Recurrent Clustering for Unsupervised Feature Extraction with Application to Sequence Detection

Steven R. Young, Itamar Arel
The University of Tennessee
Machine Intelligence Lab
{syoung22, itamar}@eecs.utk.edu

***Abstract*—In many unsupervised learning applications both spatial and temporal regularities in the data need to be represented. Traditional clustering algorithms, which are commonly employed by unsupervised learning engines, lack the ability to naturally capture temporal dependencies. In supervised learning methods, temporal features are often learned through the use of a feedback (or recurrent) signal. Drawing inspiration from the Elman recurrent neural network, we introduce a winner-take-all based recurrent clustering algorithm that is able to identify temporal regularities in an unsupervised manner. We explore the potential pitfalls that result from adding feedback to an incremental clustering algorithm and apply the proposed technique to several time series inference problems in the context of semi-supervised learning. The results clearly indicate that the framework can be broadly applied with particular relevance to scalable deep machine learning architectures.**

***Index Terms*—recurrent clustering; time series analysis; spatiotemporal features; semi-supervised learning**

I. Introduction

Many machine learning applications require unsupervised learning algorithms that capture both spatial and temporal features in data. Existing clustering algorithms lack the means to capture temporal dependencies in a natural, data-driven manner. Hence, the mainstream approach for capturing temporal dependencies in existing clustering algorithms is to concatenate observations from several time-steps. Acceptable results using this method require prior (expert) knowledge of the problem in order to choose the correct time-steps to combine. This problem has been extensively addressed in supervised learning methods, such as recurrent neural networks. In these methods, a feedback signal is employed to learn temporal regularities. An example of such system is the Elman recurrent neural network, in which the hidden node activations are latched and form part of the input at the subsequent time slot [1].

By employing this idea of latching state information to augment the input during the next time step, we propose a clustering algorithm that captures temporal features in a natural, data-driven way. Such an algorithm requires an incremental clustering algorithm, since each state is dependent on the previous state, and a method for creating state information. Thus, in addition to learning centroids as a compact representation of dense regions of the input space, the algorithm must maintain a belief state that is a result of encoding each new observation relative to the current centroids. This belief state is latched and appended to the subsequent input presented to the clustering algorithm. As a result, the centroids inherently represent spatiotemporal patterns found in the data, rather than simple spatial templates. In the remainder of this paper, we discuss the hazards of adding feedback to an incremental clustering algorithm, detail the proposed recurrent clustering algorithm and evaluate it on a time series inference problem.

II. Recurrent Clustering Algorithm

We next present a recurrent incremental clustering algorithm which forms centroids in a space comprising a concatenation of the current observation and the previous belief state. We refer to this combination as the input vector. The belief state is a measure of the likelihood that the current input vector belongs to each of the centroids. Recurrent clustering requires an incremental algorithm since each belief state is dependent on the preceding belief state.

A. Incremental Clustering

The incremental clustering algorithm used in this work is based on a winner-take-all scheme involving calculation of a mean μ and variance σ^2 for each centroid in each dimension. The update rules for the mean and variance are given by

$$\mu_x = \alpha\mu_x + (1-\alpha)(o-\mu_x) \tag{1}$$

$$\sigma_x^2 = \beta\sigma_x^2 + (1-\beta)\left|(o-\mu_x)^2 - \sigma_x^2\right| \tag{2}$$

where o is the current observation, x is the centroid being updated, and $0 < \alpha, \beta < 1$ are learning rates. The winning centroid is chosen by weighting the distance between the current input vector and each centroid by the starvation trace ψ , as shown in (3), first introduced in [2]. The starvation trace decays with each iteration in which a centroid is not selected for an update, and thus shrinks the apparent distance to centroids that have been updated frequently. The update rule for the starvation trace for each centroid c is given in (4).

$$x = \text{argmin}_{c \in C}\left[\psi_c \left\|o-\mu_c\right\|\right] \tag{3}$$

$$\psi_c = \gamma\phi_c + (1-\gamma)\mathbb{1}_{x=c} \tag{4}$$

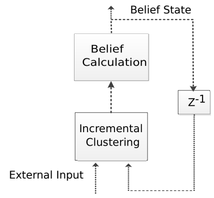

Fig. 1. Recurrent Clustering: external input is augmented by the previous belief state forming the input to an incremental clustering algorithm.

B. Belief State Formulation

After a selected centroid is updated, the belief state b is obtained using the normalized Euclidean distance between the input vector and each centroid c, such that

$$n_c = \sum_{i=1}^{d} \frac{(o_i - \mu_{c,i})^2}{\sigma_{c,i}^2} \tag{5}$$

$$b_c = \frac{n_c^{-1}}{\sum_{c' \in C} n_{c'}^{-1}} \tag{6}$$

where d is the number of dimensions of the observation. This formulation is used as it does not tend to form beliefs that only indicate which centroid is closest. Rather, it takes the variance in each dimension into account. This formulation tends to form richer association measure between an input vector and the centroids.

C. Feedback Construct

The belief state is then used as part of the input to the clustering algorithm forming the recurrent clustering system, as illustrated in Figure 1. The feedback loop that recurrent signal introduces into the incremental clustering algorithm presents many design hazards. The most difficult challenge is choosing a method for selecting the centroid to be updated. The approach taken must strike a balance between the importance of the current observation (or spatial features) and the previous state (or temporal features). If the centroid variances were used for selecting the winning centroid, as they are when forming the belief state, the resulting system would form overly confident beliefs that are based only on temporal attributes. The resulting system can only act as a counter since beliefs are not based upon the spatial input. As such, the selection method used considers only the Euclidean distance between the input vector and the centroids. It may be necessary to use a normalized Euclidean distance with a constant normalization vector for selection if the variance of the observations is much larger or smaller than the variance of the beliefs, but this has not been observed in any problems explored here.

III. Simulation Results

In order to evaluate the temporal capacity of the algorithm, it was tested on a sequence detection task with the goal of demonstrating the ability of the recurrent clustering algorithm to capture long term temporal dependencies. N binary vectors of length D were chosen as sequences of interest, with differences between each of the sequences being at the beginning. These sequences are then provided to the algorithm at random, while the belief state that is generated by the clustering algorithm is delivered to a feed-forward neural network for classification. Results shown in Figure 2 indicate that capturing longer-term temporal dependencies with this method is increasingly difficult as the span of the dependencies increases, much like the case in recurrent neural networks [3].

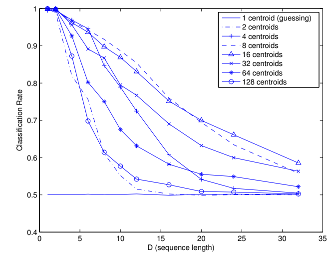

Fig. 2. Classification accuracy for a two-class (two sequences of interest) problem as a function of the number of centroids and various sequence lengths. This figure demonstrates how classification performance decreases as the length of the sequence of interest (D) increases and how different results are acheived for different numbers of centroids. Results indicate that an optimal number of centroids for longer sequences is 16.

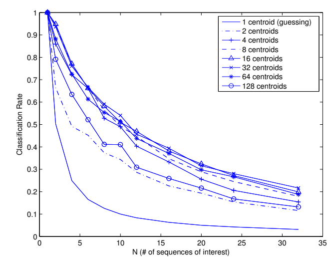

Fig. 3. Classification accuracy with various number of centroids , a fixed sequence length at $D = 8$ and varying number of sequences of interest. This figure demonstrates how classification performance decreases as the number of classes (N) increases and how different results are acheived for different numbers of centroids. Results indicate that an optimal number of centroids for this problem is 32.

IV. Conclusions

The recurrent clustering algorithm proposed here operates in a data-driven manner and is an unsupervised analogue to the Elman recurrent neural network. Experimental results demonstrate that it is able to characterize temporal attributes of arbitrary length and generate feature vectors that can be used by a classifier to perform sequence detection.

References

[1] J. L. Elman, "Finding structure in time," *Cognitive Science*, vol. 14, no. 2, pp. 179–211, 1990.

[2] S. Young, I. Arel, T. Karnowski, and D. Rose, "A fast and stable incremental clustering algorithm," in *to appear in 7th International Conference on Information Technology*, April 2010.

[3] Y. Bengio, P. Simard, and P. Frasconi, "Learning long-term dependencies with gradient descent is difficult," *IEEE Transactions on Neural Networks*, vol. 5, no. 2, pp. 157–166, 1994.