CaFe DBSCAN: A Density-based Clustering Algorithm for Causal Feature Learning

Pascal Weber[a], Lukas Miklautz[a], Akshey Kumar[a], Moritz Grosse-Wentrup[b], Claudia Plant[b]
[a]*Doctoral School of the University of Vienna*
[b]*Data Science@Uni Vienna*
Faculty of Computer Science, University of Vienna
Vienna, Austria
Email: {firstname.lastname}@univie.ac.at

***Abstract*—Causal Feature Learning (CFL) infers macro-level causes (e.g., an aggregation of pixels in a traffic light image) from micro-level data (e.g., pixels of the image) by clustering the predicted probabilities of effect states (e.g., state of the traffic light). The current method for CFL uses a two-step procedure. First, a classifier for the effect states is trained, and afterwards, the predicted effect state probabilities are clustered. With CaFe DBSCAN, we present a novel density-based clustering method that conducts CFL directly by estimating conditional probabilities during clustering. To this end, we introduce the notion of clustering regions with similar conditional probabilities of the effect states given their micro-level data points. Our single-step approach has the following benefits: (1) CaFe DBSCAN introduces a comprehensive approach to Causal Feature Learning. Unlike existing methods, CaFe DBSCAN uses a probabilistic framework and does not require separate classification and clustering steps implemented by different algorithms relying on various assumptions, parameter settings, and optimization goals. (2) We do not need to train and tune a classifier first, hence the algorithm is more runtime-efficient than the current approach. (3) Due to the properties of density-based clustering algorithms, CaFe DBSCAN is robust against noise and outliers, which leads to purer clusters. (4) Our algorithm automatically infers a reasonable number of clusters, i.e., macro-level causes. We demonstrate the benefits of CaFe DBSCAN on synthetic and real-world data.**

***Index Terms*—Causal Feature Learning, Density-based Clustering, Macro-level Causal Effects**

I. Introduction

Inferring macro-level causes from micro-level data is one of the key pursuits in natural sciences. This task is concerned mainly with finding the right granularity, i.e., how to group micro-level features into macro-level ones. Applications are manifold, e.g., in neuroscience, determining which neurons are responsible for a specific action [1], in computer vision and human-computer interaction, determining which aggregation of pixels in an image caused a specific behavior in a user [2], or meteorology, which composition of micro-level features, like temperature and wind strength, is causing which weather phenomenon [3]. A real world example is shown in Figure 1, different aggregations of micro-level pixels are (not) responsible for the state of a traffic light image (see Section VII).

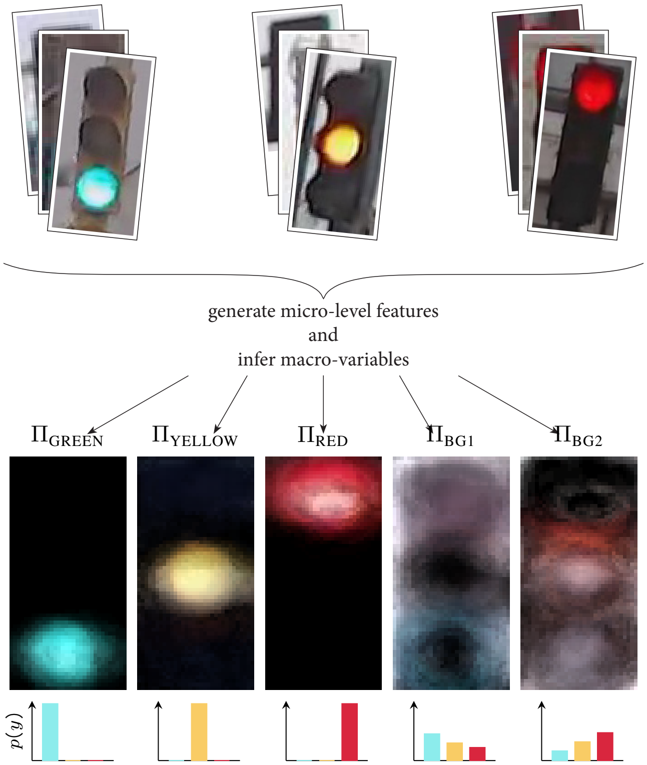

Fig. 1. Real World Experiment: Traffic Light Images.
We generated micro-level features of traffic light images and identified the observational classes Π_{GREEN}, Π_{YELLOW}, Π_{RED}, Π_{BG1}, Π_{BG2} with CaFe DBSCAN. Each class is represented by an image where the sum of all data points belonging to their respective class is scaled to $[0, 255]$. Under each representative, the corresponding effect state probability distribution is shown.

Causal Feature Learning (CFL) is a framework that combines elements from machine learning, data mining, and causal inference. Its task is to find macro-variables from given micro-level data and their corresponding caused effects. These macro-variables can be learned by clustering the conditional probabilities of the effect states, e.g., the observed action of a user, given the micro-level data, e.g., the image they were shown. The knowledge of the found macro-variables can then be used to learn causal relations, by doing $\mathcal{O}(k)$ experiments/interventions, where we check the causal influence of each of the k learned macro-variables. After performing the interventions, one can eliminate non-causal macro-variables to construct a "causal data set". This causal data set can then be used to train a classifier that faithfully represents the underlying causal relations. E.g., in Figure 1 CAFE DBSCAN found two different background macro-variables. Apparently, the used data set consists of more red traffic light images recorded in the night, whereas the green traffic lights were more often recorded during day time. With human knowledge or by performing experiments, we can then deduce that the background lighting should not have an effect on the state of the traffic light (see Section VII).

Chalupka et al. proposed in [4] an algorithm that first learns the conditional probabilities with a classifier and then clusters the resulting learned class probabilities with k-means. This two-step solution relies heavily on the performance of the used classifier. The classifier and the clustering method rely on different assumptions and follow different optimization goals. Furthermore, classifier training and hyperparameter tuning are computationally expensive. Additionally, this approach assumes that we already know the number of clusters apriori and that the data contains no noise or outlier points, all of which limit its applicability in practice.

In our novel density-based clustering algorithm for CFL, **Ca**usal **Fe**ature **DBSCAN** (CAFE DBSCAN), we solve these problems. We do this by estimating the conditional probabilities through ϵ-neighborhoods. These estimations are computed during the clustering process, which results in a single-step approach with no need to train a classifier. Our main contributions are the formulation of how to cluster conditional probabilities in the CFL setting meaningfully to learn macro-level variables from micro-level ones and, based on that, the development of our algorithm. In our experiments in Section VI, VII, and VIII, we verify that our algorithm outperforms the current state-of-the-art in the following aspects:

1) **Standalone algorithm**. CAFE DBSCAN introduces a comprehensive approach to Causal Feature Learning. Instead of learning the conditional probabilities first and clustering them afterwards, we integrate the estimation of the probability distributions into the clustering algorithm. This makes our algorithm more faithful to the underlying data and improves the clustering performance.
2) **Runtime**. The training and tuning of a classifier, which is needed for the two-step approach of CFL, in addition to the tuning of the clustering algorithm, is time-consuming. Our novel algorithm does not need a classifier. Thus, it finds the clusters magnitudes faster than the current state-of-the-art.
3) **Robustness to noise**. The micro-level data are often contaminated by noise. Noise and outliers have a negative impact on the accuracy of the classifier and the subsequent algorithm that then clusters the learned conditional probabilities. CAFE DBSCAN is robust against noise and outlier points due to its relation to DBSCAN, making it more suitable for real-world data and allowing us to learn purer clusters that are not contaminated by noise.
4) **Automatic learning of the number of clusters**. The current approach of CFL in [4] assumes that the number of clusters is already known in advance. Our approach automatically determines a reasonable number of clusters, as shown by experiments in Section VI.

II. Background and Related Work

There are two main research areas related to our work: *clustering* and *causality*. Regarding clustering, we focus on density-based clustering, i.e., the DBSCAN family, which is described briefly in Section II-A. In Section II-B, we emphasize the difference between clustering in CFL and other existing clustering methods. For causality, we briefly describe how CFL relates to existing work on causality in Section II-C. Finally, in Section II-D, we formally introduce the CFL framework.

A. Density-based Clustering Algorithms

Clustering is the task of finding groups in a data set. For density-based clustering, "a cluster is a set of data objects spread in the data space over a contiguous region of high-density objects" [5]. High-density clusters are then separated by low-density regions and points in these regions are considered as outliers or noise points. The original density-based clustering algorithm DBSCAN [6] can find arbitrary-shaped clusters, automatically determines the number of clusters, and identifies noise points and outliers. Our algorithm CAFE DBSCAN is also part of the density-based clustering family and inherits these advantages.

There exists a plethora of work on density-based clustering spanning more than twenty years of research. There are many extensions of DBSCAN, e.g., for high-dimensional data [7]–[9], for very large data sets [10], varying densities [11] or automatic estimation of ϵ [12]. CAFE DBSCAN is the first density-based clustering approach that directly solves CFL. Hence, we focus on its core properties and neglect all these extensions. Nonetheless, it would be interesting for future work to adopt these existing extensions for CFL as well.

B. Clustering of Conditional Probabilities

For clustering, we always need the notion of similarity. In CFL, similarity is defined w.r.t. the conditional probabilities of the effect states Y given their (micro-level) data X, i.e., two data points in X are considered similar if they have a similar conditional probability $P(Y \mid X)$, we will explain this in more detail in Definition 2.

We want to highlight that this is different from the "usual" clustering setting in which the goal is to infer cluster labels from the plain data X. Note, that this is also different from just including the effect state labels Y into our data and clustering those, which would correspond to clustering the samples of the joint distribution of data and effect states. (Semi-)supervised clustering algorithms [13] are also different because they only (partially) exploit the label information to get a better clustering. In contrast to CFL, they still cluster the data X just with the help of the information of Y, but they are not clustering the conditional probability $P(Y \mid X)$. Probabilistic clustering methods like EM [14] are also different from the CFL setting in that they learn the probability distribution $P(X)$ in the plain data, but are again not clustering with respect to the conditional probability.

To our knowledge, clustering of conditional distributions for CFL has not been studied in the clustering community. There exists related work from statistics on the clustering of probability distributions [15], [16]. Importantly, these methods assume that the probability distributions are already given, which is not the case in the CFL setting. One could estimate the conditional probability distributions with a classifier first, but this would lead to the same problems as in the current CFL approach of [4].

C. Causality

In causality, we originally only focus on causal macro-variables [17], e.g., the traffic light state, and their influence on other macro-variables, e.g., the behavior of people recognizing this state. However, these macro-variables are often an aggregation of smaller micro-variables, e.g., pixels of an image. In CFL, we now want to learn these aggregations of micro-level variables automatically. Considering the traffic light example, this would amount to finding the parts of the image (green, yellow, and red areas) which cause the traffic light state, but also the background of the image that does not influence the traffic light state. Logically, with only observational data, we can also only find *observational* macro-variables, but they are a great help for designing experiments, as they indicate in which areas we have to intervene in the system to find the *causal* macro-variables. In the traffic light example, we can already deduce the causal macro-variables by ourselves without experiments with common sense.

D. Causal Feature Learning (CFL)

CFL is the task of learning causal macro-variables given (high-dimensional) micro-level data and their respective effect states. [18] calls this problem *Causal Representation Learning* and states that it is “a central problem for AI and causality”. However, their work does not solve this problem, but addresses basic concepts of causal inference and its relationship to machine learning under the premise that all causal macro-variables are already given. Interestingly, CFL was already introduced earlier in [4] but not mentioned in their work. In the following sections, we adopt the notion of [4] and state the main ideas of CFL.

Let $\mathcal{X}$ be the domain of the micro-level data and X a random variable that takes on values $x \in \mathcal{X}$. Further, let $\mathcal{Y} = \{1, 2, \dots, m\}$ denote the discrete and finite domain of effect states[1] and Y a random variable that can take on values $y \in \mathcal{Y}$. The intuitive interpretation of an effect state is the condition, state, or situation which is observed for a given micro-level data point. E.g., the effect state of all the pixels in one traffic light image is the state of the traffic light in this image.

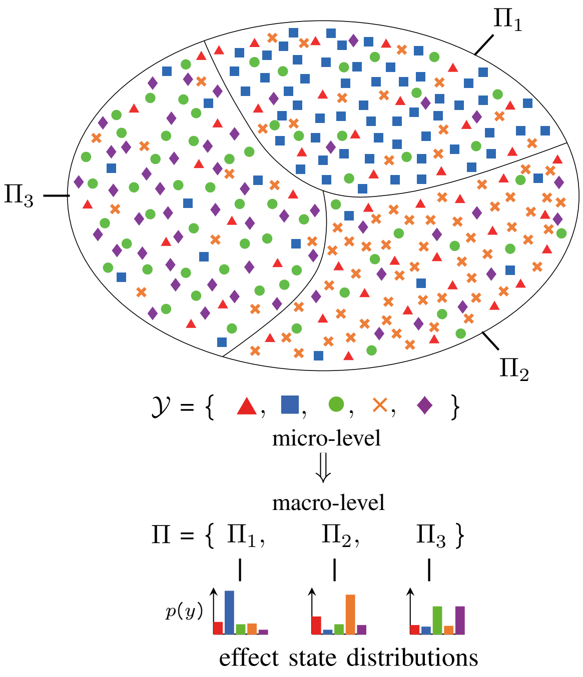
Π_1
Π_3
Π_2
$\mathcal{Y}$ = { ▲, ■, ●, ×, ◆ }
micro-level
⇓
macro-level
Π = { Π_1, Π_2, Π_3 }
$p(y)$
effect state distributions

Fig. 2. Example with $m = 5$ effect states. The micro-level data points with their respective effect states are visualized in the top figure. Each effect state is represented by a unique shape/color. The causal classes Π_i can each be identified by a unique probability distribution (see distribution plots). By clustering the micro-level features by their conditional probabilities the classes Π_i can be identified.

In causality, interventions are an important concept. By applying them, we can deduce which macro-variables are causal and which are only spurious. However, if the macro-variable consists of several micro-variables, we need to intervene in the micro-level domain to find out, which aggregations of micro-variables are causal. In the traffic light example, one possible intervention would be setting an aggregation of pixels to a specific color. Such a micro-level intervention is denoted with $do(X = x)$ which means that X is set to x while the rest of the system remains unperturbed in all micro-level data points. Here, X and x can also be sets, which means that the *do*-notation can also be applied to an aggregation of micro-level variables. If we can apply interventions in the micro-level domain, we are also able to deduce causal partitions:

Definition 1. (causal partition)
The causal partition $\Pi^{(c)}(\mathcal{Y}, \mathcal{X})$ of $\mathcal{X}$ w.r.t. the effect states $\mathcal{Y}$ is the partition induced by the equivalence relation $\sim$ such that $x \sim x'$ with $x, x' \in \mathcal{X}$ iff

$$P(Y \mid do(X = x)) = P(Y \mid do(X = x')). \qquad (1)$$

The causal partition groups micro-level data together if they still have the same conditional probability after applying the same intervention on all the micro-level data. Such a group is then called *causal class*. In the traffic light example, there exist four causal classes: the green, yellow, and red

[1] Without loss of generality, we can assume the states to be numbered from 1 to m. Every discrete, finite domain with size m can be mapped to this range of numbers.

areas, respectively, and the background area. Whereby only the former three are really causal for the traffic light state.

Often, we can not intervene on micro-variables, e.g., due to ethical considerations (medicine) or physical constraints (weather). Additionally, there are complexity constraints that make interventions in the micro-level domain infeasible because we would have to intervene on all possible combinations of micro-variables for all data points. In the traffic light example, we would need to intervene on all possible aggregations and combinations of pixels and their values, which would be infeasible in practice. To prevent this explosion in complexity, we can deduce an observational partition to reduce the number of interventions to $\mathcal{O}(k)$, where k is the number of observational classes:

Definition 2. (observational partition)
The observational partition $\Pi^{(o)}(\mathcal{Y}, \mathcal{X})$ of $\mathcal{X}$ w.r.t. the effect states $\mathcal{Y}$ is the partition induced by the equivalence relation $\sim$ such that $x \sim x'$ with $x, x' \in \mathcal{X}$ iff

$$P(Y \mid X = x) = P(Y \mid X = x'). \tag{2}$$

A cell of the observational partition $\Pi_i^{(o)} \in \Pi^{(o)}$ is also called *observational class*, where $i \in [1..k]$ and $k = |\Pi^{(o)}|$ is the total amount of observational classes.

Put in plain words, the observational partition groups the observed micro-level data into observational classes that have the same conditional probability over the effect states. In Figure 2, an example with five possible effect states, three observational classes, and their corresponding effect state distributions are illustrated.

In [4] it is proven (see Theorem 1) that in general, the causal partition is a coarsening of the observational partition, i.e., the causal partition aligns with the observational partition. This offers valuable information for practitioners, e.g., on how to perform experiments/interventions to find the causal partition.

Theorem 1. (causal coarsening theorem) [4]
Among all the generative distributions which induce a given observational partition $\Pi^{(o)}$, all except for a subset of Lebesgue measure zero induce a causal partition $\Pi^{(c)}$ that is a coarsening of $\Pi^{(o)}$.

Considering this theorem, it is a good strategy to first find a meaningful observational partition and then use this information to design experiments. Through $\mathcal{O}(k)$ experiments/interventions, where k is the number of observational classes, we will obtain the causal partition, which then can be used for learning a causal classifier. For this reason, we focus on finding a good observational partition $\Pi^{(o)}$ and simplify the notation for it with Π.

So far, CFL has been performed in a two-step process that involves training a neural net to estimate the conditional probability distribution, followed by clustering in the probability space [3], [4], [19]. Here, we present an algorithm that can directly learn the partition from the micro-level data.

III. Formal Definition

In this section, we formalize the notion of density-based clustering in the context of CFL. We use the notation of CFL and density-based clustering to introduce the concept of clustering conditional probabilities.

Symbol	Interpretation
d	Dimensionality of micro-level data
m	Number of different effect states
n	Number of observed data points
k	Number of observational classes
$\mathcal{X} \subseteq \mathbf{R}^d$	Domain of micro-level data
X	Random variable of the micro-level domain $\mathcal{X}$
x, x', x''	A realization of variable X
$\mathcal{Y} = \{i\}_{i=1}^{m}$	Domain of effect states
Y	Random variable of the effect state domain $\mathcal{Y}$
y	Target state of $\mathcal{Y}$
$\epsilon \in \mathbf{R}^+$	Parameter for the ϵ-neighborhood
$\mu \in \mathrm{N}$	Parameter for minimum number of points
$\tau \in [0..1]$	Threshold parameter (see Def. 5)
$\mathit{dist}(\cdot, \cdot)$	A distance function, e.g., the Euclidean norm
$\mathcal{N}_\epsilon(x)$	The ϵ-neighborhood of x
$\delta(\cdot, \cdot)$	A probability distance function
$\mathcal{D}$	Observed data points $\{(x_i, y_i)\}_{i=1}^{n}$
$\mathcal{D}_X$	Observed micro-level data points $\{x_i\}_{i=1}^{n}$
$\mathcal{D}_Y$	Observed effect states $\{y_i\}_{i=1}^{n}$
x_i	i^{th} micro-level data point observation
y_i	i^{th} effect state observation
$\Pi = \Pi^{(o)}$	Observational partition
$\Pi_i = \Pi_i^{(o)}$	Observational class, equivalent to a cluster

TABLE I
List of Notations.

Let $\mathcal{D} = (\mathcal{D}_X, \mathcal{D}_Y) = \{(x_i, y_i)\}_{i=1}^{n}$ be a set of n observed micro-level data points and their respective effect states. We assume the observed effect states $\mathcal{D}_Y$ to be sampled i.i.d. from the effect state domain $\mathcal{Y}$ by k different not known probability distributions $P_i(Y \mid X = x)$ with $x \in \Pi_i$ for $i \in [1, k]$. The function $\mathit{dist}(\cdot, \cdot)$: $\mathbb{R}^d \times \mathbb{R}^d \rightarrow \mathbb{R}$ returns the metric distance between two points $x, x' \in \mathcal{D}_X$, e.g., the Euclidean norm ($\| x - x' \|_2$). With an $\epsilon \in \mathbb{R}^+$ the ϵ-neighborhood of $x \in \mathcal{D}_X$ is denoted by $\mathcal{N}_\epsilon(x)$, i.e., all points x' with $\mathit{dist}(x, x') \leq \epsilon$ are included in the set $\mathcal{N}_\epsilon(x)$ (cf. [6]). The ϵ-neighborhood of a set $\mathcal{N}_\epsilon(\mathcal{S})$ contains all points which are included in the ϵ-neighborhood of all points in the set $\mathcal{S}$, i.e., $\mathcal{N}_\epsilon(\mathcal{S}) = \{\mathcal{N}_\epsilon(x) \mid x \in \mathcal{S}\}$. Let Π be an observational partition with the corresponding observational classes $\Pi_i \in \Pi$. Then each observational class Π_i is characterized by a corresponding probability distribution $P_i(Y \mid X)$ and for all points $x \in \Pi_i$ the probability $P_i(Y \mid X = x)$ is equal.

In the following, we describe how we estimate the probability distribution of a *probability region* and how it should be extended such that the probability distribution remains the same within one observational class Π_i.

We do not know the exact probability distributions $P_i(Y \mid X = x)$ with $x \in \Pi_i$ for $i \in [1..k]$, thus, we can only estimate them. Therefore, we consider a subset $\mathcal{D}' \subseteq \mathcal{D}$ in which we assume that the distribution of the effect states does not change. The random variable X then can take on values in ${\mathcal{D}'}_X$. We first estimate the probability for one specific effect state y of ${\mathcal{D}'}_Y$ with

$$\hat{P}(Y = y \mid X) = \frac{|\{(x_i, y_i) \in \mathcal{D}' \mid y_i = y\}|}{|\mathcal{D}'|}. \quad (3)$$

Definition 3. (estimated probability distribution of ${\mathcal{D}'}_{X,Y}$)
We then can define the *estimated probability distribution* of $\mathcal{D}'$, $\hat{P}(\mathcal{D}')$, which contains the conditional probabilities of all possible effect states in $\mathcal{D}'$:

$$\hat{P}(\mathcal{D}') = \{\hat{P}(Y = 1 \mid X), \ ..., \ \hat{P}(Y = m \mid X)\} \quad (4)$$

Note that as a discrete probability distribution the sum of all the estimates of the conditional probabilities of the effect states in one estimated probability distribution always sums up to 1, i.e.,

$$|\hat{P}(\mathcal{D}')| \ = \sum_{y \in \mathcal{Y}} \hat{P}(Y = y \mid X) = 1 \quad (5)$$

Definition 4. (probability region)
Let $\epsilon \in \mathbb{R}^+$ and $\mu \in \mathbb{N}^+$. A *probability region* is a subset $\mathcal{D}'$ of $\mathcal{D}$ w.r.t. ϵ and μ, if

1) $\exists x \in \mathcal{D}' : \forall x' \in \mathcal{D}', x' \in \mathcal{N}_\epsilon(x)$
2) $|\mathcal{D}'| \geq \mu$ (core region condition).

A probability region is a region where all points are within the neighborhood of one point and which contains enough points for an initial estimation of the probability distribution.

We assume, that for small $\mathcal{D}'$ the conditional probability of the effect states y_i given their data points x_i is the same for all $(x_i, y_i) \in \mathcal{D}'$, hence, these points can be grouped together and their conditional probability can be estimated with the probability distribution of $\mathcal{D}'$.

Definition 5. (directly reachable probability regions)
Let $\epsilon, \tau \in \mathbb{R}^+$, $\mu \in \mathbb{N}$, and $\delta(\cdot, \cdot)$ be a statistical probability distance. A probability region $\mathcal{D}''$ is *directly reachable* from another probability region $\mathcal{D}'$ w.r.t. ϵ, τ, μ, and $\delta(\cdot, \cdot)$ if

1) $\exists x \in \mathcal{D}' : x \in \mathcal{N}_\epsilon(\mathcal{D}'')$
2) $\delta(\hat{P}(\mathcal{D}'), \hat{P}(\mathcal{D}'')) \leq \tau$.

Intuitively, two probability regions are directly reachable if they share a point x and the estimated probability distributions of the two regions do not differ too much. For the latter, we introduce the threshold variable τ and use the statistical probability distance function δ. As our domain of effect states $\mathcal{Y}$ is categorical and we are only interested in the difference of the conditional probabilities of each effect state y, we chose the total variation distance (see [20] for an overview on comparing distributions). In this case, the total variation distance is related to the Manhattan distance with $\delta(P, Q) = \frac{1}{2}||P - Q||_1$ [21, proposition 4.2, p. 48]. Hence, the δ-distance is bounded between $[0, 1]$ and thus τ should be chosen from this interval.

Definition 6. (connected probability regions)
A probability region $\mathcal{D}'$ is connected to another probability region $\mathcal{D}''$ if there exists a chain of probability regions $\mathcal{D}_1, ..., \mathcal{D}_l$ with $\mathcal{D}_1 = \mathcal{D}'$ and $\mathcal{D}_l = \mathcal{D}''$ such that $\mathcal{D}_{i+1}$ is directly reachable from $\mathcal{D}_i \ \forall i \in [1, l-1]$.

All connected probability regions are characterized by the same estimated probability distribution and form clusters:

Definition 7. (cluster)
Π_i is a cluster in $\mathcal{D}$, i.e., an observational class of $\mathcal{D}$, w.r.t. ϵ, τ, and μ if

1) $\Pi_i \neq \emptyset$
2) $\forall$ *probability regions* $\mathcal{D}', \mathcal{D}''$: if $\mathcal{D}' \in \Pi_i$ and $\mathcal{D}''$ are *connected* to $\mathcal{D}' \Rightarrow \mathcal{D}'' \in \Pi_i$.

The conditional probability $P_i(Y \mid X)$ in cluster Π_i is then estimated by the probability distribution $\hat{P}(\{x \mid \forall x \in \Pi_i\})$. Every cluster Π_i represents an observational class where the estimated probability distribution of the effect states does not vary much.

Definition 8. (noise)
Let Π be the set of detected clusters, i.e., the observational partition, of $\mathcal{D}$ w.r.t. the parameters ϵ, τ, and μ. All points $n \in \mathcal{D}$ which do not belong to any cluster $\Pi_i \in \Pi$ are considered as noise, i.e.,

$$\textit{noise} = \{n \in \mathcal{D} : n \notin \Pi_i \mid \forall \, \Pi_i \in \Pi\}. \quad (6)$$

A noise point is a point that does not contribute to any of the observational classes and can therefore be excluded from the CFL. By discarding these noise points we get purer observational classes, which also makes experiments/interventions on them more reliable by only considering non-noise points for experiments.

IV. The CaFe DBSCAN Algorithm

Our algorithm CaFe DBSCAN is a density-based clustering method that finds the observational partition Π. In addition to the parameters ϵ and μ of DBSCAN, our method needs a threshold parameter τ which indicates how much the estimated probability distribution of different probability regions can differ to still be considered as connected. The core concept of CaFe DBSCAN is that it extends an initial probability region until the distribution of neighboring regions changes. Eventually, in addition to the clusters, i.e., observational classes Π_i, CaFe DBSCAN finds the estimation of the effect state distributions $\hat{P}(\Pi_i)$. It also separates adjacent observational classes with different effect state distributions, while this would not be possible for DBSCAN alone.

The pseudo-code of CaFe DBSCAN is stated in Algorithm 1. We will now go through the algorithm step-by-step. In the beginning, we regard every ϵ-neighborhood as a possible start region (Line 1 and 2). If we find a region with enough points, i.e., the number of unclustered points in the ϵ-neighborhood exceeds μ (Line 3), we introduce a new cluster Π_i (Line 4). Here the ϵ and μ parameters are responsible for the initial estimation of the effect state distribution of a cluster. We

Algorithm 1: CaFe DBSCAN

```
Param.: $\epsilon$, $\mu$, and $\tau$
Input  : Observed data points $\mathcal{D}$
Output: Observational partition $\Pi$ of $\mathcal{D}$
1  foreach unclassified $x \in \mathcal{D}$ do
2      $\mathcal{D}' = \{x' \mid x' \in \mathcal{N}_\epsilon(x), x'$ is unclassified$\}$;
3      if $|\mathcal{D}'| \geq \mu$ then
4          generate new $\Pi_i$;
5          assign all $x' \in \mathcal{D}'$ to $\Pi_i$;
6          insert all $x' \in \mathcal{D}'$ into queue $\Phi$;
7          while $\Phi \neq \varnothing$ do
8              $x'$ = first point in $\Phi$;
9              $\mathcal{D}'' = \{\mathcal{D}'' \mid x'' \in \mathcal{N}_\epsilon(x'), x''$ is unclassified$\}$;
10             if $\delta(\hat{P}(\mathcal{D}''), \hat{P}(\Pi_i)) \leq \tau$ then
11                 assign all $x'' \in \mathcal{D}''$ to $\Pi_i$;
12                 insert all $x'' \in \mathcal{D}''$ into queue $\Phi$;
13             end
14             remove $x'$ from $\Phi$;
15         end
16     end
17 end
18 mark all unclassified $x \in \mathcal{D}$ as noise;
```

assign all the points of this initial ϵ-neighborhood to cluster Π_i (Line 5) and insert them into a queue Φ (Line 6). Now we go through all points of this queue and calculate the ϵ-neighborhood for each of them (Line 9). We compare the estimated probability distribution of each ϵ-neighborhood with the one of the current cluster (Line 10). If the difference is within the margin of the threshold parameter τ, we enclose all the points in this ϵ-neighborhood into the current cluster Π_i (Line 11) and again insert all the points into the queue Φ (Line 12). Since the effect state distribution within a cluster, i.e., observational class, stays the same, the estimation of the effect state distribution of the observational class gets more accurate with every ϵ-neighborhood which is enclosed in the cluster. In the end, all remaining unclassified data points are marked as noise (Line 18).

V. Determining Parameters ϵ, μ, and τ

We will briefly summarize the interpretation for the parameters ϵ, μ, and τ and propose an approach on how to find good values for them.

ϵ-parameter

The ϵ-parameter defines the size of an ϵ-neighborhood, with which we are estimating the effect state distribution. Thus ϵ also influences the accuracy of the estimation.

μ-parameter

The μ-parameter adjusts the sensitivity to noise and the accuracy of the estimated effect state distribution of the initial probability region.

τ-parameter

The τ-parameter specifies the maximum allowed distance between the effect state distributions of two regions.

Setting ϵ, μ, and τ: We start by setting μ high enough so that the number of clusters is in a sensible range. For our synthetic data set, values $\mu \in [10, 100]$ worked best. Secondly, we chose ϵ such that multiple ϵ-neighborhoods will contain more than μ points. Note, that the ϵ and μ parameters influence each other, so for increasing ϵ, one should also increase μ. Finally, the value for τ slightly depends on ϵ. For small ϵ values, the estimation of the effect state distribution is less accurate, hence the value for τ can be chosen high, e.g, $\tau = 0.3$. If we want more observational classes with more accurate effect state distributions, the value for τ should be set low, e.g., $\tau = 0.1$. In our experiments, values in the range $[0.1, 0.4]$ worked best.

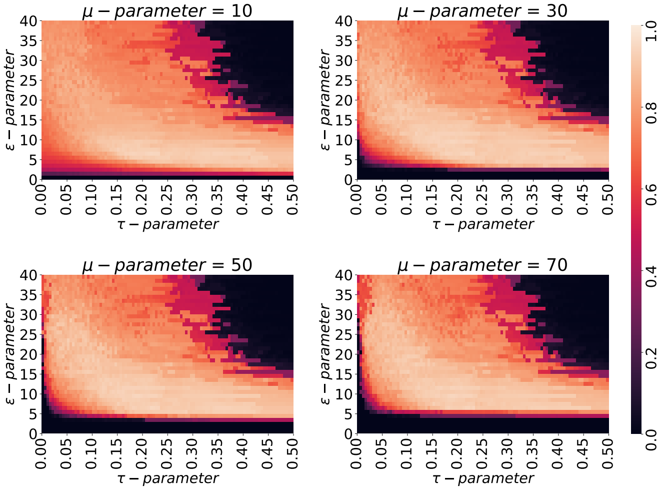

Fig. 3. Hyperparameter sensitivity w.r.t. NMI for DS2.

The impact of these parameters from our algorithm is shown in a hyperparameter sensitivity experiment in Figure 3. Overall, we see that the sensitivity landscape contains large regions with high NMI. The parameters are dependent on each other. For higher values for μ, we also need greater ϵ-neighborhoods and hence a larger value for ϵ. For very high ϵ values, the algorithm merges clusters and the NMI gets worse, but we can still separate adjacent clusters if we set τ to a small value. Over the different μ values, the NMI performance is in a similar range for $\tau \in [0.1, 0.4]$. We see that our algorithm returns stable results for various settings of the hyperparameters.

VI. Experiments

The following is an overview of the conducted experiments that we use to demonstrate our contributions.[2] We first describe and motivate our **experiment setup**. We use this setup to analyze our **cluster effectiveness** and show how we outperform the current CFL approach in the presence of noise, while automatically determining a reasonable number of clusters. We conduct a **runtime comparison** and we have already analyzed the **hyperparameter sensitivity** of our algorithm in the previous section (see Figure 3). Additionally, we performed

[2] Our Python implementation of CaFe DBSCAN, the performed experiments, and their visualizations are made available here: https://gitlab.cs.univie.ac.at/pascalw777/cafe-dbscan/

Method	**DS1: Separated Circles** (n = 3, 496, d = 2)				**DS2: Adjacent Circles** (n = 4, 416, d = 2)				**DS3: Rectangles / Circles** (n = 9, 343, d = 2)			
	Clf. acc.	k	NMI	ARI	Clf. acc.	k	NMI	ARI	Clf. acc.	k	NMI	ARI
Ground Truth	1.00	6(+1)	1.00	1.00	1.00	7(+1)	1.00	1.00	1.00	7(+1)	1.00	1.00
CaFe DBSCAN	-	6(+1)	0.97	0.97	-	7(+1)	0.94	0.94	-	7(+1)	0.86	0.85
CaFe DBSCAN$^+$	-	6(+1)	**0.98**	**0.99**	-	7(+1)	**0.95**	**0.95**	-	7(+1)	**0.91**	**0.91**
LR / k-Means	0.33	6*	0.66	0.45	0.50	7*	0.79	0.74	0.30	7*	0.40	0.18
RF / k-Means	0.33	6*	0.66	0.46	0.50	7*	0.68	0.54	0.32	7*	0.78	0.70
SVC / k-Means	0.33	6*	0.31	0.13	0.50	7*	0.59	0.40	0.27	7*	0.38	0.15
MLP / k-Means	0.33	6*	0.65	0.41	0.52	7*	0.76	0.65	0.34	7*	0.79	0.67
LR / DBSCAN	0.33	6(+1)	**0.98**	0.98	0.50	5(+1)	0.86	0.70	0.30	10(+1)	0.60	0.37
RF / DBSCAN	0.33	6(+1)	0.74	0.60	0.50	45(+1)	0.69	0.50	0.32	10(+1)	0.77	0.70
SVC / DBSCAN	0.33	5(+1)	0.51	0.28	0.50	4(+1)	0.64	0.42	0.27	5(+1)	0.59	0.33
MLP / DBSCAN	0.33	9(+1)	0.86	0.77	0.52	13(+1)	0.87	0.83	0.34	20(+1)	0.77	0.66
LR / HDBSCAN	0.33	6(+1)	0.90	0.83	0.50	5(+1)	0.87	0.71	0.30	27(+1)	0.52	0.25
RF / HDBSCAN	0.33	6(+1)	0.72	0.59	0.50	17(+1)	0.65	0.36	0.32	21(+1)	0.60	0.32
SVC / HDBSCAN	0.33	20(+1)	0.69	0.46	0.50	9(+1)	0.84	0.69	0.27	9(+1)	0.51	0.23
MLP / HDBSCAN	0.33	5(+1)	0.88	0.81	0.52	12(+1)	0.87	0.87	0.34	13(+1)	0.78	0.70

TABLE II

Performance of CaFe DBSCAN against different combinations of CFL with different clustering algorithms and classifiers. Best values in bold and runner-up is underlined. For density-based methods (+1) indicates that a noise cluster was found. (LR=Logistic Regression, RF=Random Forest, SVC=Support Vector Classifier, MLP=Multi-Layer Perceptron)

real world experiments on traffic light images in Section VII and weather data in Section VIII.

Experiment Setup: We generated three synthetic data sets to show that the current approach of CFL, which consists of training a classifier and clustering the conditional probabilities with k-means, does not work properly if noise is present. Additionally, we show that it is not enough to just replace k-means with DBSCAN or HDBSCAN, but that indeed a new approach is needed (see Figure 4).

In Table II, we compare CaFe DBSCAN against CFL with different classifiers and clustering algorithms w.r.t. normalized mutual information (NMI) [22] and adjusted rand index (ARI) [23], two commonly used metrics in clustering. For our comparison methods, we choose linear and non-linear classifiers implemented in scikit-learn. We tuned the classifiers with a grid search and a train/test split to find the best hyperparameters. We then trained them on the full data with the best hyperparameters to estimate the conditional probabilities of the effect states. Afterwards, we cluster the conditional probabilities with k-means as in [4] and also with DBSCAN and HDBSCAN [12]. We tuned the parameters of (H)DBSCAN using a grid search and chose the parameters with the highest NMI and ARI. Note, that it is usually not possible to tune a clustering algorithm with access to ground truth labels, but this serves as a strong baseline against our method. For k-means, we set k equal to the number of clusters (marked with *) and report the average of the best ten runs out of 100 for each data set. For our method, we report two results. CaFe DBSCAN, which was tuned with our heuristic described in Section V, and CaFe DBSCAN$^+$, which was tuned with a grid search (like (H)DBSCAN) and serves as a best-case scenario.

Cluster effectiveness: Table II shows that our CaFe DBSCAN$^+$ performs best for all data sets and the version with

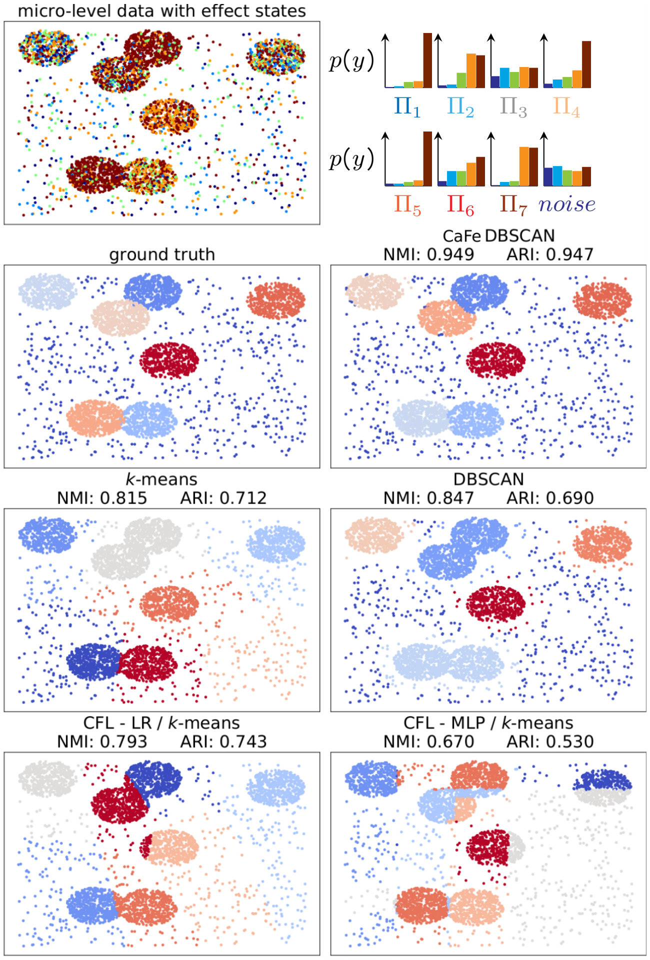

Fig. 4. Top left: The micro-level data points with their effect state in different colors. Top right: The effect state distribution for each macro-level observational class Π_i. Below: Clustering of CaFe DBSCAN; k-means and DBSCAN on the micro-level data; CFL with two different classifiers.

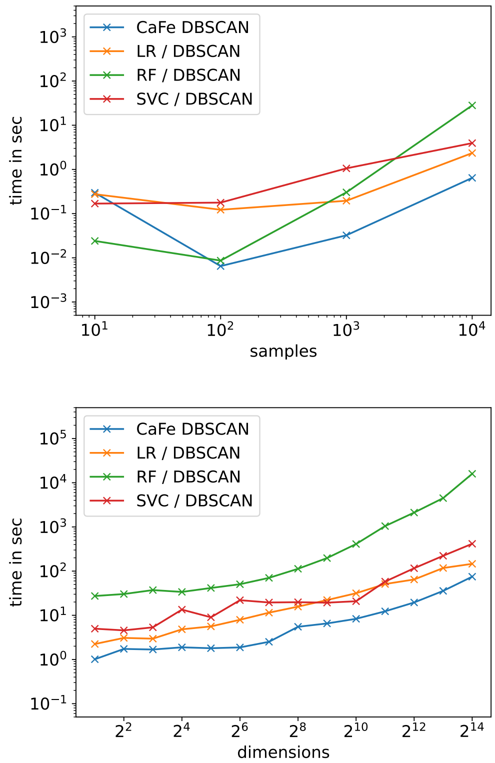

Fig. 5. Runtime for different number of samples and 2 dimensions (top) and different number of dimensions and $10,000$ samples (bottom). For the former CFL algorithm only the abbreviation of the used classifiers are stated. The plot then summarizes the time needed for training the classifier and the k-means clustering. Note, that this excludes the time needed for tuning the classifier with a grid search and cross-validation.

our heuristic from Section V performs second best, for all but DS1. This is not surprising as DS1 contains well-separated clusters and thus DBSCAN is well-performing. For DS2 and DS3, which contain adjacent clusters we outperform all methods by a large margin. In Figure 4, we visualize the clustering results of our method against the CFL approach with k-means for DS2. Additionally, we present the results of applying k-means and DBSCAN only on the micro-level data. We see that applying k-means and DBSCAN only on the micro-level data is not enough to separate the adjacent clusters. Additionally, the classifiers can not handle the noise points very well. In contrast, CaFe DBSCAN can find the clusters almost perfectly, and most importantly it can separate them from noise, leading to purer clusters.

Runtime comparison: We compare the runtime of the CFL for an increasing number of data points and dimensions in Figure 5. We see that CaFe DBSCAN outperforms all methods in the low sample regime and performs similar to Logistic Regression (LR) and Random Forests (RF) for increasing dimensionality. This is of course only a fraction of the runtime needed for the two-step CFL approach because usually the classifiers are trained via cross-validation and tuned via grid search, which can consist of hundreds of runs and not just a single one. The runtime experiments were conducted on a server running Ubuntu 20.04 LTS, using two Intel Xeon processors with 2.20 GHz and 8 GB RAM. For ease of comparison, we report single-core performance.

VII. Real World Experiment: Traffic Light Images

We conducted experiments on real world traffic light images from the *Traffic Light Classifier Project*[3]. We resized 33 green, yellow, and red images into 64×32 pixels and extracted a micro-level data point for each pixel of one image, resulting in $2,048$ samples that all have the same effect state (traffic light state) per image. In total, we obtained $202,752$ micro-level data points and their respective effect state. On this data set, CaFe DBSCAN found five observational classes: BG1, BG2, GREEN, YELLOW, and RED. Each of these observational classes is associated with a macro-variable that represents the aggregation of pixels with their corresponding information (color and location). Figure 1 shows a representative image for each macro-variable.

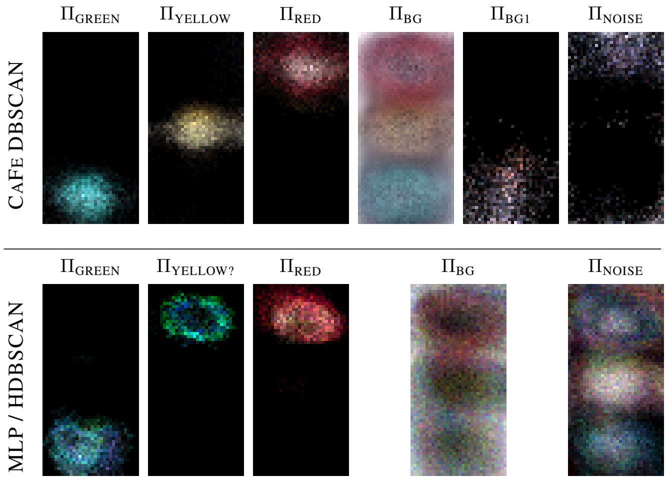

Fig. 6. Top: Observational classes learned by CaFe DBSCAN for traffic light images with noise ratio = 0.8. Bottom: Observational classes learned by an MLP classifier and HDBSCAN clusterer. Each image is the sum of all data points belonging to their respective class scaled to $[0, 255]$.

As mentioned in Section II-D, the traffic light images should be clustered into four causal classes: the green, yellow, and red areas, respectively, and the background area. However, some of the traffic light images were recorded at night and others during the daytime. That is why CaFe DBSCAN detects two different observational classes for the background. In class BG1, we can detect the bright background of images taken during the day. In class BG2, we can see the reflection in the yellow and green lights during the night. Apparently, more green traffic light images were recorded during the day, and more red traffic light images were recorded at night, leading to green artifacts in class BG1 and red artifacts in class BG2. With human understanding, we know that there should only exist one causal class for the background, so we can merge the two observational classes BG1 and BG2 into the causal class

[3] https://github.com/endymioncheung/iSDC-P5-traffic-light-classifier

BG. Contrary, a classifier could mistakenly learn these spurious macro-variables for classification. With the help of the causal macro-variables, a "causal" classifier can be trained to better classify traffic light images.

The probability distribution of the effect states of the causal class BG is the average of the ones in BG1 and BG2. Hence, the causal class BG consists of an almost equal number of pixels from red, yellow, and green traffic light images. Thus, this class does not provide any information about the traffic light state. Contrary, the probability distribution in the green, yellow, and red causal classes indicate that almost all pixels in these classes belong to their respective traffic light state. Hence, the corresponding images in Figure 1 visualize the cause of their respective traffic light states.

We additionally checked the effects of noise on our algorithm and compared it to the state-of-the-art algorithm. To do this, we added a random value between $[-255, 255]$ multiplied by a ratio constant to each pixel. In Figure 6, we can see that CaFe DBSCAN is still able to detect meaningful observation classes for a ratio of 0.8. On the other hand, if we run the classifier/clusterer combination which performed best in Section VI, i.e., learning the probabilities with an MLP first and cluster them with HDBSCAN afterwards, we can not find the yellow observation class anymore.

VIII. Real World Experiment: El Niño

We analyzed a data set of [3] containing daily wind and temperature measurements of a spatial grid with an interval of $2.5^\circ \times 2.5^\circ$ of the $(-10^\circ, +10^\circ)N \times (140^\circ, 280^\circ)E$ equatorial band of the Pacific Ocean of 36 years. These result in $13,140$ west-to-east zonal wind strength (ZWS) and sea surface temperature (SST) data points, each represented by a 9×55 matrix, hence 495 features.

An interesting question is whether there exists an aggregation of micro-variables in the ZWS data points that influences the SST. We applied CaFe DBSCAN to the micro-level measurements of the ZWS and the corresponding averaged SST of the same time. The results can be seen in Figure 7. Two out of the four inferred macro-variables can be interpreted as the climate phenomena El Niño and La Niña.

Altogether the algorithm found four observational classes. The ZWS consists of "Wind to southeast", "Wind from southeast", "High-pressure area", and "Low-pressure area" and their corresponding SST classes "Cold", "Warm", "La Niña", and "El Niño". We can see that the winds to the southeast and northwest probably create a small low-pressure area, where the cold air from higher altitudes is influencing the SST ("Cold"). On the other hand, the winds from the southeast and northwest are creating a small high-pressure area, where the warm air from above the sea influences the SST ("Warm"). In the third row, we see how a high-pressure area (the wind goes east and west creating a suction force downwards) strongly cools the SST ("La Niña"). In comparison to that, the low-pressure area (the wind comes from west and east creating a pressure upwards) strongly heats the SST ("El Niño").

In [3], the authors considered the same problem with the difference that they clustered the conditional probabilities of the high dimensional data of the SST given the high dimensional data of the ZWS. They used a neural network to learn the conditional probabilities first and clustered them afterwards. This procedure is not only more complex, but in performed experiments also approximately 30 times slower than our approach.

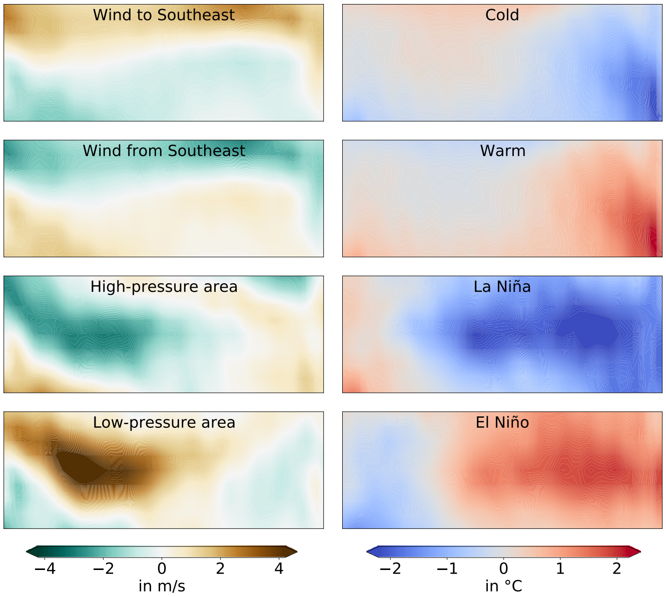

Fig. 7. El Niño macro-variables discovered by CaFe DBSCAN. For each observational class, the average difference from the mean of the data set is shown. On the left side: The macro-variables of the wind zones. On the right side: The respective temperature zones.

IX. Conclusion

In this work, we have introduced the idea of density-based clustering of conditional probabilities in the CFL setting and presented our novel algorithm CaFe DBSCAN, which is the first single-step clustering algorithm for CFL. We showed in various experiments that our algorithm outperforms the previous state-of-the-art approach for CFL, which consists of two steps, estimating the conditional probabilities of effect states and clustering them with k-means, in terms of clustering quality, speed, and robustness to noise. Additionally, we showed that it is not enough to just use other clustering algorithms, like DBSCAN, on the learned conditional probabilities of the classifiers, but that a new approach is needed and justified. For future work, it would be interesting to consider the variants of DBSCAN to improve our algorithm, e.g., for high-dimensional data, varying densities, or automatic estimation of ϵ. Another research direction for CFL in general and CaFe DBSCAN in particular, would be interpretable machine learning, e.g., to automatically find concepts (macro-variables) that then can be used in *concept activation vectors* [24].

References

[1] S. Kato, H. Kaplan, T. Schrödel, S. Skora, T. Lindsay, E. Yemini, S. Lockery, and M. Zimmer, "Global brain dynamics embed the motor command sequence of caenorhabditis elegans," *Cell*, vol. 163, no. 3, p. 656–669, Oct 2015.

[2] A. Fire and S.-C. Zhu, "Using causal induction in humans to learn and infer causality from video," in *Annual Meeting of the Cognitive Science Society*, vol. 35, 2013.

[3] K. Chalupka, T. Bischoff, P. Perona, and F. Eberhardt, "Unsupervised discovery of el niño using causal feature learning on microlevel climate data," in *Conference on Uncertainty in Artificial Intelligence*, ser. UAI '16. AUAI Press, 2016, p. 72–81.

[4] K. Chalupka, P. Perona, and F. Eberhardt, "Visual causal feature learning," in *Conference on Uncertainty in Artificial Intelligence*, ser. UAI '15. AUAI Press, 2015, p. 181–190.

[5] H.-P. Kriegel, P. Kröger, J. Sander, and A. Zimek, "Density-based clustering," *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, vol. 1, no. 3, p. 231–240, May 2011.

[6] M. Ester, H.-P. Kriegel, J. Sander, and X. Xu, "A density-based algorithm for discovering clusters in large spatial databases with noise," in *International Conference on Knowledge Discovery and Data Mining*, ser. KDD '96. AAAI Press, 1996, p. 226–231.

[7] C. Bohm, K. Kailing, H.-P. K. Kriegel, and P. Kröger, "Density connected clustering with local subspace preferences," in *IEEE International Conference on Data Mining*, ser. ICDM '04. IEEE Computer Society, Nov 2004, p. 27–34.

[8] P. Kröger, H.-P. Kriegel, and K. Kailing, "Density-connected subspace clustering for high-dimensional data," in *SIAM International Conference on Data Mining*, ser. SDM '04, vol. 246–257. SIAM, Apr 2004.

[9] H.-P. Kriegel, P. Kröger, and A. Zimek, "Clustering high-dimensional data: A survey on subspace clustering, pattern-based clustering, and correlation clustering," *ACM Transactions on Knowledge Discovery from Data*, vol. 3, no. 1, p. 1–58, Mar 2009.

[10] C. Böhm, R. Noll, C. Plant, and B. Wackersreuther, "Density-based clustering using graphics processors," in *ACM Conference on Information and Knowledge Management*, ser. CIKM '09. Association for Computing Machinery, 2009, p. 661–670.

[11] L. Qian, C. Plant, and C. Böhm, "Density-based clustering for adaptive density variation," in *IEEE International Conference on Data Mining*, 2021, p. 1282–1287.

[12] R. J. G. B. Campello, D. Moulavi, and J. Sander, "Density-based clustering based on hierarchical density estimates," in *Advances in Knowledge Discovery and Data Mining*, J. Pei, V. S. Tseng, L. Cao, H. Motoda, and G. Xu, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2013, pp. 160–172.

[13] E. Bair, "Semi-supervised clustering methods," *Wiley interdisciplinary reviews. Computational statistics*, vol. 5, no. 5, p. 349–361, 2013.

[14] A. P. Dempster, N. M. Laird, and D. B. Rubin, "Maximum likelihood from incomplete data via the em algorithm," *Journal of the Royal Statistical Society, Series B*, vol. 39, no. 1, p. 1–38, 1977.

[15] J.-H. Chen and W.-L. Hung, "An automatic clustering algorithm for probability density functions," *Journal of Statistical Computation and Simulation*, vol. 85, no. 15, p. 3047–3063, 2015.

[16] T. Vo-Van and T. Pham-Gia, "Clustering probability distributions," *Journal of Applied Statistics*, vol. 37, no. 11, p. 1891–1910, Nov 2010.

[17] J. Pearl, *Causality: Models, reasoning and inference*, 2nd ed. Cambridge University Press, 2009.

[18] B. Schölkopf, F. Locatello, S. Bauer, N. R. Ke, N. Kalchbrenner, A. Goyal, and Y. Bengio, "Towards causal representation learning," *Computing Research Repository*, vol. 109, no. 5, p. 612–634, 2021.

[19] K. Chalupka, F. Eberhardt, and P. Perona, "Causal feature learning: An overview," *Behaviormetrika*, vol. 44, no. 1, p. 137–164, 2017.

[20] K. Muandet, K. Fukumizu, B. Sriperumbudur, and B. Schölkopf, *Kernel mean embedding of distributions: A review and beyond*. now, Jan 2017.

[21] D. A. Levin, Y. Peres, and E. L. Wilmer, *Markov chains and mixing times*, 2nd ed. American Mathematical Society, 2017.

[22] N. X. Vinh, J. Epps, and J. Bailey, "Information theoretic measures for clusterings comparison: Variants, properties, normalization and correction for chance," *Journal of Machine Learning Research*, vol. 11, p. 2837–2854, Dec 2010.

[23] L. Hubert and P. Arabie, "Comparing partitions," *Journal of Classification*, vol. 2, no. 1, p. 193–218, Dec 1985.

[24] B. Kim, M. Wattenberg, J. Gilmer, C. J. Cai, J. Wexler, F. B. Viégas, and R. Sayres, "Interpretability beyond feature attribution: Quantitative testing with concept activation vectors (tcav)," in *International Conference on Machine Learning*, ser. ICML '18, vol. 80. PMLR, 2018, p. 2673–2682.

[25] M. Ankerst, M. M. Breunig, H.-P. Kriegel, and J. Sander, "Optics: Ordering points to identify the clustering structure," in *ACM SIGMOD International Conference on Management of Data*, ser. SIGMOD '99, vol. 28. ACM Press, Jun 1999, p. 49–60.

[26] Y. Yang and H. Wang, "Multi-view clustering: A survey," *Big Data Mining and Analytics*, vol. 1, no. 2, p. 83–107, Jun 2018.

[27] K. Chalupka, P. Perona, and F. Eberhardt, "Multi-level cause-effect systems," *arXiv preprint 1512.07942, 2015*, 2015.

Appendix

A. Synthetic Data Set Description

Data set description: We generated three synthetic data sets: Data set 1 (DS1) consists of six well-separated circular clusters, with $n = 3,496$ data points. Data set 2 (DS2) has seven circular clusters with some of them being adjacent and $n = 4,416$. Data set 3 (DS3) consists of seven clusters with three adjacent circles and four adjacent rectangles, with $n = 9,343$. All data sets contain uniform noise and are two dimensional, for plots see Figures 8. In Figure 4 the estimated probability distributions of the clusters and the performances of several clustering algorithms on DS2 are shown.

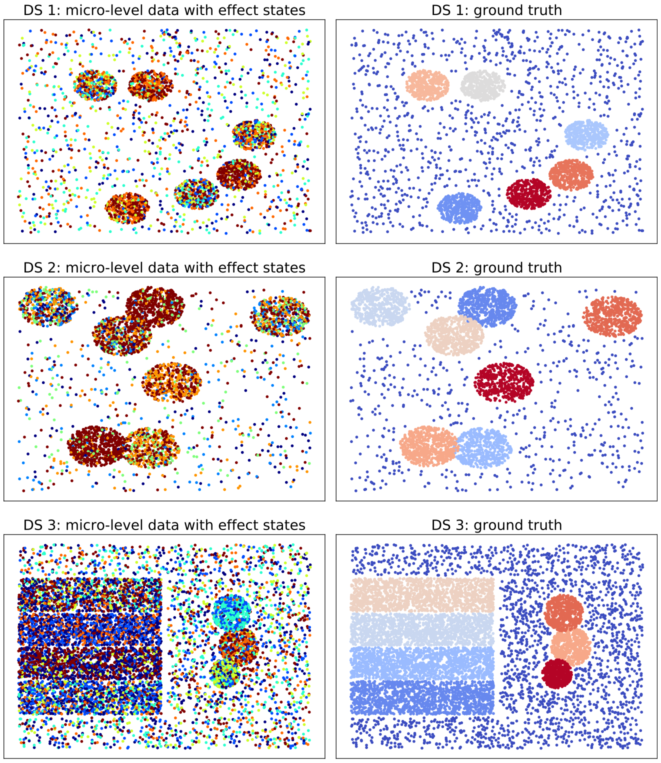

Fig. 8. DS1: Well-separated circular clusters. D2: Adjacent circular clusters. DS3: Adjacent, differently shaped clusters.