A Pattern-Hierarchy Classifier for Reduced Teaching

KIERAN GREER,
Distributed Computing Systems, Belfast, UK.
http://distributedcomputingsystems.co.uk

Abstract – This paper describes a design that can be used for Explainable AI and also autonomous clustering. A 2-level structure is proposed, with a lower level ensemble of patterns created by self-organisation, linking with an upper knowledge-based level that can be hierarchical. This provides a transition from mixed ensemble masses to specific categories. Clusters are learned in an unsupervised manner, using a new self-organising algorithm, but can then be split again when the upper knowledge layer is able to define each related category. Links between the two levels help the system understand that the concepts are learned, or missing links can define that they are guessed only. A main contribution is a new clustering algorithm for producing the pattern ensembles, that is itself an ensemble which then converges through agglomerates. To help with the problem of random data ordering, multiple solutions can also be combined using the same clustering technique, when the averaged result is more robust. Tests measure both how coherent the ensembles are, which means that every data row in the cluster belongs to the same category, and also the optimal number of clusters produced by a solution. Results show good accuracy over a set of benchmark datasets. As part of the theory, a teaching phase would help the classifier to learn the true category prototype in the knowledge layer. This knowledge would be global and then used to infer correct classifications in any unsupervised and local cluster, thereby reducing the teaching time.

Key-Words: classifier, self-organise, unsupervised, supervised, teach.

1 Introduction

This paper describes a design that can be used for Explainable AI and also autonomous clustering. It can probably be integrated into an existing cognitive model [1] (also [2]-[4]) at the boundary between the lower and middle levels, where the knowledge is aggregated. The design describes a lower level self-organising unit that is a nested ensemble of patterns. The upper level can be hierarchical, where each end node represents an individual concept, so there is a transition from mixed ensemble masses to specific categories. Pattern ensembles are learned in an unsupervised manner, using a new self-organising algorithm, but can then be split again when the upper knowledge layer is able to define each related category. Links between the two levels help the system understand that the concepts are learned, or missing links can define that they are guessed only. This paper proposes some new clustering algorithms for producing the pattern ensembles, that are themselves an ensemble which converges through agglomerates. To help with the problem of random data ordering, multiple solutions can also be combined using the same clustering technique, when the averaged result is more robust. Tests measure both how coherent the ensembles are, which means that every data row in the cluster belongs to the same category, and also the optimal number of clusters produced by a solution. Test results show good accuracy over a set of benchmark datasets. As part of the theory, a teaching phase would help the classifier to learn the true category prototype in the knowledge layer. This knowledge would be global and then used to infer correct classifications in any unsupervised and local cluster, thereby reducing the teaching time. This would lead to each category aggregating from several unsupervised clusters, but also feeding back to the ensembles to help define nesting. As the information is added, cross-referencing between the two structures allows it to be used more widely.

With this process, a unique structure can build up that would not be possible by either method alone. The upper level stores aggregated prototype information as categories, but must link back with the data sources that created it and represent features. If a pattern sub-cluster becomes associated with two or more categories, that sub-cluster is separated, but only needs to recognise the difference in the row sets that belong to its base cluster classifier, not the whole dataset. The discrimination problem is therefore made simpler by reducing the problem size. There is also a lot of cross-referencing between the self-organised clusters and the taught tree and the globally shared category information would help the classifiers to learn more quickly and to share partial results. The algorithms in this paper mostly use processes and equations that the author

has used previously, but it is more important to understand the broad algorithm and underlying theory, because a lot of the functions could probably be replaced by other ones.

The rest of this paper is organised as follows: section 2 describes some related work. Section 3 describes the unsupervised clustering theory, while section 4 describes two clustering algorithms that have been tried. Section 5 then introduces the supervised clustering theory with the teaching and section 6 gives some test results. Section 7 then gives a discussion on some open issues, while section 8 gives some conclusions on the work.

2 Related Work

A relatively recent AI topic is Explainable AI (XAI) [5]. With this, the system is able to give an explanation, in human terms, of how it came to a decision. This is intended to increase trust in the system that is no longer a black box, but can be more transparent. It would also allow humans to interact with the system more easily because it will have to share a common language for the explanation. DARPA (The US Defence Advanced Research Projects Agency) [6] consider this to be the next stage in AI, especially with regard to autonomous systems that may take actions on their own. Concerning this paper, there is a small amount of feedback available from the new structure that could be used to allow for more intelligent interaction by a human operator.

2.1 Unsupervised Learning

Other AI models have been developed based on similar types of design, with a knowledge layer sorting a pattern layer. The Adaptive Resonance Theory neural network [7][8], for example, has an architectural similarity. There are different variants, but it was designed with an unsupervised bottom layer that would try to match its input with a static set of categories (memory) in the upper layer. The upper layer is trained to cluster input from the lower layer into categories. After a category is learned, it only responds to new input that passes a matching vigilance threshold, and so future updates also maintain the current category learning. If there is no suitable match, a new category may be created by promoting a node in the upper layer to that status and using it as the new category prototype. Matching the input to the category uses a self-organising winner-takes-all approach and leads to a type of resonance between the two layers, which has often been compared with the human brain. The network was later found to suffer from a statistical property that meant the order in which the data was presented would affect how it was clustered. A relatively new version called TopoART [9] is able to address this problem, as the shapes of the clusters do not depend on the order of creation of the associated categories. The Frequency Grid algorithm [2] that is used later, also suffers from this problem, where an improvement is suggested in section 4.2.1. The comparison with the ART network is the upper knowledge layer that is a static memory and a lower self-organising layer. Both systems would balance weights, flowing in both directions, but different to the ART network would be that the self-organising units of this paper produce unlabelled categories that are then correctly labelled, or the labels are later corrected, whereas with the ART network it is the whole data row. The next example is therefore a closer match that provides another layer on-top of the ART network, to store the type of information that is imagined.

The paper [10] also uses a Fuzzy-ART network and explains it quite nicely. 'Fuzzy ART performs unsupervised learning of categories under continuous presentation of inputs through a process of 'adaptive resonance' in which the learned patterns adapt only to inputs considered to be relevant. Thus the ART models solve the so-called *stability-plasticity* dilemma where new patterns are learned without forgetting those already learned.' The paper tackles the problem of when an input might belong to more than 1 category, or cluster. They take a hybrid approach of combining the neural network with a background theory made from defeasible logic. Defeasible logic programming (DeLP) contains both strict and defeasible rules, in the form of Horn clauses. The rules are created as part of the clustering process and then help to determine future membership when there is ambiguity. It is interesting that this architecture would match with the cognitive model [1][3] and the first procedural logic design.

2.2 Supervised Learning

While generative models [11][12] are a hot topic, it is not clear that this design can make use of them. A generative model captures the essential features of a pattern distribution, but its purpose is then to produce new examples of it, not necessarily verify correctness. The upper knowledge layer of this paper would expect to gain information from external sources as well, and some attempt to summarise it might be considered. Because existing structure is used, Transfer Learning [13][14][15] might be a better option. With this, knowledge learned from one problem is applied to a different problem. The paper

[14] describes a process for labelling unlabelled data that is also self-taught. They describe that semi-supervised learning typically makes the assumption that the unlabelled data can be labelled with the same labels as the classification task, and that these labels are merely unobserved [16]. Transfer learning typically requires further labelled data from a different but related task, and at its heart typically transfers knowledge from one supervised learning task to another. Because self-taught learning places significantly fewer restrictions on the type of unlabelled data, in many practical applications it is much easier to apply. Then there is also self-taught clustering [15] that uses auxiliary data to learn the salient features in the problem dataset. This would be related to unsupervised clustering.

2.3 Clustering

Concerning other algorithms, the Self-Organising Map (SOM) [17] is obviously of interest, or SOM with extensions [18]. The teaching phase would more than likely be a winner-takes-all approach and would override what the self-organising clusters decide. There is no idea of a topology with this new algorithm however, just a similarity match. Other algorithms that consider sets of closest nodes include DBSCAN [19], kNN or k-Means [20]. k-Means clustering is a global method, where *k* data points are selected as centroids or prototypes and the other data points are clustered with the nearest centroid (mean). Measurements are therefore always taken with the centre of the cluster. kNN clustering is a local method, where a point is assigned to the class most common among its *k* nearest neighbours. In fact, the method of this paper clusters locally first in the unsupervised layer and would then correct that through global categories in the proposed supervised layer. DBSCAN stands for 'Density-based spatial clustering of applications with noise'. It uses a density-based approach, where points closely packed together, inside of a certain radius for example, are clustered together. A difference with these algorithms is that the new method considers not only the points nearest to the node in question, but the points nearest to its nearest nodes as well. In that respect, DBSCAN may be more similar, because considering these node sets would produce overlap in node selection, leading to a count for how many times any node is included, through its second-order associations. In that respect the algorithm is looking for a more densely packed region of shared nodes.

A fully-connected architecture has always been suggested for a biological model, see for example [21], which would support the idea of considering second-order associations. While the proposed whole system is most likely new, it is the use of these second-order counts, along with the other novel clustering algorithms (Frequency Grid [2]) that are the main contribution of this paper. Then if aggregating results are required, Random Forests [22][23] are another ensemble method that are used with Decision Trees [24]. While Decision Trees branch on attributes and not category, the clustering process is very similar. Training with Random Forests is probably quite different however. In that case, the dataset is split into *n* different sets, each with maybe 60-80% of the original dataset. Each variation is trained on a Decision Tree and the results are aggregated together. The Random Forest is therefore the training process that uses multiple variations of the dataset and also the aggregation process afterwards. Section 4 describes the clustering algorithms of this paper.

The paper [25] makes some interesting comments about Boolean Factor Analysis that would relate to this ensemble-hierarchy and may therefore be earlier related work. Their Hopfield-modified network takes the input signal vector and factors it into a low-level signal space of relations or clusters. The low-level factors would represent the first clustering stage. One idea is to further self-organise based on distinct features, as well as closest distances. Columnar characteristics can therefore become important and decisions can be taken, maybe with some judgement on related features. At the heart of Deep Learning [26] is the idea of learning an image in discrete parts. Each smaller part is an easier task and the next level can then combine the smaller parts until the whole image is learned. It might be interesting to compare the branching with something like this, because it also reduces the problem complexity.

3 Unsupervised Clustering Theory

Self-organisation is more often used to extract patterns from data, than to learn known categories and does this using some type of distance or similarity measurement. The self-organising process relies on some basic theories as follows: The process starts by associating every data row with the row it is closest to, according to some measure, such as Euclidean Distance. If each row is then clustered with its closest row, this should actually lead to natural breaks in the data that lead to a set of natural clusters. It is very likely that there will be more breaks than actual categories in the dataset and so each actual category will be represented by several

clusters. However, if each cluster is considered in isolation, it will also be found to have sub-clusters that can be recognised through the same closest link mechanism. These sub-clusters are only obvious when the larger enclosing pattern is removed and the cluster is considered by itself. The sub-clusters might then be helpful, because they can isolate data rows that do not really belong together. Clustering using centroids has to consider average values, where it cannot skew weight values to obtain a desired result. Therefore, data from different categories can easily get mixed together. A re-clustering phase would then be able to move the more isolated data rows to other clusters. Through this method, the cluster may become a centre of attraction for the category it represents and its centroid will become more accurate, as more and more data rows for the same category are added.

While that is the theory, it does not work out quite so well in practice. One big problem with self-organisation is the fact that it has to choose the centre of the data that it is clustering. The algorithm does not know what the actual category is and so it cannot directly discriminate. Taking any sub-clusters too literally is probably dangerous as well, because the averaged values rely on there being distance consistency across the whole cluster, which does not have to be the case. This is OK if there are few categories and the data is well-balanced, but the self-organising mechanism cannot learn any inherent skew in the dataset. A supervised approach, on the other hand, is able to adjust its discrimination lines, because it can be told directly about a particular error and so it can then adjust a weight set based on this. The teaching phase is therefore intended to make the self-organised patterns more accurate. It is postulated that because some of the classification has been learned and can be used as a sound basis, the teaching phase can help to build up a more accurate picture of the whole data set with fewer presentations. The upper knowledge layer is firstly presented with prototypes and so it knows what to look for in its other sources. An experience-based approach would look for these prototypes and verify their accuracy in the real world. Matching with an object in the real world would help to label it and confirm the structure, for example. It would then feed this new 'knowledge' back down to the unsupervised layer, to help to correct it and if a prototype has been learned from one cluster set, it is then global and can be used to verify other cluster sets as well.

4 Clustering Algorithms

Two clustering algorithms have been tried for this problem. A first attempt was based more on node distances and is not the current algorithm of interest, whereas a second attempt, based more on the Frequency Grid, is currently the algorithm of interest.

4.1 Distance-Based

This first attempt uses closest nodes to create the large clusters and then a frequency grid inside of each cluster to split them again, with the intention of providing some robustness through shifting the centroid centres each time. The frequency grid is equivalent to clustering based on popular count associations. The algorithm could be accurate in some cases but it produced too many clusters to be practical. This failing is shown in Table 1 of section 0. The self-organisation phase would cluster based on closest distance, but it would also try to create the largest and most coherent cluster sets possible. Algorithm 1 gives an example for this type of clustering.

Algorithm 1. Closest-Distances Example

1. Link each data row with the row it is closest to, according to some measure.
2. Create clusters by placing all data rows that are linked together into a cluster.
3. For each cluster:
   a. Use a Frequency Grid to do a count of the rows any other row is closest to.
   b. Use the grid to create sub-clusters in the cluster.
   c. Special cases include a sub-cluster with only 1 entry, or single column features.
4. Create branches in each base classifier for each sub-cluster part and create a centroid for each sub-cluster. Also add a new sub-cluster for any additional rows.
5. Try to combine any of the base clusters as follows:
   a. Determine an average distance $\bar{u}$ between the sub-clusters in the cluster.
   b. Determine a distance x between two clusters.
   c. If the distance x is less than the average sub-cluster distance $\bar{u}$, then combine the two clusters.
6. Re-calculate the centroids for each cluster and sub-cluster.
7. Take each data row in turn again and add it to the cluster whose centroid it is now closest to. Go to step 3.

8. The process can stop when data rows are not moved or the total number of clusters does not change.

4.2 Distance and Frequency Grid

The algorithm of section 4.1 could be accurate in some cases, but it is also a bit messy. Relying on node distances inside of clusters is not very reliable. The average row position and distance can change across the whole cluster, for example. The algorithm of this section is a lot cleaner but also more simplistic and so extensions to it are also suggested. This second algorithm uses a brain-inspired idea of full linking between the nodes, to find a better closest match. A difference with something like DBScan, for example, is that this algorithm considers not only the points nearest to the node in question, but the points nearest to its nearest nodes as well. Each clustering phase measures the k-nearest clusters to every other cluster, where a cluster is a single node to start with. But instead of just considering the node's k-clusters, it aggregates all of the k-clusters for the nodes closest to the node in question. The intention is to produce a more robust association count, which can consider that while a node may be closer, if it is really part of a different category, it will have other associations that the rest of the k-cluster nodes do not have. Filtering over a combined and cross-referenced list for several mini clusters, would therefore help to remove this as noise. The algorithm for the brain-inspired with frequency grid clustering is described in Algorithm 2.

Algorithm 2. Single Pass Algorithm

1. Set each node, representing a single data row, to be a separate cluster.
2. Create a new layer of clusters using Algorithm 3.
3. Set the new layer as the next layer to cluster and check the stopping criteria.
   3.1. This can include a closest match with a preferred number of clusters.
4. If stopping criteria not met, then Go to step 2.

Algorithm 3. Brain-Inspired Frequency Grid

1. Measure the distances between all of the clusters and for each cluster, store the closest *k* other clusters.
2. Each cluster then is associated with *k* other clusters and they are also each associated with *k* other clusters.
3. For each cluster:
   3.1. Do a count over all of the associated cluster names, to find the most popular *k* clusters overall.
   3.2. Store this set as the local cluster set for the cluster node.
4. Convert each cluster set into an event, or input data line for the frequency grid.
   4.1. Train the frequency grid with the input data lines and it will produce another set of clusters, based on most popular association counts.
   4.2. Create a new layer in the model that groups all nodes (clusters) suggested by the frequency grid together.
   4.3. Let each new cluster be represented by its centroid value, or averaged data row.

4.2.1 Ensemble Improvement

The problem with the frequency grid is that it is sensitive to the order in which nodes are processed. This includes data row ordering, or simply the order that they occur in a layer of nodes. If the dataset is randomly ordered therefore, it will have an effect on the result and so it may be better to produce several results that are related to each other and aggregate over those, so that the more commonly occurring parts are kept and the more-noisy parts are removed.

There are at least two solutions to the problem of random ordering. One option would be to make the set membership fuzzy, where the frequency grid would allow a node to be a member of all clusters that have an equal association count with it. But that drastically changes the nature of the frequency grid. It is supposed to have that type of relationship for between-cluster links, but not for a full membership of more than one cluster. It would in effect, merge those cluster parts, when they would lose some meaning. The second option is to use an ensemble of solutions, in the style of random forests. For this problem, the random solutions stem from the same base and so they have the same root set of values. This is simply something that might be statistically significant. Aggregating over several solutions, the more commonly occurring parts will remain and the noisy or less commonly occurring parts will receive lower statistical counts and can be removed. The batch process that uses the ensemble algorithm is described in Algorithm 4.

Algorithm 4. Batch process for averaged test results

1. While (run ensemble test)
   a. For (i = 0 to test runs)

i. Generate a randomised data ordering.
ii. Use Algorithm 3 to produce a cluster set.
iii. Save the cluster set as a solution set.

b. Process all solution sets together, using the ensemble method of Algorithm 5.
i. This produces something like (tr*tr*n) new solutions from the multiple single-pass algorithm runs.

c. Convert the new solutions to cluster lists of node names.
d. Use cluster lists as the input to another frequency grid. Only one final stage, so don't randomise.
e. Convert frequency grid clusters to a new cluster list of node names.
f. Create new layer and nodes from that ordering and cluster using Algorithm 3.
g. Save these clusters as a solution.

2. If stopping criterion met, then stop.
3. Average the results to give the final result set.

Algorithm 5. Ensemble algorithm

1. For (i= 0 to phases - *helps to reduce the number of clusters*)
   a. Convert last solution set to cluster lists of node names.
   b. For (j = 0 to test runs)
      i. Randomise the first cluster list ordering.
      ii. Use as input to a frequency grid.
      iii. Convert frequency grid clusters to second cluster list of node names.
      iv. For (k = 0 to test runs)
         1. Randomise the second cluster list ordering.
         2. Create new layer and nodes from that ordering.
         3. Cluster the new layer using Algorithm 3.
         4. Save the result as a solution set for the next phase.

4.2.2 Consolidate through Hierarchy

In fact, there is a further opportunity to use a clustering process at the very end of the ensemble tests. If each ensemble run produces a result, then instead of averaging the result scores, the result solutions can be stored and used as the input node sets for a final clustering process. This introduces the idea of hierarchical clustering that can cluster the cluster units themselves. It is more like the whole process being repeated in new levels.

5 Supervised Teaching Theory

As part of a theory, new to the system would be a second supervised stage that would teach the correct category and help to explain the classification decisions. When a value for an actual category is learned, this information might also be used to measure a confidence that the classification is correct. For example, if links between the self-organised cluster and the taught cluster do not exist, then there is a high probability of a guess, but if links are present, then the information of the self-organised cluster can be used with confidence. Figure 1 is a graphic that describes some of the processes, including a new layer of linking nodes.

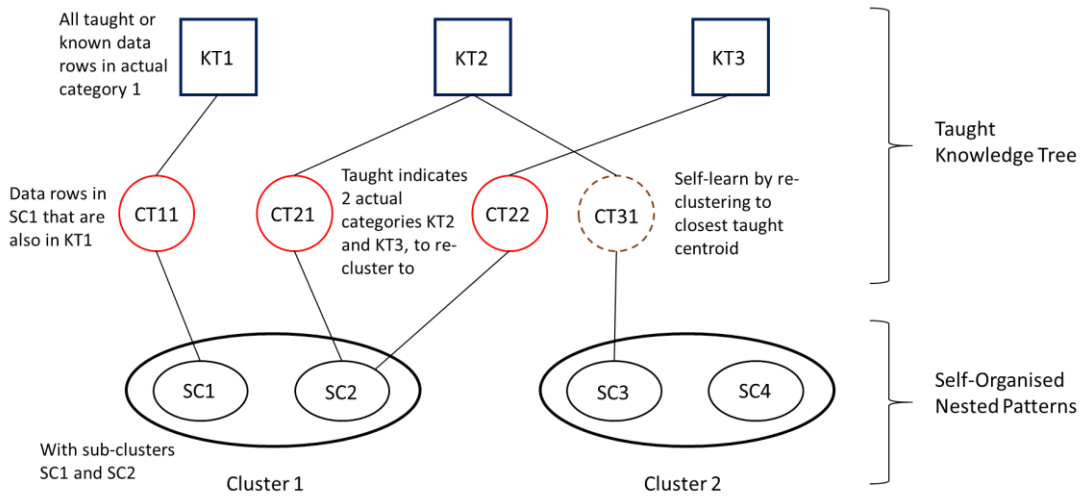

Figure 1. Graphic of the possible interactions between the two cluster structures.

Through this process, unlabelled categories can be labelled. The idea is that in the real world we may make some assumptions based on what we can determine, but we would also know that they are guesses. We would wait for proof before considering them to be true and we would then use the 'known' knowledge to correct any of the related assumptions. The self-taught learning system of [14] probably has similar aspirations and they note that self-taught learning perhaps also more accurately reflects how humans may learn, since much of human learning is believed to be from unlabelled data.

With the supervised training phase, the classifier is allowed to ask for the actual category(s) of the selected cluster(s) and will make a permanent record of that in the knowledge layer, with links back to the unsupervised parts. Once learned, the classifier can therefore ask the knowledge layer for some matching evidence and use it to correctly classify a new part of some unsupervised cluster. Any new proof is added to the knowledge tree as global evidence, from where it can be used by any of the cluster groups. As the more local and unsupervised parts become aggregated in a supervised category, that category prototype can return a truer centroid value. This again looks like branching on category type and not feature and so it should maintain links to the source data that created it. In fact, the unsupervised clusters are separated based on features and then link with the knowledge trees that are based on category. It would also be expected that the knowledge layer in a real system could learn from other sources as well, making the prototype representations more robust.

5.1 Tree Structure

As more rows are learned, the category node in the knowledge tree therefore becomes more accurate and there may be a constant ripple effect of updating the unsupervised mini clusters and re-assigning the data rows with minimal structural changes. To start with however, more major structural changes are likely, until the knowledge layer prototypes stabilise. A tree structure is appropriate because that is how the category groups can correct themselves and it reflects the same nesting in the ensemble. For example, ensemble group SC2 has been split into abstract nodes CT21 and CT22, because the known prototypes indicate two categories in the ensemble group SC2. Further learning might indicate that, for example, abstract node CT21 can also be split into more categories. That split would produce a new tree level with new abstract nodes that would re-link the tree structure to their respective prototypes. After the prototypes are stable however, it may be a case of structurally changing the new clusters only. The tree structure however starts to look like Category Trees [4], which is known to be very accurate.

6 Implementation and Testing

It has only been possible to test the self-organising structure so far. A computer program has been written in the C# dotnet language and used to test some benchmark datasets that can all be found in the UCI Machine Learning repository [27]. Two tests have been carried out, one for the algorithm of section 4.1 and one for the algorithm of section 4.2. No information about the clusters was given to the program, except for the desired number as a stopping criterion. The clusters were generated internally by the program, resulting in sets of nodes inside of each cluster that would hopefully be coherent with each other, meaning that they would all belong to the same actual category. It was then possible to calculate the error for those as follows:

1. For every sub-cluster, retrieve from the dataset, the category for each row.
2. Remove the set of rows with the largest count for a single category.
3. The coherence error is then the number of rows left.

So, for example, if a cluster set contains data rows for categories as follows: A, A, A, B, B, then there would be 2 incorrect nodes. If the dataset actual categories were: A, A, A, B, B, C, C, then the error would be 4. The tests in section 6.1 are really only a marker and a more complete set of tests is described in section 6.2.

6.1 Distance-Based

As a marker to compare with, tests were firstly carried out on the algorithm of section 4.1. If the data was already well separated and the number of actual categories was low, then the self-organising process could realise the original categories by itself, but a stopping criterion, or knowing when it had converged might be problematic. This was the case for the Iris [28] and the Wine [29] and Zoo [30] datasets. A lot of other datasets showed that the self-organising structure cannot perform well enough by itself. This was also found to be the case in [18] who used variants of the SOM to successfully cluster the Iris data but could not cluster the Abalone dataset,

for example. It was also a characteristic of the process that in cases when incoherent data was higher, it might start with a smaller number of clusters, but by trying to reduce this, the number would in fact increase. So, by trying to move data rows from the first-assigned cluster would in fact fragment the clusters more. Other factors such as re-combining clusters, can also really increase the error.

Dataset	Incoherent	S-O Clusters	Actual Clusters
Iris	2 of 150	30	3
Wine	4 of 178	18	3
Zoo	7 of 101	18	3

Table 1. Example of self-organised coherence. 'Incoherent' shows how many data rows were not coherent, or of the same actual category, as the rest of their cluster. 'S-O Clusters' shows how many separate clusters were created. 'Actual Clusters' gives the correct number.

6.2 Distance-Based and Frequency Grid

This section gives test results for the algorithm of section 4.2, which could probably be considered for practical problems. For this test, the preferred number of clusters was entered and the algorithm was run and stopped at the number of clusters just above or equal to the preferred number. The algorithm would initially decide the cluster sets using the closest-node associations and then reduce the number using the frequency grid. Comparisons were made between ordered and random datasets, and the single pass algorithm or the ensemble version. The only configuration for the ensemble was the number of runs inside the ensemble, set at 10 and the number of closest-node associations, set at 5, meaning a total of 5x5 = 25 node counts.

Using an ensemble is still a heuristic process and the result of each ensemble search can be different. The program therefore ran a number of ensemble tests for each dataset configuration and then averaged the results to get the final totals. With the ensemble testing, there would be a maximum of 50 separate test runs for each dataset configuration. Each run would produce an averaged number of clusters and percentage of correct associations, for the clustering process. Those 50 results would then be averaged to produce a final result for the test. The test results are listed in Table 2.

	O:B	O:B,F	R:B,F	R:B,F,E	R:B,F,E,H
Iris (150-3)	4-68%	5-96.5% V1	5-61% V1	6-87.5% V1	4-89.5% V1
Wine (178-3)	4-93%	4-97% V1	11-62% V1	6-86% V1	5-91.5% V1
Zoo (101-7)	12-79%	8-92% V1	8-52.5% V1	12-83% V1	8-88% V1
Hayes-Roth (132-3)	5-44.5%	4-42% V1	4-44% V2	9-51.5% V1	4-47.5% V1
Heart Disease Cleveland (303-5)	8-57.5%	6-56% V1	6-55% V1	8-58% V2	6-58% V2
Sonar (208-2)	3-54%	4-59% V1	3-55% V1	5-62% V1	6-61% V2
Wheat Seeds (210-3)	5-89.5%	5-88.5% V1	5-60.5% V1	5-85.5% V2	5-87.5% V1
Average Cluster Error	2.14	1.43	2.29	3.57	1.71
Average Accuracy Percentage	69.5%	76%	55.5%	73.5%	74.5%

Table 2. Comparison of clustering methods: O: Ordered dataset, R: Random dataset, B: Brain-Inspired Closest nodes, F: Frequency Grid, E: Ensemble, H: Hierarchy.

As a comparison, the first column gives the result using the closest-nodes clustering only, as described in section 4.1. It shows the number of clusters the test was stopped at, followed by the accuracy percentage for that number. The second column shows the full clustering algorithm, described in section 4.2, for ordered datasets, for a single test run. The third value in the cell is which version of the

frequency grid was used. The third column shows the full algorithm for a single run on randomly ordered datasets and then the fourth column runs the same test but uses the ensemble result of section 4.2.1. The fifth column adds a final clustering stage, described in section 4.2.2. There are two versions of the frequency grid algorithm, indicated by 'V1' or 'V2' in the table. The original version V1 seemed to work better, but in fact a lot of the comparisons were very close. Each cell value therefore indicates the best number of clusters with the related accuracy percentage and then which frequency grid version was used.

With relation to other heuristics, a value above 90% is considered to be acceptable for unsupervised clustering for the more separable datasets, such as the Iris dataset. That percentage level could be achieved for a single run, for example, 97% accuracy was possible, but with maybe twice as many clusters. In general however, the clustering result fell a bit below that level. Adding the final clustering stage does not seem to improve the accuracy by very much, but it does appear to reduce the number of clusters, which suggests that it has helped to consolidate the result. This means that it can get closer to the desired number of clusters through the frequency grid stages. What is of interest is the difference between the first random column 3 and the ensemble versions, where the ensemble consistently outperforms the single clustering run. Comparing with the ordered dataset results of column 2, there is now a much smaller difference and so the ensemble process has compensated for the loss of accuracy in the frequency grid. The whole process however is quick and easy to use and even if the frequency grid is not the strongest algorithm for a particular problem, using the ensemble framework has consistently improved the results. Some other datasets with larger numbers of rows actually produced good results but training time was too slow (several hours) for the ensemble method at the moment. So this is certainly not a universal classifier but it can work well in some cases.

7 Discussion

While a teaching stage is proposed, it has not been fully formalised yet. It is interesting with respect to Explainable AI, which is important for improving trust in the system. For one thing, the reduced training time and the ability to infer from another cluster's update would make the system more independent. It is then also able to reason about how confident it is in its decision. If there are no links from an ensemble part to the hierarchy part, for example, then the system can say that it does not know for certain that the ensemble part is correct, or that the system should try to find out more about that section of data. The system of this paper should also fit in well with the whole cognitive model [1] and with the ensemble-hierarchy structure in that model. It is interesting that the tree nodes become abstract representations of the ensembles, essentially for linking only, and the prototypes are aggregations of those. The transitions from local to global, or feature to category look very nice and fit well together, but that is probably just making explicit what is implicit in the standard models.

It is probably the case that unsupervised clustering cannot be as accurate as supervised and that is probably a good thing. On the one hand, the supervised clustering is making use of a lot of other information that has already determined what the correct clusters are, probably input by a human user. On the other hand, the human must have made these decisions at some stage through an unsupervised process and so if enough information is available, the computer system should be able to do it as well. The author still thinks that if AI becomes intelligent enough to take over, there is no reason to think that it will be evil, but is more likely to do good. However, the 'paper clips' scenario and the consequences of that is a bit clearer to him now. If a system is able to self-organise accurately enough with just some raw data, then it does not have any understanding outside of that and so it might in fact make bad decisions while having good intentions. So that could make the system more dangerous. Then again, if it ever reached a human level of intelligence, it would surely be able to learn when it had made a mistake and be able to correct it.

8 Conclusions

This paper describes an unsupervised clustering approach that can then be corrected through a teaching process. The teaching phase may also allow the system to infer other classifications by itself and therefore reduce the time required to learn correctly. Data rows are firstly assigned to a classifier in an unsupervised manner, which represents them using a centroid. If there are errors, then further layers can create new centroids for subsets of the classifier rows. These centroids therefore define paths through the classifier branches and guide an input to its closest match. As part of the unsupervised system, it would probably be better to have more smaller clusters that are more accurate. This is because the supervised part can provide additional help not available for the unsupervised clustering, but the unsupervised part still needs to make a reasonable attempt at producing accurate estimates. The

estimates might then be used for something like Transfer Learning, for example. The test results in section 6.2 suggest that the unsupervised clustering method could be a practical choice for some applications. This paper explains the theory of the process and has described some unsupervised results only. It will be difficult to implement the whole system and there are variations on what the best procedures might be, but if Category Trees [4] form the tree structure, then they are known to be accurate.

As part of the ensemble learning, a final clustering stage does give the prospect of a hierarchical system in the ensemble clustering, that might increase its accuracy at each level. The clustering process would repeat, but instead merge results from the previous ensembles through a new frequency grid, where tests indicate that it would improve the accuracy. The test results show that it has consolidated the cluster structure and so this would also be passed on to the next stage. Any increase in accuracy would be attributed to the algorithm framework and so the frequency grid heuristic could be replaced by another heuristic and results could still be aggregated together. So that is really interesting for a modular system such as the human brain, for example.

References:

[1] Greer, K. (2020). New Ideas for Brain Modelling 7, available on arXiv at https://arxiv.org/abs/2011.02223.

[2] Greer, K. (2019). New Ideas for Brain Modelling 3, Cognitive Systems Research, Vol. 55, pp. 1-13, Elsevier. doi: https://doi.org/10.1016/j.cogsys.2018.12.016.

[3] Greer, K. (2018). New Ideas for Brain Modelling 4, BRAIN. Broad Research in Artificial Intelligence and Neuroscience, Vol. 9, No. 2, pp. 155-167. ISSN 2067-3957.

[4] Greer, K. (2018). An Improved Oscillating-Error Classifier with Branching, WSEAS Transactions on Computer Research, Vol. 6, pp. 49 - 54. E-ISSN: 2415-1521. For the updated version, see Category Trees (2020), available on arXiv at https://arxiv.org/abs/1811.02617.

[5] Dosilovic, F.K., Brcic, M. and Hlupic, N. (2018). Explainable Artificial Intelligence: A Survey, MIPRO 2018 – 41st International Convention Proceedings, Opatija, Croatia, pp. 232 – 237.

[6] DARPA. (2017). Explainable Artificial Intelligence, https://www.darpa.ml/program/explainable-artificial-intelligence. (last accessed 16/4/20)

[7] Carpenter, G.A. and Grossberg, S., 1988. The ART of adaptive pattern recognition by a self-organizing neural network. Computer, 21(3), pp.77-88.

[8] Carpenter, G.A. and Grossberg, S., 2010. Adaptive resonance theory.

[9] Tscherepanow, M., 2010, September. TopoART: A topology learning hierarchical ART network. In International Conference on Artificial Neural Networks (pp. 157-167). Springer, Berlin, Heidelberg.

[10] Gómez, S.A. and Chesnevar, C.I. (2004). A Hybrid Approach to Pattern Classification Using Neural Networks and Defeasible Argumentation, In Flairs conference, pp. 393-398.

[11] Goldt, S., Reeves, G., Mézard, M., Krzakala, F. and Zdeborová, L. (2020). The Gaussian equivalence of generative models for learning with two-layer neural networks. arXiv preprint arXiv:2006.14709.

[12] Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A. and Bengio, Y. (2014). Generative adversarial nets. In Advances in neural information processing systems, pp. 2672 - 2680.

[13] Bozinovski, S. and Fulgosi, A. (1976). The influence of pattern similarity and transfer learning upon training of a base perceptron B2. (original in Croatian) Proceedings of Symposium Informatica 3-121-5, Bled.

[14] Raina, R., Battle, A., Lee, H., Packer, B. and Ng, A.Y. (2007). Self-taught learning: transfer learning from unlabeled data. In Proceedings of the 24th international conference on Machine learning, pp. 759-766.

[15] Dai, W., Yang, Q., Xue, G.R. and Yu, Y. (2008). Self-taught clustering. In Proceedings of the 25th international conference on Machine learning, pp. 200-207.

[16] Nigam, K., McCallum, A., Thrun, S. and Mitchell, T. (2000). Text classification from labeled and unlabeled documents using EM. Machine Learning, Vol. 39, pp. 103 - 134.

[17] Kohonen, T. (1990). The Self-Organising Map, Proceedings of the IEEE, Vol. 78, No. 9, pp. 1464 - 1480.

[18] Canetta, L., Cheikhrouhou, N. and Glardon, R. (2005). Applying two-stage SOM-based clustering approaches to industrial data analysis, Article in Production Planning and Control, DOI: 10.1080/09537280500180949.

[19] Ester, M., Kriegel, H-P., Sander, J. and Xu, X. (1996). A Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise, Proceedings of 2nd International Conference on Knowledge Discovery and Data Mining (KDD-96).

[20] Wu, X., Kumar, V., Quinlan, J.R., Ghosh, J., Yang, Q., Motoda, H., McLachlan, G.J., Ng, A., Liu, B., Yu, P.S., Zhou, Z-H., Steinbach, M., Hand, D.J. and Steinberg, D. (2008). Top 10 algorithms in data mining, Knowl. Inf. Syst., Vol. 14, pp. 1 – 37. DOI 10.1007/s10115-007-0114-2.

[21] Anderson, J.A., Silverstein, J.W., Ritz, S.A. and Jones, R.A. (1977) Distinctive Features, Categorical Perception, and Probability Learning: Some Applications of a Neural Model, Psychological Review, Vol. 84, No. 5.

[22] Breiman, L. (2001). Random Forests. Machine Learning, Vol. 45, No. 1, pp. 5 - 32.

[23] Adnan, M.N. and Islam, M.Z. (2015). Improving the random forest algorithm by randomly varying the size of the bootstrap samples for low dimensional data sets. In Proceedings of the European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning, pp. 391 - 396.

[24] Breiman, L., Friedman, J., Stone, C.J. and Olshen, R.A. (1984). Classification and regression trees. CRC press.

[25] Frolov, A.A., Husek, D., Muraviev, I.P. and Polyakov, P.Y. (2007). Boolean Factor Analysis by Attractor Neural Network, IEEE Transactions on Neural Networks, Vol. 18, No. 3, pp 698 - 707.

[26] Hinton, G.E., Osindero, S. and Teh, Y.-W. (2006). A fast learning algorithm for deep belief nets, Neural computation, Vol. 18, No. 7, pp. 1527 - 1554.

[27] UCI Machine Learning Repository (2019). http://archive.ics.uci.edu/ml/.

[28] Fisher, R.A. (1936). The use of multiple measurements in taxonomic problems, Annual Eugenics, 7, Part II, pp. 179-188, also in 'Contributions to Mathematical Statistics' (John Wiley, NY, 1950).

[29] Forina, M. et al. (1991). PARVUS - An Extendible Package for Data Exploration, Classification and Correlation. Institute of Pharmaceutical and Food Analysis and Technologies, Via Brigata Salerno, 16147 Genoa, Italy.

[30] Zoo database (2016). https://archive.ics.uci.edu/ml/datasets/Zoo. (last accessed 16/9/20)