less than 1, may encounter some classification errors around 1, and will ultimately fail beyond values significantly greater than 1. We have also shown through experiment 3 that COMPOSE can recover from a drift at a drift quantified by DCR around 1, if this drift does not remain in effect for an extended period of time.

COMPOSE's relatively poor performance on the Electricity dataset is in fact a testament of the utility of the new metric as well as a confirmation of the algorithm's basic assumptions of operation. This metric also proves that the commonly used electricity data has indeed an abrupt drift. The primary contribution of this work is therefore the formal definition and a quantitative description of the drift rate. The experiments demonstrate that the DCR is indeed a suitable metric, and can be used for finding the areas of drift that will lead to failure of concept drift algorithms that make the gradual drift assumption. We note while labeled data is needed to compute the DCR, in practice it would not be uncommon to have several annotated timesteps to perform the DCR analysis, and to determine if the environment is suitable for learning by algorithms that make the gradual drift assumption.

Our future work includes tailoring the DCR metric to different types of data using more appropriate distance metrics. Also, more work needs to be done to normalize the metric when using non-linear distance metrics. The ultimate goal would be to modify the DCR to work in an ILNSE where it does not need labeled information to calculate the metric. It then can be run alongside COMPOSE or any other concept drift algorithm in real time to detect drift before the data is misclassified. The DCR is a positive first step in quantifying what limited drift means and can be applied to different types of classification algorithms and datasets.

References

[1] K. Dyer, R. Capo, and R. Polikar, "COMPOSE: A semi-supervised learning framework for initially labeled nonstationary streaming data," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 25, no. 1, pp. 12-26, 2014.

[2] K. Dyer and R. Polikar, "Semi-supervised learning in initially labeled and nonstationary environments with gradual drift," *The 2012 International Joint Conference on Neural Networks (IJCNN), 2012*, Brisbane, Australia, pp. 1-9.

[3] Alippi, C.; Boracchi, G.; Roveri, M., "An effective just-in-time adaptive classifier for gradual concept drifts," Neural Networks (IJCNN), *The 2011 International Joint Conference on Neural Networks (IJCNN)*, vol., no., pp.1675,1682, July 31 2011-Aug. 5 2011

[4] D. Gregory and R. Polikar. "Incremental learning of concept drift from streaming imbalanced data." Knowledge and Data Engineering, *IEEE Transactions on Knowledge and Data Engineering* 25.10 (2013): 2283-2301.

[5] G. Krempl, The algorithm APT to classify in concurrence of latency and drift, in *Advances in Intelligent Data Analysis* (Lecture Notes in Computer Science), vol. 7014, J. Gama, E. Bradley, and J. Hollmen,Eds. Berlin Heidelberg, Germany: Springer-Verlag, 2011, pp. 222233.

[6] T.Alexey. "The problem of concept drift: definitions and related work." Computer Science Department, Trinity College Dublin 106 2004.

[7] X. Zhu, "Semi-supervised learning literature survery," Computer Sciences, University of Wisconsin-Madison, Tech. Rep., 2002.

[8] X. Zhu and Z. Ghahramani, "Learning from labeled and unlabeled data with label propagation," Carnegie Melon University, Pittsburgh, PA, Rep. CMU-CALD-02-107, 2002.

[9] R. Capo, A. Sanchez, and R. Polikar, "Core supper extraction for learning from initially labeled nonstationary environments using COMPOSE," *The 2014 International Joint Conference on Neural Networks (IJCNN)*, Beijing, China, 2014.

[10] R. Capo, K. Dyer, and R. Polikar, "Active learning in nonstationary environments," *Neural Networks (IJCNN),The 2013 International Joint Conference on Neural Networks (IJCNN)*, Dallas, TX, 2013.

[11] S. Kullback and R. A. Leibler, "On information and sufficiency," *Annals of Mathematical Statistics*, vol. 22, pp. 49-86, 1951.

[12] M. Hazewinkel, *Encyclopaedia of mathematics: an updated and annotated translation of the Soviet "Mathematical encyclopaedia"*. Dordrecht Boston Norwell, Ma. U.S.a: Reidel Sold and distributed in the U.S.A. and Canada by Kluwer Academic Publishers, pp. 188.

[13] M. Harries, "SPLICE-2 Comparative Evaluation: Electricity Pricing," University of New South Wales, Australia, Rep. UNSW-CSE-TR-9905, 1999.