GALILEO: A Generalized Low-Entropy Mixture Model

Cetin Savkli, Jeffrey Lin, Philip Graff, and Matthew Kinsey
JHU Applied Physics Laboratory, 11100 Johns Hopkins Road, Laurel, MD 20723, USA

Abstract – *We present a new method of generating mixture models for data with categorical attributes. The keys to this approach are an entropy-based density metric in categorical space and annealing of high-entropy/low-density components from an initial state with many components. Pruning of low-density components using the entropy-based density allows* ***GALILEO*** *to consistently find high-quality clusters and the same optimal number of clusters.* ***GALILEO*** *has shown promising results on a range of test datasets commonly used for categorical clustering benchmarks. We demonstrate that the scaling of* ***GALILEO*** *is linear in the number of records in the dataset, making this method suitable for very large categorical datasets.*

Keywords: clustering, categorical, mixture model, density-based, scalability

1. INTRODUCTION

The growth of large-scale datasets and diversity of data brings an urgency to the development of analytic methods that can handle high volume and dimensionality as well as data that include a mixture of categorical and numerical attributes. Approaching analysis from a probabilistic perspective wherein data is represented as a high-dimensional mixture model provides a transparent representation and a tool that supports common operations such as clustering, anomaly detection, and classification.

While mixture models are a powerful tool, they are often employed for numerical data where mathematical functions, such as multivariate Gaussians, can be used. Each mixture component concisely captures the contribution of a dense region in the high-dimensional space to the distribution as a whole.

In this paper, we present an new algorithm, the GenerALIzed Low-EntrOpy mixture model (**GALILEO**), to extend mixture models to categorical attribute space using a new definition of component density that applies to categorical data. Our concept of categorical density remediates the lack of a natural distance metric in categorical space [1] and contributes to building mixture models with high-density components that represent natural clusters.

The proposed approach involves starting with a high number of initial components and using an annealing process to iteratively remove low-density components. This procedure results in high-density/low-entropy distributions that accurately fit the data. In each step of the process, an expectation-maximization (EM) algorithm is used to generate a fit to the data; pruning of low-density components is then performed using an entropy-based density metric.

We demonstrate that this process generates an optimal solution with respect to the density metric for the `mushroom` dataset as well as producing comparable state-of-the-art results on other datasets commonly used in the literature.

GALILEO is easily parallelizable and scales as $\mathcal{O}(Nk\log(k))$ for N data points to generate a distribution with k mixture components, making it suitable for use on large datasets. Implementation and testing of the algorithm has been done on **SOCRATES**, a scalable analytics platform developed at JHU/APL [2].

This paper is organized as follows: In the next section, we introduce the concept of a generalized density metric that provides the key ingredient of the algorithm. In Section 3, we present a generalization of mixture model that leverages the density metric. Section 4 describes the procedure for determining the optimal number of clusters. Then, we review similar algorithms in Section 5. In Section 6, we present test results on various commonly used datasets.

2. A GENERALIZED DENSITY METRIC

One of the challenges in categorical space is the evaluation of the quality of a mixture component. In numerical space, a natural measure for the quality of a component is provided by the variance of the distribution; high-variance components represent sparsely populated regions of space.

Fig. 1. For numerical distributions density/sparsity of a cluster can be measured concisely with variance, σ^2 (or the covariance matrix for higher dimensional data). For example, it is evident that the density, ρ, $\rho_1 > \rho_2$ when $\sigma_1 < \sigma_2$ for these sample Gaussians.

When a mixture model is initialized with components far from high-density regions, the EM process steers the components towards regions with higher density to eventually find a reasonable solution. In categorical space, the EM process is hindered by a lack of analytic representation that can leverage features of the distribution. Combined with the lack of a component center and a universal distance metric, the EM process can lead to poor results by converging to sub-optimal distributions. To remedy this problem, we follow an approach that starts with a high number of components in the mixture

model and uses a fitness criterion and pruning process to remove low-quality components.

The fitness criterion used in pruning of low-quality components is given by a generalized density metric. Consider, for example, the following one-dimensional distributions:

Fig. 2. Two distributions that span the same domain with same number of data points. A metric similar to standard deviation is needed to describe the density of these distributions.

Whereas a naïve Cartesian density metric, defined as number of particles per unit length, for these two distributions is identical, the distribution on the right is clearly not as "dense" as distribution on the left – i.e., the distribution on the right is more uniform than the one on the left. We therefore propose an effective length d for the axis using the entropy S of the distribution,

$$d \equiv \exp(S), \tag{1}$$

$$S = -\sum_{a=1}^{N} p_a \log p_a. \tag{2}$$

With this definition for the effective length, the length of these distributions is given by $d_L = 3.51$ and $d_R = 4$, therefore the densities ($\rho = N/d$) are $\rho_L = 2.28 > \rho_R = 2$.

As this simple example illustrates, the density definition indeed favors the left distribution. This definition of density (Eq. 1) applies to numerical data as well as categorical data. For example, in a Gaussian distribution, the exponentiation of the differential entropy is proportional to the standard deviation of the distribution [3], i.e., $\sigma = (2\pi e)^{-1/2}\exp(S)$.

It is possible to show that many distributions also have a similar relationship between standard deviation and entropy (e.g. $\sigma = \exp(S-1)$ for an exponential distribution and $\sigma = \exp(S)/(\sqrt{2}e)$ for a Laplace distribution). Extending the entropy-based effective length specification to higher dimensions, the entropy-based effective volume of a hyper-cube in attribute space, with M attributes, can be defined as

$$V = \prod_{m=1}^{M} d_m = \prod_{m=1}^{M} \exp(S_m) = \exp\left(\sum_{m=1}^{M} S_m\right), \tag{3}$$

which leads to a definition of a generalized density in higher dimensions,

$$\rho \equiv \frac{N}{V} = N \exp\left(-\sum_{m=1}^{M} S_m\right). \tag{4}$$

Although it is possible to use the definition of density given in Eq. 4 for both categorical and numerical variables, the numerical subspace requires some care in how entropies are defined. If a multivariate distribution has a high degree of correlation between its variables, treating variables as independent leads to an over-estimation of the effective volume as off-diagonal regions are sparsely populated. Therefore, it is more appropriate to define the volume of the numerical subspace in terms of entropies along the principal axes defined by Principal Component Analysis (PCA) [4]. Looking to the relationship between entropy and standard deviation for guidance, the entropies of numerical subspace along principal components can be estimated using

$$S_q \equiv \log\left(\sqrt{\lambda_q}\right), \tag{5}$$

where λ_q represent eigenvalues of the covariance matrix for numerical attributes.

Some simple examples of the density calculation (Eq. 4) in two dimensions are provided by Fig. 3.

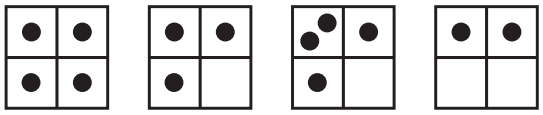

Fig. 3. 2D distribution densities are $\rho_1 = 1$, $\rho_2 = 0.84$, $\rho_3 = 1.3$, $\rho_4 = 1$

Definition of the entropy-based density implies that a uniform distribution leads to a density of 1 independent of the size and shape of the cube for data without duplicates. Furthermore, in this case of data without duplicate points, the density is bounded by 1, a constraint that follows from Shannon's entropy inequality. For a cube of N particles without duplicates, the joint entropy, S, is given by

$$p_a = \frac{1}{N}, \tag{6}$$

$$S = -\sum_{a=1}^{N} p_a \log p_a = \log N, \tag{7}$$

where p_a is the probability of an individual particle.

The entropy of a multivariate distribution follows the inequality

$$S \le \sum_{m=1}^{M} S_m \tag{8a}$$

$$\log N \le \sum_{m=1}^{M} S_m \tag{8b}$$

$$N \le \exp\left(\sum_{m=1}^{M} S_m\right) \tag{8c}$$

$$\rho \le 1. \tag{8d}$$

Having defined a metric through Eqs. 2, 3, and 4 to measure the quality of an individual component, we next discuss how it can be used within the context of a mixture model to generate high-density components in categorical space.

3. Generalized Mixture Model

A mixture model is defined by a superposition of probability distributions for k components,

$$\Pr(x) = \sum_{i=1}^{k} \Pr(x|C_i)\Pr(C_i), \tag{9}$$

where each component distribution, C, is subject to the normalization condition,

$$1 = \sum_{x_j \in C} \Pr(x_j|C), \tag{10}$$

and the components priors determine the relative size of each components,

$$1 = \sum_{i=1}^{k} \Pr(C_i). \tag{11}$$

The individual component distributions can be modeled by any suitable distribution depending on the problem and types of data attributes involved. When the attributes are all categorical, a high-dimensional nonparametric distribution based on a clique tree may be used [5] to estimate the full joint probability. Such a distribution is a good option when data has sub-spaces where attributes are highly correlated. However, since individual component distributions are not required to model the entire space, but only a dense region, a complex structure such as a clique tree for individual components is not necessary. Correlations within dense regions are much less significant and a naïve assumption of attribute independence inside a component is typically sufficient. The fact that a mixture model comprises many components captures the structure of correlations that a clique tree represents. Therefore, the naïve probability for a data point with M attributes to be a member of a component is given by

$$\Pr(x_a|C_i) = \prod_{m=1}^{M} \Pr(x_{am}|C_i). \tag{12}$$

Each component contains a discrete probability distribution for each attribute of the dataset. Numerical attributes may be considered at this time by discretizing them and treating them as categorical. Alternatively, numerical subspaces can be represented using a multivariate distribution as is done in the Gaussian Mixture Model. However, our focus in this paper is the generalized categorical mixture model. The potential benefits of annealing using an entropy-based density metric for numerical data will be considered in future work.

GALILEO starts by initializing the mixture model with a large number of components, $k_{\max}$. The initialization of the components is performed by generating random component "centers" according to the global distribution of the data. Since initial components need a probability distribution (and a single center point does not provide that), we use an equally-weighted average of the global distribution with the randomly generated component center. In other words, each component starts with the probability distribution given by all N data points plus a random center inserted a further N times.

Following creation of the initial components, **GALILEO** will then iteratively:

1) Use expectation-maximization to fit the distribution to data at a given k,
2) Sort the mixture components using the density metric (Eq. 4),
3) Prune the lowest-density components,

until the number of components has been reduced to 1. The EM algorithm in Step 1 evaluates component memberships in a probabilistic manner, assigning each data point fractionally to each component. This fractional assignment is given by the posterior probability of a measurement belonging to a component, which follows from Bayes' theorem as

$$\Pr(C_i|x_a) = \frac{\Pr(x_a|C_i)\Pr(C_i)}{\Pr(x_a)}. \tag{13}$$

An optimal solution for k is then selected using an optimality criterion as described in Section 4.

A detailed description of the steps of the algorithm is provided in Fig. 4.

Require: $k_{\max}$: Maximum component centers, β : step root.
Initialization:
Set all component distributions f_i to global distribution G
1: $f_i = G, i = 1, \ldots, k_{\max}$.
Generate $k_{\max}$ random centers $x_i, i = 1, \ldots, k_{\max}$ and insert each center to a distribution, f_i, N times to define the initial mixture model:
2: $g_k = \{(\alpha_i, f_i)\}$ for $i = 1, \ldots, k_{\max}$ where initially components have equal weight $\alpha_i = 1/k_{\max}$
3: Define $k[0] = 1$, $k[i+1] = k[i] + \beta^i$, $k[i_{\max}] \leq k_{\max}$.
4: $i = i_{max}$, $\hat{k} \leftarrow k[i]$
Annealing:
5: **while** $\hat{k} \geq 1$ **do**
Perform expectation-maximization:
6: $g_k \leftarrow EM(g_k, Data)$
Estimate component density & average density:
7: $\rho_{ki} \leftarrow \rho(f_i)$
8: $\bar{\rho}_k \leftarrow \sum_{i=1}^{k} \alpha_i \rho_{ki}$
9: Use g_k to calculate AIC, BIC
Sort components based on density:
10: $f \rightarrow f_1, f_2, \ldots, f_{\hat{k}}$ where $\rho_{ki} \geq \rho_{k(i+1)}$
Remove lowest density m components:
11: $f \rightarrow f_1, f_2, \ldots, f_{\hat{k}-m}$ and $\alpha \rightarrow \alpha_1, \alpha_2, \ldots, \alpha_{\hat{k}-m}$
Redefine g_k in terms of remaining components:
12: $g_{\hat{k}-m} = \{(\alpha_i, f_i)\}$ for $i = 1, \ldots, (\hat{k} - m)$
13: $i = i - 1$, $\hat{k} \leftarrow k[i]$
14: **end while**
Selection:
15: **return** Best result, $g_{k^\star}$, according to AIC, BIC, or $\bar{\rho}$

Fig. 4. Algorithm to generate mixture model through density based annealing

This method is similar to the *finite mixture model* used by [6], [7], with further details in [8], [9]. However, the introduction of the denisty metric and the procedure for optimizing k make **GALILEO** a unique application of this model.

4. SEARCH FOR OPTIMAL k

There are general rules of thumb about the relationship between the optimal number of components, $k^\star$, and the number of data points, N. However, in general, it is not possible to make a definitive statement about such a relationship. From our experiments, we find that $k_{\max}$ should be picked such that it is at least twice as large as the expected optimal number of components, $k^\star$. This choice gives the annealing process the

opportunity to converge to the optimal solution consistently. Larger values of $k_{\max}$ will not affect the value of $k^\star$, but will take longer to converge due to more steps being required.

In practice, we use the following procedure to step down from $k_{\max}$. By choosing a parameter, $\beta \geq 1$, we then inspect the set of k values that are defined by the relationship,

$$k[i+1] = \lfloor k[i] + \beta^i \rfloor \tag{14}$$

where $k[0] = 1$ and $k[i_{\max}] \leq k_{\max}$. The parameter β determines how finely the optimization of number of components is performed. Using such a rule, the number of possible mixture models inspected scales as $\log(k_{\max})$.

For each value of k, an EM procedure is performed to converge to a solution using available components at the level. Next, the quality of the mixture model solution is measured. Two commonly used metrics for model selection are the Akaike Information Criterion (AIC, [10]) and Bayesian Information Criterion (BIC, [11]). These are given by

$$\begin{aligned} \text{AIC} &= 2\nu - 2\log(L), \\ \text{BIC} &= \log(N)\nu - 2\log(L), \end{aligned}$$

where ν is the degrees of freedom of the model. A detailed description and comparison of these metrics is given by [12], [13]. In addition to the AIC and BIC, we also evaluate the size-weighted average density of the components,

$$\bar{\rho} = \sum_{i=1}^{k} \alpha_i \rho_i. \tag{15}$$

where

$$\alpha_i = \frac{1}{N} \sum_{a=1}^{N} \Pr(C_i | x_a) \tag{16}$$

and $\Pr(C_i|x_a)$ is given by Eqn. (13) with $\Pr(C_i) = \alpha_i$ and initially $\alpha_i = 1/k_{\max}$ as we start with $k_{\max}$ components. Whereas one would seek to minimize the AIC or BIC, we wish to maximize the density, $\bar{\rho}$, of the mixture model. The density measure, $\bar{\rho}$ has the benefit over the AIC and BIC in that it scales only with the number of clusters, number of attributes, and cardinality of attributes – there is no dependence on the dataset size so it will be simpler and faster to compute than likelihood-based metrics. In later examples, we compare using each of these three criteria to determine $k^\star$, finding that they agree in certain cases. The choice of which to use may be data-dependent and is up to the user to choose.

5. Relevant Literature

To date, most of the work in the realm of clustering algorithms has been focused on the realm of numerical data [14], [15]. However, there has been some work done in regard to the clustering of categorical and mixed data. In this respect, there are a handful of algorithms that represent the state of the art, namely **ROCK** and **COOLCAT**. **DBSCAN** is a numerical clustering algorithm that uses a density notion similar to that of **GALILEO**. In this section we will briefly review each of these algorithms. In [16], the authors present a review of the clustering literature and propose a different entropy-based method for determining optimal cluserting of mixed data. Due to space constraints, further comparisons with other algorithms are deferred to future work.

5.1. ROCK

ROCK [17] is often used a benchmark for the quality of a categorical clustering algorithm. **ROCK** first computes the Jaccard coefficient between all pairs of data points. By then applying a threshold, θ, to these coefficients, **ROCK** assigns each data point a list of "neighbors" and computes the matrix L_{ab}, the number of common neighbors shared by points a and b. **ROCK** then agglomeratively finds k clusters that maximize the criterion function,

$$E_l = \sum_{i=1}^{k} n_i \sum_{a,b \in C_i} \frac{L(a,b)}{n_i^{1+2f(\theta)}}, \tag{17}$$

where $f(\theta)$ is a cluster fitness function chosen by the user that depends on the data and type of cluster desired. While **ROCK** has been shown to produce high-quality results, it suffers from a poor worst-case complexity of $\mathcal{O}(N^2 \log N)$. Additionally, it requires the user to tune the algorithm to the data through the choice of both the thresholding parameter θ and the fitness function $f(\theta)$.

5.2. COOLCAT

COOLCAT [18] uses the notion of entropy as the means to cluster the data. The algorithm begins by selecting k samples that collectively have the highest entropy. These k points will be the initial k cluster centers. **COOLCAT** then proceeds by adding each sample in the dataset to the cluster that will result in the smallest increase in entropy.

As a result of this sequential process, **COOLCAT** is sensitive to the ordering of the data. In order to limit this sensitivity, the data is processed in batches and a re-clustering procedure is performed after each batch. This procedure takes some fraction of the most poorly-fit data points and reassigns them to the clusters.

Even with the re-clustering procedure, **COOLCAT** results are strongly dependent on the ordering of the data. Moreover, the process of choosing the initial k clusters is $\mathcal{O}(S^2)$ where S is some representative sample of N, limiting **COOLCAT**'s effectiveness for large datasets.

5.3. DBSCAN

DBSCAN [19], much like **GALILEO**, uses a notion of density in order to find clusters of points. Unlike the methods covered to this point, **DBSCAN** is strictly for use on numerical data, requiring a distance metric to calculate distances between points in the data. The authors have even extended **DBSCAN** to cluster spatially extended objects like polygons [20]. The algorithm is able to automatically find the number of clusters as well as find clusters of arbitrary shape.

6. RESULTS

In this section we will present the results of **GALILEO**'s clustering on a few publicly available datasets. **GALILEO** clusters by assigning each data point to its most probable component in the $k^\star$ components in the optimal mixture model, $g_{k^\star}$. We first describe the datasets to be used and then demonstrate **GALILEO**'s performance, including some comparisons to other algorithms mentioned previously.

6.1. EXPERIMENTAL DATASETS

6.1.1) Congressional Votes: The Congressional votes dataset[1], `votes`, is from the UCI Machine Learning Repository [21]. This dataset consists of the 1984 voting history of each member of Congress with respect to 16 different issues. Each member of Congress is assigned 16 binary (yes/no) vote attributes as well as a classification label (Republican or Democrat). The dataset contains 267 Democrats and 168 Republicans. The classification label was ignored for the purpose of clustering so that it could be used as an independent measure of clustering results.

6.1.2) Mushrooms: We have also benchmarked our code using the `mushroom` dataset[2] from the UCI Repository [21]. This dataset contains the physical properties of 8124 gilled mushrooms from 23 species in the Agaricus and Lepiota family, as well as their edibility. In addition to the binary edibility, there are 22 other categorical attributes, each admitting up to twelve possible values. These attributes describe various properties such as color, odor, and shape. All attributes were used for clustering in order to be consistent with the procedure of the **ROCK** paper [17].

6.1.3) Soybean: Another standard categorical dataset, `soybean`[3], consists of 19 classes each with 35 categorical attributes. This dataset categorizes the properties of various types of diseases in soybeans [22]. It was also obtained from the UCI Machine Learning Repository [21].

6.1.4) Zoo: The `zoo` dataset[4] consists of 17 different attributes related to each of 101 species of animal. These attributes represent, for example, how many legs an animal has or if it has feathers. This dataset was also obtained from the UCI Machine Learning Repository [21].

6.1.5) Synthetic: In order to test data of various sizes, we used `datgen` [23] to generate categorical datasets of arbitrary size. These datasets were generated using a set of rules to cluster records in the attribute space. In our tests, each record had 10 attributes with 20 possible values constrained by one of five rules.

6.2. EXPERIMENTAL RESULTS

In order to show in detail how the algorithm works, we use the `mushroom` dataset (Section 6.1.2). Fig. 5 shows the AIC, BIC, and density curves produced by **GALILEO** when clustering this dataset. All three metrics agree that $k^\star = 23$, although there are visible differences in how clear this selection is. It is worth noting that $\bar{\rho} = 1$ for $k^\star = 23$. Recall that for data that has no duplicates, the theoretical upper limit on the density of a cluster according to the inequality given by Eq. 8d is 1. Interestingly, this optimal solution corresponds to clusters that contain only all edible or all poisonous mushrooms. Furthermore, the 23 clusters corresponds exactly with the number of species of mushrooms represented in the dataset (unfortunately, the species identification is not in the dataset so we are unable to perform a direct comparison). Reaching the maximum average density of 1 in a generic clustering problem when data is categorical is clearly not always achievable.

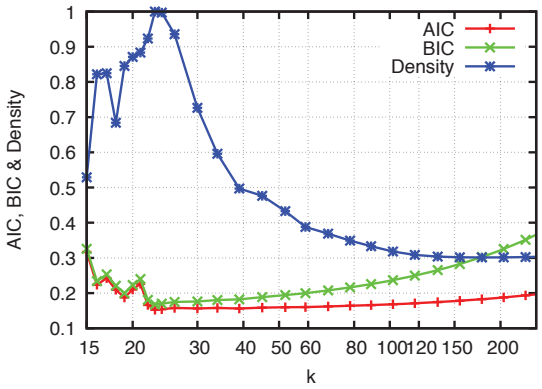

Fig. 5. For the `mushrooms` dataset, the optimum of all three metrics (AIC, BIC, density) coincide at $k^\star = 23$ which corresponds to $\bar{\rho} = 1$.

The role of density in obtaining this result can be understood by changing the pruning criteria from our entropy-based density to a naïve Cartesian density. Fig. 6 demonstrates that when a simplistic Cartesian density is used it is not possible to reach an optimal result, instead finding that $k^\star = 30$. Whereas the results are comparable for high values of k, the Cartesian density is less able to determine which clusters are best to prune as the number of clusters begins to approach $k^\star$.

Another consideration in the execution of the algorithm is the choice of $k_{\max}$. The results shown in Fig. 7 illustrate that as long as $k_{\max} \geq 40$, the annealing process converges to the same optimal result, $k^\star = 23$. If $k_{\max}$ is set lower, the annealing process does not have sufficient time to converge to the optimal solution. We observe a similar behavior on other datasets tested and in general find that using a starting point that has at least twice the expected number of clusters is a good rule of thumb to reach an optimal solution.

Results represented by Figs. 6 and 7 show that both annealing and using an entropy-based density metric contribute to achieving an optimal result.

6.3. COMPARISON TO OTHER ALGORITHMS

We now compare **GALILEO**'s results to that of other commonly-used categorical clustering algorithms (Sec. 5). In evaluating the quality of our clustering results we will make use of the Category Utility function [24], [25], [26], $\bar{CU}$. This function provides a measure of the predictive advantage gained with knowledge of the clustering relative to without that knowledge.

[1] https://archive.ics.uci.edu/ml/datasets/Congressional+Voting+Records
[2] https://archive.ics.uci.edu/ml/datasets/Mushroom
[3] https://archive.ics.uci.edu/ml/datasets/Soybean+(Large)
[4] https://archive.ics.uci.edu/ml/datasets/Zoo

TABLE I
RESULTS FOR THE VOTES DATASET FOR **GALILEO**, **ROCK**, AND A TRADITIONAL HIERARCHICAL METHOD.

	GALILEO			**ROCK**			Trad. Hier. Method		
Cluster	N	Rep.	Dem.	N	Rep.	Dem	N	Rep.	Dem
1	230	9	221	206	5	201	226	11	215
2	205	159	46	166	144	22	209	157	52

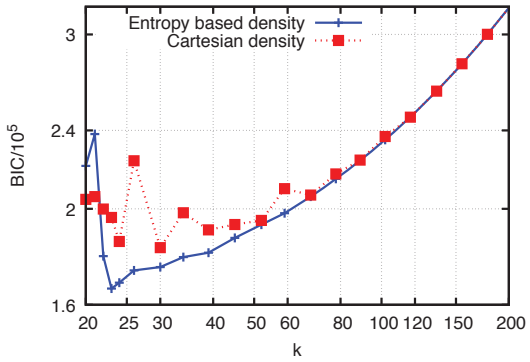

Fig. 6. Entropy-based density versus Cartesian density for the mushroom dataset. Use of Cartesian density as a metric for pruning weak clusters does not produce an optimal solution. The entropy-based pruning criterion leads to a solution where $\bar{\rho} = 1$.

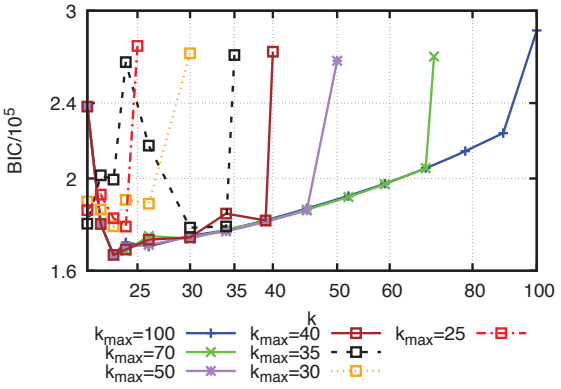

Fig. 7. Dependence of reaching global minimum in initial cluster size, $k_{\max}$, for the mushroom dataset. Runs where $k_{\max} < 40$ do not reach the optimal solution of $k^\star = 23$ clusters with $\bar{\rho} = 1$.

For a comparison of the clustering results of **GALILEO** to that of **ROCK**, see Tables I and II. For the votes dataset, we find our results to be consistent with the clusters found using **ROCK**. Since **ROCK** implements an outlier removal scheme, they report fewer total members for each cluster than we do. While this comparison is not exact, we have, however, demonstrated an improvement in the clustering when compared to the traditional centroid-based hierarchical clustering algorithm [27], [28] that **ROCK** used as a baseline.

On the mushroom dataset, **GALILEO** finds roughly the same clusters as **ROCK**, with the only exception being that **GALILEO** naturally converges to 23 clusters as opposed to 21 with **ROCK** [17]. These extra clusters result from splitting two of the **ROCK** clusters, including the one with mixed edibility. **GALILEO** identifies no clusters with mixing in the edibility attribute.

TABLE II
CLUSTERS FROM FOUND IN THE MUSHROOM DATASET BY **GALILEO** AND **ROCK**. CLUSTERS THAT **GALILEO** IDENTIFIED AS TWO WHERE **ROCK** HAD ONE ARE MARKED WITH PAIRED ANNOTATIONS.

	GALILEO		**ROCK**	
Cluster	Edib.	Pois.	Edib.	Pois.
1	96	0	96	0
2	0	256	0	256
3†	512	0	704	0
4	96	0	96	0
5	768	0	768	0
6	0	192	0	192
7	1728	0	1728	0
8	0	32	0	32
9	0	1296	0	1296
10	0	8	0	8
11	48	0	48	0
12	48	0	48	0
13	0	288	0	288
14	192	0	192	0
15‡	0	72	32	72
16	0	1728	0	1728
17	288	0	288	0
18	0	8	0	8
19	192	0	192	0
20	16	0	16	0
21	0	36	0	36
22‡	32	0	0	0
23†	192	0	0	0

Finally, in Table III we report our results for the various datasets from the UCI Machine Learning repository. It is noteworthy that these values show comparable results for $\bar{CU}$ to the results of **COOLCAT** (Sec. 5.2), obtained using the **coolcat-r** package[5] (except for the mushroom dataset – marked with * –, which we obtain from [18] and normalize by an assumed 21 clusters).

TABLE III
SUMMARY OF RESULTS FOR UCI DATASETS.

	k	CU_G	CU_C	$\bar{\rho}$	S
mushroom	23	0.3266	0.3393*	1	0.2893
votes	2	1.4686	1.4674	0.01627	0.5988
soybean	7	1.0912	0.9362	0.02327	0.5151
zoo	9	0.5711	0.5970	1.4595	0.1882

6.4. SCALING RESULTS

In order to test the computational time complexity of our algorithm, we used synthetic categoric data (Section 6.1.5) of various sizes, built using the same rules. For each dataset, **GALILEO** was able to find the known true $k^\star$ and accurately cluster the data points. Figure 8 shows the timing results of this test; multiple runs were performed for each value of N yielding highly consistent timings. Once the number of records reaches a certain threshold, our scaling is very close to the theoretical time complexity $\mathcal{O}(N)$, for a fixed k.

[5]https://github.com/clbustos/coolcat-r

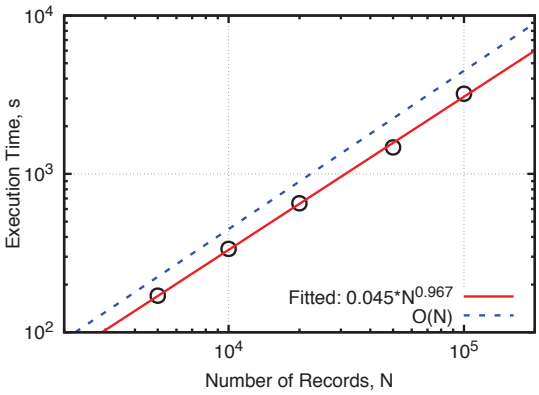

Fig. 8. Scaling of execution speed with respect to number of records for synthetic data. Fitting was performed for $N \geq 10^3$ to minimize the effect of overhead present at low N.

7. Conclusion

In this paper, we have presented a new method of generating mixture models in linear time for data with categorical attributes. The keys to this approach are the entropy-based density metric in categorical space and the annealing of high-entropy/low-density components from an initial state with many components. Pruning of low-density components using the entropy-based density allows **GALILEO** to consistently find high-quality clusters and the same optimal number of clusters. **GALILEO** has shown promising results on a range of test datasets commonly used for categorical clustering benchmarks. In particular, we have shown **GALILEO**'s annealing approach and density-based pruning consistently finds the optimal clustering (based on our concept of density) on the `mushroom` dataset. Perhaps more importantly, we have demonstrated that the scaling of **GALILEO** is linear $\mathcal{O}(Nk\log(k))$ in the number of records in the dataset, making this method suitable for very large categorical datasets. **GALILEO** can be naturally extended to include numerical attributes and datasets with mixed attribute types. In the future, we will expand the applications of this method for use on datasets consisting of mixed attributes and compare **GALILEO**'s performance on numerical data to traditional numerical clustering algorithms.

Acknowledgments

This work was supported by internal research and development funding provided by JHU/APL.

References

[1] S. Boriah, V. Chandola, and V. Kumar, "Similarity measures for categorical data: A comparative evaluation," in *Proceedings of the 2008 SIAM International Conference on Data Mining*. SIAM, 2008, pp. 243–254.

[2] C. Savkli, R. Carr, M. Chapman, B. Chee, D. Minch, and C. Savkli, "Socrates-a system for scalable graph analytics." in *HPEC*, 2014, pp. 1–6.

[3] M. F. Huber, T. Bailey, H. Durrant-Whyte, and U. D. Hanebeck, "On entropy approximation for gaussian mixture random vectors," in *Multisensor Fusion and Integration for Intelligent Systems, 2008. MFI 2008. IEEE International Conference on*. IEEE, 2008, pp. 181–188.

[4] K. Pearson, "Principal components analysis," *The London, Edinburgh and Dublin Philosophical Magazine and Journal*, vol. 6, no. 2, p. 566, 1901.

[5] C. Savkli, J. R. Carr, P. Graff, and L. Kennell, "Bayesian learning of clique tree structure," in *Proceedings of the International Conference on Data Mining (DMIN)*. The Steering Committee of The World Congress in Computer Science, Computer Engineering and Applied Computing (WorldComp), 2016, p. 201.

[6] C. Meek, B. Thiesson, and D. Heckerman, "The learning-curve sampling method applied to model-based clustering," *Journal of Machine Learning Research*, vol. 2, no. Feb, pp. 397–418, 2002.

[7] P. Cheeseman, M. Self, J. Kelly, W. Taylor, D. Freeman, and J. C. Stutz, "Bayesian classification." in *AAAI*, vol. 88, 1988, pp. 607–611.

[8] D. M. Titterington, A. F. Smith, and U. E. Makov, *Statistical analysis of finite mixture distributions*. John Wiley and Sons, 1985.

[9] G. J. McLachlan and K. E. Basford, *Mixture models: Inference and applications to clustering*. Marcel Dekker, 1988, vol. 84.

[10] H. Akaike, "Information theory and an extension of the maximum likelihood principle," in *Selected Papers of Hirotugu Akaike*. Springer, 1998, pp. 199–213.

[11] G. Schwarz, "Estimating the dimension of a model," *Ann. Statist.*, vol. 6, no. 2, pp. 461–464, 1978.

[12] S. I. Vrieze, "Model selection and psychological theory: a discussion of the differences between the akaike information criterion (aic) and the bayesian information criterion (bic)." *Psychological methods*, vol. 17, no. 2, p. 228, 2012.

[13] K. Aho, D. Derryberry, and T. Peterson, "Model selection for ecologists: the worldviews of aic and bic," *Ecology*, vol. 95, no. 3, pp. 631–636, 2014. [Online]. Available: http://dx.doi.org/10.1890/13-1452.1

[14] C. C. Aggarwal and C. K. Reddy, *Data clustering: algorithms and applications*. Chapman and Hall/CRC, 2013.

[15] A. Fahad, N. Alshatri, Z. Tari, A. Alamri, I. Khalil, A. Y. Zomaya, S. Foufou, and A. Bouras, "A survey of clustering algorithms for big data: Taxonomy and empirical analysis," *IEEE transactions on emerging topics in computing*, vol. 2, no. 3, pp. 267–279, 2014.

[16] J. Liang, X. Zhao, D. Li, F. Cao, and C. Dang, "Determining the number of clusters using information entropy for mixed data," *Pattern Recogn.*, vol. 45, no. 6, pp. 2251–2265, Jun. 2012. [Online]. Available: http://dx.doi.org/10.1016/j.patcog.2011.12.017

[17] S. Guha, R. Rastogi, and K. Shim, "Rock: A robust clustering algorithm for categorical attributes," *Information systems*, vol. 25, no. 5, pp. 345–366, 2000.

[18] D. Barbará, Y. Li, and J. Couto, "Coolcat: an entropy-based algorithm for categorical clustering," in *Proceedings of the eleventh international conference on Information and knowledge management*. ACM, 2002, pp. 582–589.

[19] M. Ester, H.-P. Kriegel, J. Sander, X. Xu *et al.*, "A density-based algorithm for discovering clusters in large spatial databases with noise." in *Kdd*, vol. 96, no. 34, 1996, pp. 226–231.

[20] J. Sander, M. Ester, H.-P. Kriegel, and X. Xu, "Density-based clustering in spatial databases: The algorithm gdbscan and its applications," *Data mining and knowledge discovery*, vol. 2, no. 2, pp. 169–194, 1998.

[21] M. Lichman, "UCI machine learning repository," 2013. [Online]. Available: http://archive.ics.uci.edu/ml

[22] R. S. Michalski and R. L. Chilausky, "Learning by being told and learning from examples: An experimental comparison of the two methods of knowledge acquisition in the context of developing an expert system for soybean disease diagnosis," *International Journal of Policy Analysis and Information Systems*, vol. 4, no. 2, pp. 125–161, 1980.

[23] G. Melli, "The `datgen` dataset generator, http://www.datasetgenerator.com," 1999.

[24] M. A. Gluck and J. E. Corter, "Information Uncertainty and the Utility of Categories," in *Proceedings of the Seventh Annual Conference of Cognitive Science Society*, 1985, pp. 283–287.

[25] J. E. Corter and M. A. Gluck, "Explaining basic categories: Feature predictability and information." *Psychological Bulletin*, vol. 111, no. 2, p. 291, 1992.

[26] D. H. Fisher, "Knowledge acquisition via incremental conceptual clustering," *Machine Learning*, vol. 2, no. 2, pp. 139–172, 1987. [Online]. Available: http://dx.doi.org/10.1007/BF00114265

[27] R. O. Duda, P. E. Hart, D. G. Stork *et al.*, *Pattern classification*. Wiley New York, 1973, vol. 2.

[28] A. K. Jain and R. C. Dubes, *Algorithms for clustering data*. Prentice-Hall, Inc., 1988.