Modeling with Fuzzy ARTMAP

Kai Bernardini

Abstract

Adaptive Resonance Theory (ART) is a family of artificial neural network models, pioneered by Stephen Grossberg and Gail Carpenter, that learn to categorize, recognize, and predict objects and events in a non-stationary world. ART models are primarily motivated by modeling human phenomenology with supporting data coming from experimental neuroscience. As such, this class of models fall under the auspices of computational neuroscience rather than statistical learning theory. That said, ART-based models offer up solutions to many problems that stifle modern Machine Learning. This paper explores such problems, and offers up simulations in support of this claim. More concretely, since ART models dynamically self stabilize, offer support for fast and slow learning, and retain previously learned information when presented with novel stimuli, such models offer a powerful alternative to Deep Learning. This paper provides a brief overview of ART, its supervised extensions, and explores the associated geometric interpretation.

I. Introduction

A. On the Problem of Learning From Data

With the widespread availability of large scale computing and seemingly endless amounts of data, statistical machine learning problems have become not only pervasive, but have also exploded in both complexity and scope. This revolution has breathed new life into the longstanding goal of creating artificial general intelligence.[1] This goal can be (informally) summarized as a program that is indistinguishable from [or superior to] a human operator with respect to some task [3]. In the early days of artificial intelligence, the field rapidly solved problems that are mentally difficult for humans to solve quickly, but are relatively straight-forward to describe with a series of formal, mathematical rules. Ironically, such mathematically formal tasks that are among the most difficult mental undertakings for humans, are easy for a computers. On the other hand, tasks that are both difficult to describe mathematically and relatively simple for humans to solve tend to be incredibly difficult for computers. That is, the biggest challenge to artificial general intelligence is solving tasks that are trivial for people to perform but hard *to describe formally*. In particular, they consist of a class of problems that we as humans solve intuitively–almost automatically without any mathematical formalism.[2]

The example commonly used to illustrate this concept is the task of describing precisely what makes a "chair" a "chair." This challenge is typically presented as a loaded philosophical dilemma, as there is currently no rigorous way to distinguish the concept of "being a chair" from the concept of "not being a chair." In the "One and Three Chairs art installation", the artist Joseph Kosuth represents one chair three different ways. The first is the physical embodiment, and is a manufactured chair. The second is as a photograph of the same chair (which is the usual way that machines observe objects) and the final way is a copy of a dictionary entry for the word "chair." The installation is thus composed of an object, an image, and a collection of words that define an abstract concept. [4] As humans, we can easily associate these markedly different formulations of a chair, but it is difficult for machines to perform the same task.

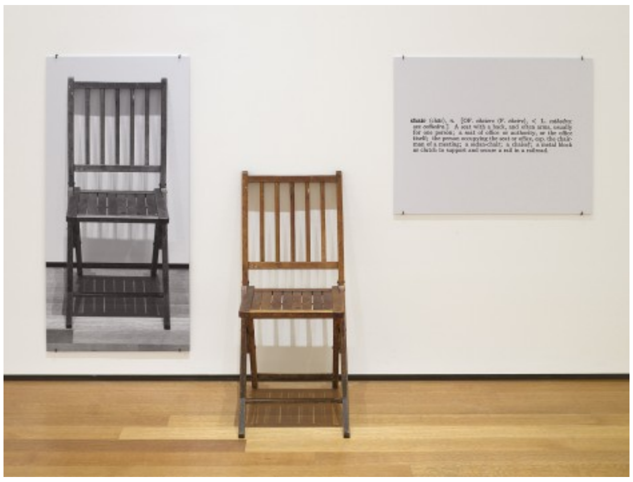

Fig. 1: One and Three Chairs

[4]

B. *The Limitations of Modern Deep Learning*

While modern methods are still unable to produce human readable rules for what defines a chair, recent breakthroughs in a subset of modern machine learning (commonly known as "deep learning") have allowed models to distinguish and identify various abstract objects by using a data driven approach. In particularity, given a massive collection of labeled examples in the form of pixel arrays, deep neural networks can be trained to consistently identify abstract objects. The method of building these models is fairly standard and involves incrementally reducing classification error by the method of gated gradient descent.[2]

Building models this way is only possible due to the ubiquity of of massive datasets, as well as the widespread availability of distributed computing, as such models require tremendous amounts of data. As an example, consider the amount of training required by DeepMind's DQN to surpass human performance at a number of Atari games. Lake et al. writes "The DQN was trained on 200 million frames from each of the games, which equates to approximately 924 hours of game time (about 38 days), or almost 500 times as much experience as the human received. Additionally, the DQN incorporates experience replay, where each of these frames is replayed approximately 8 more times on average over the course of learning."[5] The general rule of thumb for supervised classification with DNNs at a bare minimum 10,000 exemplars per output class.[2] In contrast, humans require comparatively little data to learn a new a new concept. For example, after sitting in a handful of chairs, a child will quickly learn what a chair is. Nobody shows them 10,000 photos of a chair. As Santoro et al. write, "This kind of flexible adaptation is a celebrated aspect of human learning, manifesting in settings ranging from motor control to the acquisition of abstract concepts. Generating novel behavior based on inference from a few scraps of information – e.g., inferring the full range of applicability for a new word, heard in only one or two contexts – is something that has remained stubbornly beyond the reach of contemporary machine intelligence."[6]

Another important limitation to address is the fact that deep neural networks learn stationary functions, whereas the world we inhabit is inherently non-stationary.[7] One critical symptom of this problem is known as *"plasticity- stability"* problem [8]. In particular, traditional gradient descent optimized neural networks learn through extensive iterative training. When novel data is presented to the network, the model must (inefficiently) relearn its parameters to adequately incorporate the new information without catastrophic interference[6]. Biological neural networks allow humans to quickly learn new information, without disrupting old knowledge. For example, learning how to ride a bike shouldn't interfere with a person's ability to identify their mother's face. Therefore, it is reasonable to assume that any model that is capable of producing AGI, most overcome this problem. In the context of deep learning, Multilayer Perceptrons (and its extensions) are susceptible to exploding gradients, which will annihilate previously learned patterns resulting in catastrophic interference. In general, this is known as catastrophic forgetting which is characterized by a sudden inability for an artificial neural network to a remember previously acquired knowledge upon learning something new.[8] Said more succinctly, the learning is unstable. On the other hand, if learning is to slow, no new information can be internalized in the model, and the model will be unable to incorporate new information. This manifests itself in deep learning in the form of vanishing gradients. There are many flavors of backpropogation that reduce the severity of these problems, but to my knowledge, none actually eliminate them entirety.

II. Adaptive Resonance Theory

A. *Motivation*

Motivated by building a model that both solves these shortcomings and is actually motived by the underlying biological mechanisms that power the brain, Grossberg designed a novel model for learning called Adaptive Resonance theory (ART). Adaptive Resonance theory describes a class of neural network models based on empirical scientific analysis to emulate how the brain processes information and solves problems. While there are many attractive features of ART and its extensions, the primary feature of an ART model is that learning is adaptive, self stabilizing, and it (therefore) solves the plasticity-stability dilemma. It also supports fast learning, allowing the network to learn quickly and reliably from few training samples.[7]

From the perspective of an engineer, ART is, at its core, an unsupervised clustering algorithm, where the operator does not need to know how many categories there are *a priori*. ART performs pattern matching by comparing a bottom up input signal to a top down expectation. In particular, the signal is matched against the memory of a long term memory cell. If the bottom up input is sufficiently "matches" the active code, this leads to a resonant state which allows learning to occur. If the match is not satisfactory, this triggers a search over other active codes for a better match. In the event another similar active code is found which does match sufficiently well, the internal memory representation has the option of remaining unchanged, or incorporating some of, or all of the new information derived from the bottom up input. If no such active code is found, a new node is committed and learns the current input. Grossberg calls this *match-based learning* and it forms the foundation for why ART has stable learning.[7] In particular, match-based learning allows memories to change only when bottom up inputs are close enough to internal expectations, or when a completely new input is encountered.[8]

B. ARTMAP

A natural extension to the original unsupervised ART model is the supervised variant called ARTMAP. While there are many flavors of ARTMAP, this paper only considers the simplest version that supports multidimensional analog values: Default Fuzzy ARTMAP.[10] This system is composed of several fields of neurons. F_0 is the field of nodes that represents the current input vector and receives the input signal I. F_1 is the field of nodes that receives both bottom up inputs from F_0 and top down expectations from F_2. F_2 represent the active code categories and is a list of weight vectors $w_1, \dots, w_C$ where C is the number of active codes.[9] Before entering F_1, the input vector $I = (\mathbf{a})$, $\mathbf{a} \in [0,1]^m$ is preprocessed via a *compliment coding* as follows:

$$\mathbf{a} \mapsto (\mathbf{a}, \mathbf{a}^c),$$

where

$$\mathbf{a}^c = \begin{bmatrix} 1-a_1 \\ \vdots \\ 1-a_M \end{bmatrix} = 1 - \mathbf{a}$$

Notice that as a direct result of this, the ℓ_1 norm of every encoded input signal is constant.[10] In particular,

$$||I||_1 = ||(\mathbf{a}, \mathbf{a}^c)||_1 = \sum_{i=1}^{M} \Big((a_i + (1 - a_i) \Big) = \sum_{i=1}^{M} 1 = M$$

Hence, $||I||_1$ is fixed for all inputs. Let $\mathbf{x}$ denotes the F_1 activity (choice score) and $\mathbf{y}$ denotes the F_2 category as a 1 hot encoding. Associated to each node j in F_2 is an adaptive weight

$$\mathbf{w_j} = \begin{bmatrix} w_{j1} \\ \vdots \\ w_{j2M} \end{bmatrix}$$

with M being the number of original inputs in F_0 and $w_{ij}(t)$ denoting the ith entry in the jth adaptive weight at time t. At $t = 0$, each category is uncommitted and the weight values are uniformly set to 1. When a category node codes it first input, it becomes committed and takes on the value of the bottom up input.

There are also several parameters that determine the geometric properties of the model as well as whether or not learning is fast or slow. The first parameter is $\alpha > 0$ called the choice parameter. The relative contribution from the bottom up input and top down expectation is determined by α. If α

is small, categories with small norms are favored. This results in large category feature spaces with a fewer numbers of categories. On the other hand, if $\alpha \sim 0$, this will favor larger norms[11]. The parameters that controls the speed of learning is the learning rate $\beta \in [0,1]$. This determines whether or not the model uses fast learning, or slow learning. For $\beta = 1$, and in a resonate state, this forces the adaptive weight vector to be set to the bottom up input. As β decays to 0, the amount of new information from the input that the weight vector incorporates decays to zero. Arguably the most important parameter is $\rho \in [0,1]$. This is the vigilance parameter and is used to ensure resonance takes places only when the input is sufficiency similar to the top down expectation. It also heavily influences the number of unique categories, as well as the geometry of the model.[12] In D-ARTMAP, the baseline vigilance $\bar{\rho} = 0$, however, empirical estimates carried out in this paper suggest this is not always a good decision.

During training, when an input I enters F_0, the way in which the category that "most resembles the bottom up input" is determined by the *choice function*. For each category $j = 1,\ldots,C$ in F_2 define the choice function

$$T_j(I) = \frac{||I \wedge \mathbf{w_j}||}{\alpha + ||\mathbf{w_j}||}$$

where $(p \wedge q)_i \equiv \min(p_i, q_i)$ and $||\bullet||$ is the ℓ_1 norm defined by $||p|| = \sum_i |p_i|$. The resulting category choice J is

$$J = \arg\max_j \left(T_j(I)\right) = \arg\max_j \left(\frac{||I \wedge \mathbf{w_j}||}{\alpha + ||\mathbf{w_j}||}\right)$$

Intuitively, this can be thought of the most similar category using fuzzy intersection as the measure of similarity. Further, the chosen category J can be thought of as a hypothesis which will be matched to the top down expectation. The way a prediction class is derived from the chosen category is by associating output classes to category nodes. More specifically, when a F_2 node is first committed, it is coded as the output class of the input that originally committed it. For example, if an input A has output class k, and is used to commit a node J, then for all new inputs I' that are matched to J, the model predicts class k.[12]

C. Learning in Default Fuzzy ARTMAP

The next stage of the ARTMAP system is to test its hypothesis for the category J. When the Jth category is selected, the *match function* computes a score for how closely the bottom up input matches the top down expectation. The criterion that determines whether or not the hypothesis is confirmed, is called the *vigilance criterion.* The match function is

$$\frac{||I \wedge \mathbf{w_J}||}{||I||}$$

and the vigilance criterion is

$$\frac{||I \wedge \mathbf{w_J}||}{||I||} \geq \rho \iff ||I \wedge \mathbf{w_J}|| \geq M\rho$$

If the chosen category J is does not satisfy the vigilance criterion, a parallel memory search is initiated to find the a category that does. I.E., it fails to confirm the map field prediction made by $\mathbf{w}_J$ which triggers a search for a better category. The chosen category can therefore be summarized as

$$J = \arg\max_j \left(\frac{||I \wedge \mathbf{w_j}||}{\alpha + ||\mathbf{w_j}||} \right)$$

$$\text{s.t. } ||I \wedge \mathbf{w_j}|| \geq M\rho$$

If the above system is unsatisfiable, a new category $C+1$ is committed with with weight vector A and output class k where k is the output class of I. If the vigilance criterion is satisfiable, then the models prediction for the input I is the associated output class k for category J. If the prediction matches the ground truth label, the system enters a learning or *resonant* state, wherein the top down expectation $\mathbf{w}_j$ is updated via

$$\mathbf{w}_j \leftarrow \beta \Big(I \wedge \mathbf{w}_J \Big) + \Big(1 - \beta \Big) \mathbf{w}_J$$

On the other hand, if the ARTMAP system detects a predictive error

$$||\mathbf{x}|| < \rho ||y||,$$

then the orienting system is initiated. *Match tracking* raises the vigilance parameter just enough to trigger a search for a new F_2 coding node. A signal from the map field to the orienting subsystem causes ρ to "track" the F_1 match. That is, ρ increases until it is larger than the match value. Typically, a small value ε is added to the match value so that

$$||I \wedge \mathbf{w_j}|| < M\rho$$

by setting

$$\rho \leftarrow \frac{||I \wedge \mathbf{w_j}||}{M} + \varepsilon$$

If

$$\varepsilon > 0 \implies \frac{||A \wedge \mathbf{w_J}||}{||A||} < \rho$$

Note that under this update, it is impossible to select the same category that resulted in the error as the previously selected category node J now fails to meet the matching criterion. Hence, the model will search for another F_2^a node. This process continues until either a satisfying output class J is found with correct predictive label, or new node is committed with the latter occurring when the vigilance is raised high enough to prevent any category from being selected. The vigilance is then reset to its baseline value $\rho \leftarrow \bar{\rho}$. [12] As a remark, match-based learning is the opposite of error-based learning (such as GD learning) wherein the model responds to false predictions by slowly changing internal weights to reduce the difference between the ground truth and the prediction. This is in contrast to match-based learning which will respond to a misclassification with a search for a better match. [7]

III. Geometry of ARTMAP

To help motivate this section, lets consider a contrived example. Suppose we uniformly generate some synthetic data in $[0,1]^2$. Represent the data matrix X as pairs of (x_{i1}, x_{i2}) with $x_{ij} \in_R [0,1]$ for $i = 1, \ldots N,\ j = 0, 1$. Let the ground truth value be determined by the boolean expression

$$y^i \leftarrow (x_{i1}^2 + x_{i2}^2 \leq .4)$$

Using a pseudo random number generator, the following dataset is generated and plotted.

A. Example 1: Chunked Disk

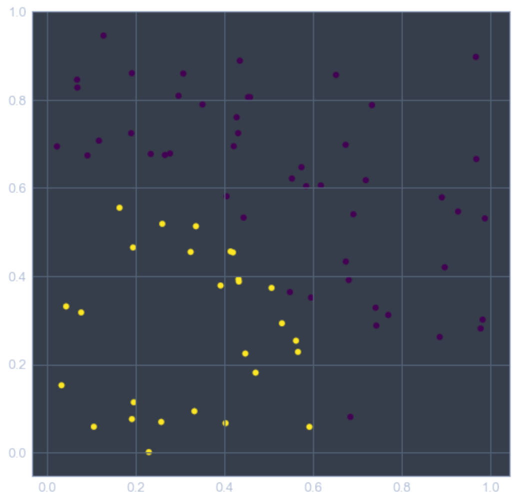

Fig. 2: Disk Chunk Classification task

After running Fuzzy ARTMAP for 1 epoch using winner take all coding on the inputs using Fast learning with $\alpha = .05$ and $\bar{\rho} = 0$, the following decision boundary is generated by classifying all points in a mesh grid ranging from $(0,0)$ to $(1,1)$.

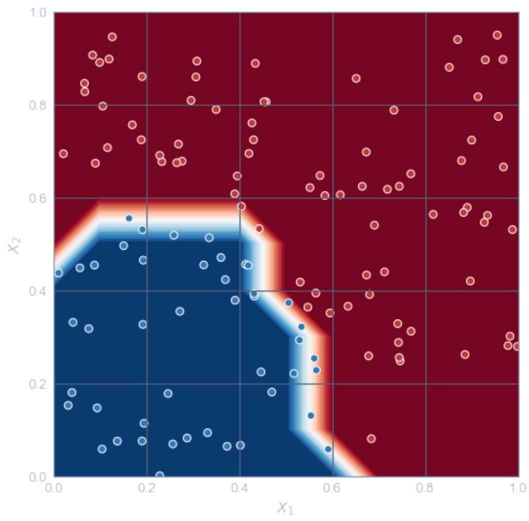

Fig. 3: Disk Chunk Decision Boundary

On an independent test set, where each input $I \mapsto (a, a^c)$, there are no predictive errors. The geometric interpretation of this classifier is each category weight vector $\mathbf{w}_j = [\mathbf{u}_j, \mathbf{v}_j^c]$ can be used to define a rectangle, where the minimum vertice (with respect to distance from the origin) is the two dimensional vector $\mathbf{u}_j$ and the maximum vertice (again with respect to distance from origin) is the two dimensional vector $\mathbf{v}_j$. These boxes cover the input space $[0,1]^2$ and can be used to interpret the models rational for a particular prediction by decomposing each choice into a series of IF THEN statements. In the 2-D case, the size of the rectangle R_j defined by u_j, v_j is

$$|R_j| \equiv |v_j - u_j|$$

which is exactly the height plus the width of

R_j[10].

After training, the model outputs the class associated to the weight vector J that for each R_J equals the smallest rectangle that encloses all vectors that have chosen category J, such that

$$|R_j| \leq 2(1-\rho)$$

For this particular example, the non-linear decision boundary is encoded via the following rectangles derived from the learned adaptive weights.

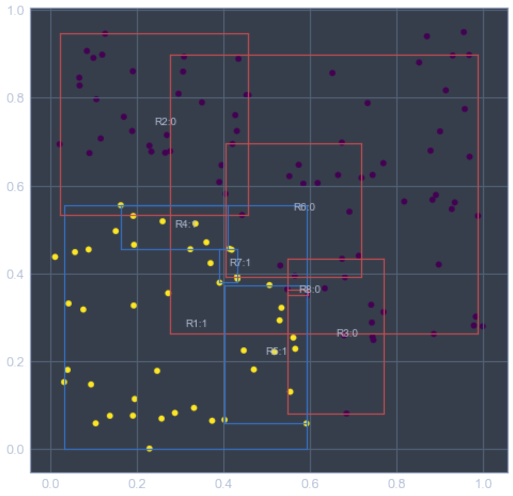

Fig. 4: Classification Rectangles for $\bar{\rho} = 0$

B. Example 2: Inner/Outer Circles

Another example to consider is generating values uniformly distributed across a centered inner annulus and an outer annulus assigning values of 0,1 for the ground truth respectively. Notice that these values are not in $[0,1]^2$ by default, and additional processing is necessary. The simplest map that comes to mind is a min-max scaler. For the datamatrix X, compute

$$X' = \frac{(X - \min(X))}{\max(X) - \min(X)}$$

where the minimum is taken across the columns. Set the new scaled X_s

$$X_s = X'(\max(X) - \min(X)) + \min(X)$$

A simplifying assumption is made that new values that are transformed using the same min and max values do not exceed 1, and are at least 0. Note that we can remedy this by applying a sigmoid squashing function after transformation if the values deviate from the max/min by a small amount. After transforming the data, the following plot is created.

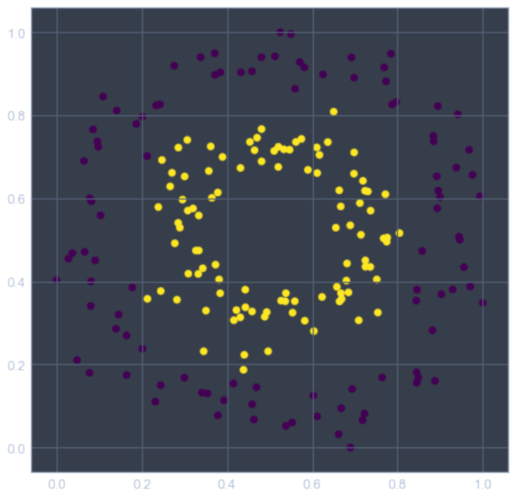

Fig. 5: Inner/Outer Circle Classification Task

As before, the model is trained with winner take all coding, fast learning, and $\bar{\rho} = 0$, resulting in the non-linear decision boundary shown below.

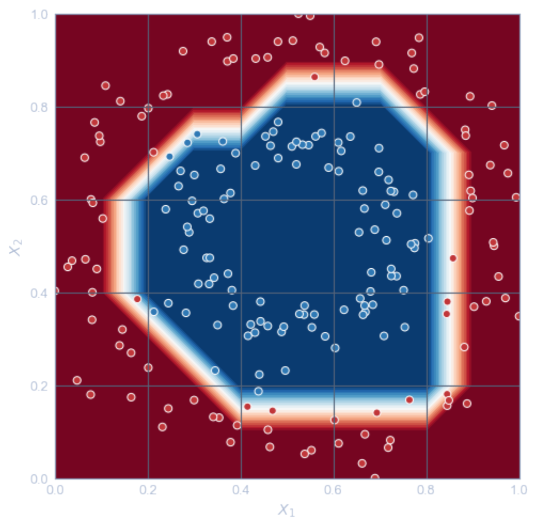

Fig. 6: Learned Decision boundary for Inner/Outer Circle Classification Task with $\bar{\rho} = 0$

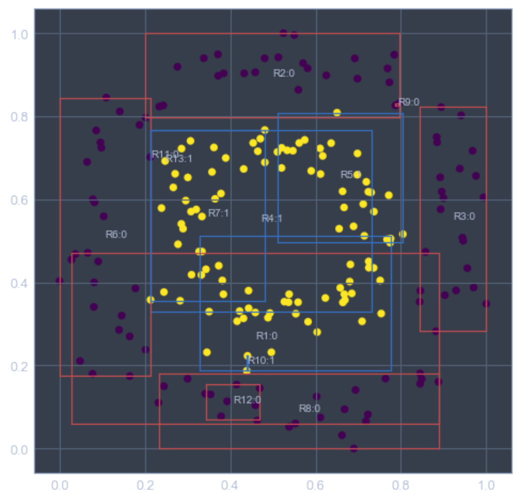

Fig. 7: Decision Rectangles for Inner/Outer Circle Classification Task with $\bar{\rho} = 0$

IV. The Role of the Vigilance Parameter

In the general case of M dimensional analog inputs, the hyper-rectangle R_j includes the two vertices $\wedge_j \mathbf{a}$ and $\vee_j \mathbf{a}$ where the ith entry of each vector is

$$(\wedge_j a)_i = \min\{a_i : \mathbf{a} \text{ has been coded by category } j\}$$

and

$$(\vee_j a)_i = \max\{a_i : \mathbf{a} \text{ has been coded by category } j\}$$

The size of R_j is given by

$$|R_j| = ||\vee_j \mathbf{a} - \wedge_j \mathbf{a}||_1$$

and weight vectors

$$|\mathbf{w}_j| = \sum_i (\vee_j a)_i + \sum (1 - (\vee_j a)_i)) = M - |\vee_j \mathbf{a} - \wedge_j \mathbf{a}|$$

as

$$\mathbf{w}_j \overset{\text{def}}{=} (\wedge_j \mathbf{a}, (\vee_j \mathbf{a})^c)$$

$$\implies |R_j| = M - |\mathbf{w}_j|$$

Hence,

$$|\mathbf{w}_j| \geq \rho M \implies |R_j| \leq (1 - \rho) M$$

This gives a very clear geometric intuition for how ρ affects the size of the rectangles. If $\bar{\rho} = 0$, there is no maximum size for the rectangle (aside from the verticies must be analog values). On the other hand, if $\rho \sim 1$, then the size of the rectangles approaches zero. A consequence of this is as $\rho \rightarrow zero$, $C \rightarrow N$ where N is the number of training examples. In particular, this slowly turns fuzzy ARTMAP into an exemplar model, where each input is assigned its own category node. Under this condition, the model simply memorizes the training set. This is because the vigilance becomes “too picky” and will constantly fail to match bottom up inputs to top down expectations. Note that the number of rectangles is bounded, and will rarely converge to the exemplar model. To illustrate the effect of ρ on the number of rectangles,and their respective sizes, consider the spiral classification task.

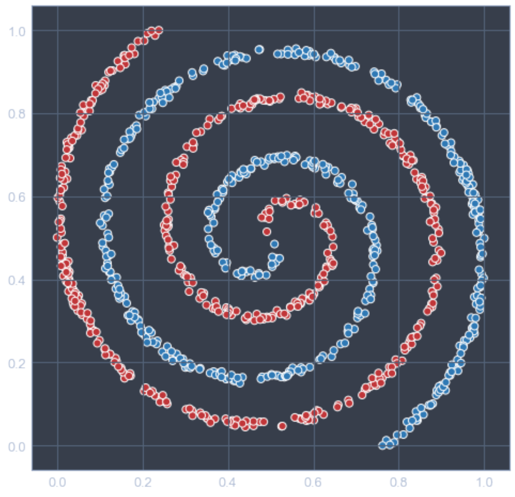

Fig. 8: Spiral Classification Task

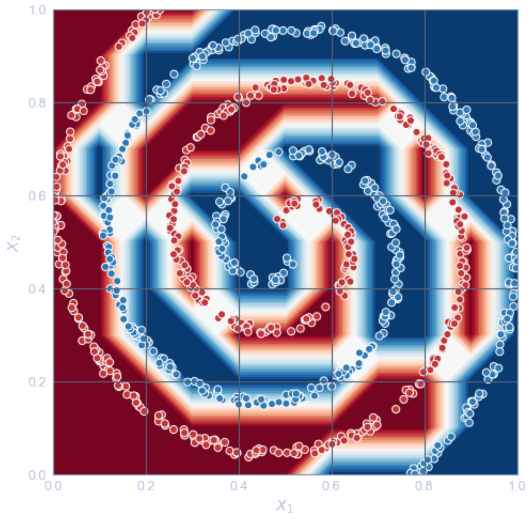

Fig. 9: Decision Boundary and Classification rectangles for $\rho = 0$

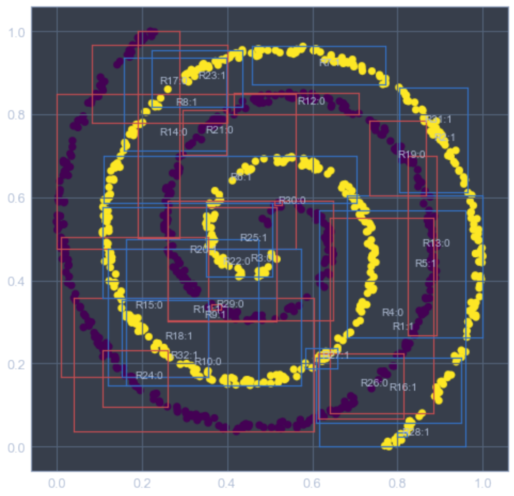

Fig. 10: Classification rectangles for $\rho = 0$

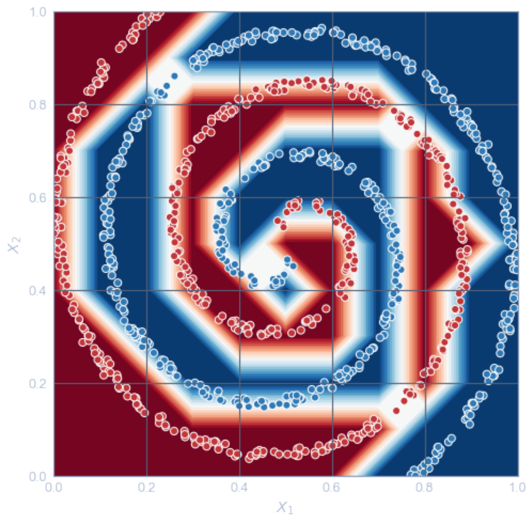

Fig. 11: Decision Boundary and Classification rectangles for $\rho = .1$

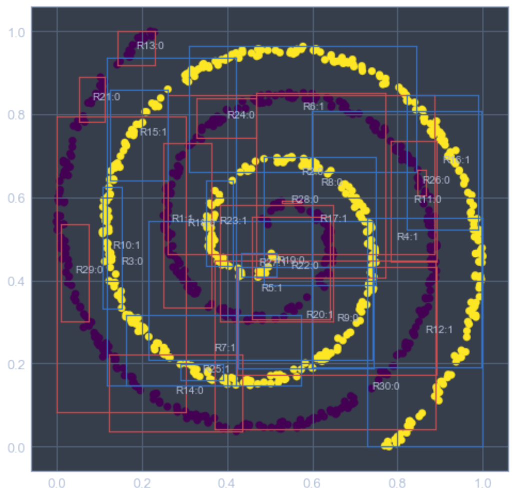

Fig. 12: Classification rectangles for $\rho = .1$

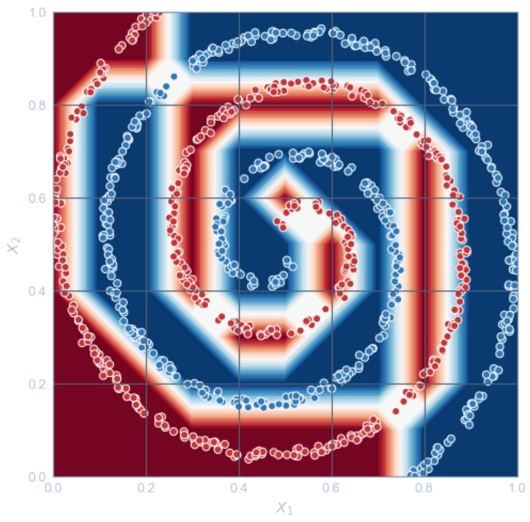
Fig. 13: Decision Boundary and Classification rectangles for $\rho = .15$

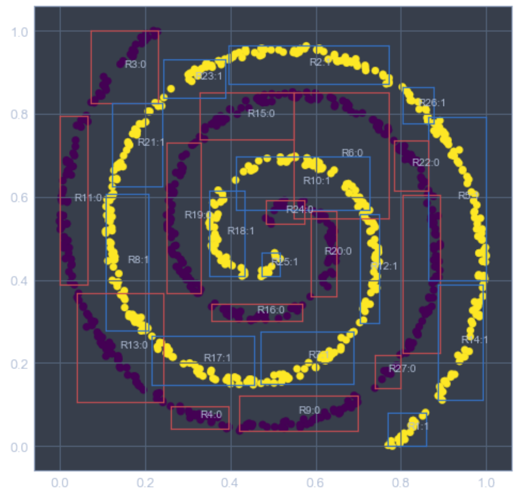
Fig. 14: Classification rectangles for $\rho = .15$

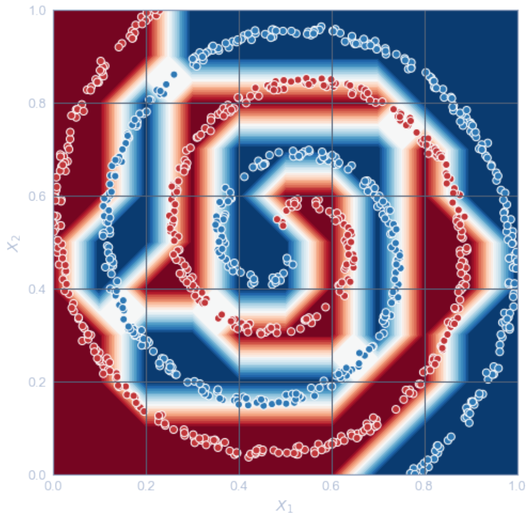
Fig. 15: Decision Boundary and Classification rectangles for $\rho = .6$

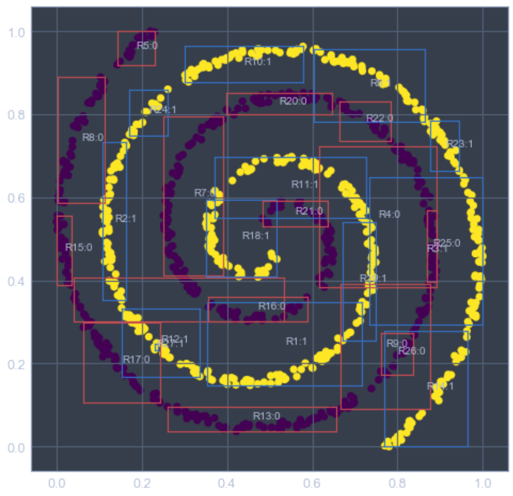
Fig. 16: Classification rectangles for $\rho = .6$

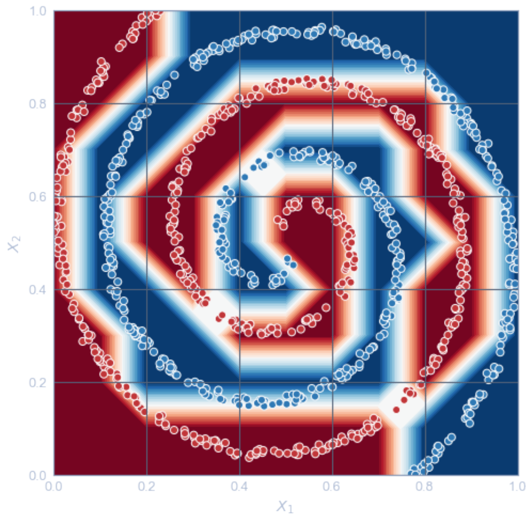
Fig. 17: Decision Boundary and Classification rectangles for $\rho = .9$

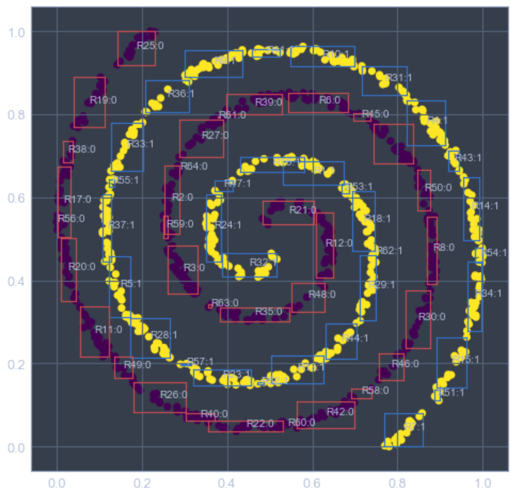
Fig. 18: Classification rectangles for $\rho = .9$

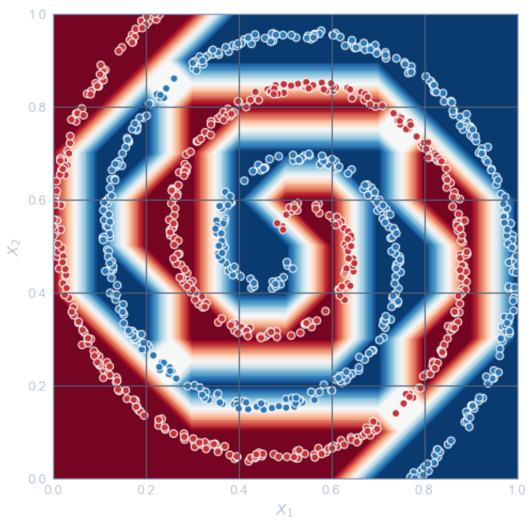

Fig. 19: Decision Boundary and Classification rectangles for $\rho = .95$

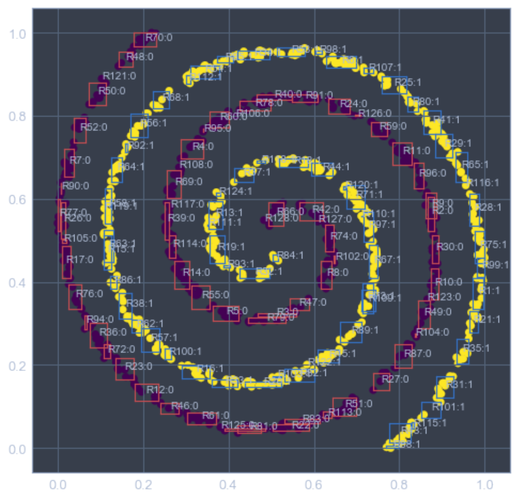

Fig. 20: Classification rectangles for $\rho = .95$

Notice how as $\rho \to 1$, the number of boxes increases, but after a. certain point, it remains bounded. Carpenter showed that using compliment coding with winner take all coding and fast learning bounds the number rectangles. Notice further, that the decision boundary itself does not change much, but the boxes that cover the data become more localized as ρ increases. What is incredible about this particular construction is 1) how quickly the model converges to the right answer, and 2), how accurate it is. It perfectly fits a training set, and also perfectly fits an independent testing sample. This is not the case for a multilayer perception. An empirical example located at `https://github.com/kaidb/Geometry_of_ML_Talk` demonstrates that a MLP with 50 neurons per layer and 6 hidden layers can perfectly fit the data. At first glance, this seems comparable to ARTMAP but the optimization for such a model requires hundreds of iterations of gradient descent. The learning is painfully slow, and even then, a poor choice of hyper parameters (such as a regularization coefficient) will result in poor testing performance. ARTMAP in each example here was trained using fast learning for 1 epoch and in each case, produced comparable accuracy to a DNN. Further, the DNN model is unstable with respect to the choice of hidden layers. In the spiral example, the training set is perfectly fit for models with 6 hidden layers, 9 hidden layers, but it fails to exceed an accuracy score of .6 for other values less than or equal to 10. This makes the optimization and parameter selection incredibly tricky in general, and is why Deep Learning has acquired a reputation of requiring "black magic" [2]. ARTMAP has only 1 parameter that is critical for model performance: $\bar{\rho}$. This isn't nearly as challenging as the selection of a Deep Learning architecture, regularization, activation functions, and other such values directly responsible for how well the DNN is optimized. Baseline vigilance is easily selected using cross validation and structured grid searches. This makes the training and model selection process almost automatic.

To wrap up this discussion, several models on 2 benchmarking datasets are built using standard statistical learning theory tools, and ARTMAP. The first is a binary classification problem of identifying whether or not a patient has breast cancer, and the second is a multi class classification problem for predicting the attribute class of an iris plant. In both cases, ARTMAP yields robust, state of the art predictive accuracy. The modeling process is carried out as follow: allocate 70% of the data for training, and 30% for testing. In each model, the model is fit with the training set, and the accuracy score is taken over the testing set. The Multi Layer Perception models are built with 3 hidden layers with 5 neurons per layer. It is feed forward with rectified linear unit activation and optimized using ADAM (a variant of gradient descent with momentum).

	Breast Cancer	Iris
Fuzzy ARTMap	0.957447	0.96
Logistic Regression	0.952128	0.84
Linear SVM	0.957447	0.98
CART	0.920213	0.98
Tuned MLP	0.978723	0.96

As can be seen, Fuzzy ARTMAP offers comparable performance to other statistical learning models, and does so with little to no tinkering. This makes ARTMAP an appealing modeling framework for many real world applications.

References

[1] Friedman, Jerome, Trevor Hastie, and Robert Tibshirani. The elements of statistical learning. Vol. 1. New York: Springer series in statistics, 2001.

[2] Goodfellow, Ian, Yoshua Bengio, Aaron Courville, and Yoshua Bengio. Deep learning. Vol. 1. Cambridge: MIT press, 2016.

[3] Turing, Alan M. "Computing machinery and intelligence." In Parsing the Turing Test, pp. 23-65. Springer, Dordrecht, 2009.

[4] Kosuth, Joseph, "One and Three Chairs.", MOMA, NY 1965

[5] Lake, Brenden M., Tomer D. Ullman, Joshua B. Tenenbaum, and Samuel J. Gershman. "Building machines that learn and think like people." Behavioral and Brain Sciences 40 (2017).

[6] Santoro, Adam, Sergey Bartunov, Matthew Botvinick, Daan Wierstra, and Timothy Lillicrap. "One-shot learning with memory-augmented neural networks." arXiv preprint arXiv:1605.06065 (2016).

[7] Carpenter, Gail A., and Stephen Grossberg. Adaptive resonance theory. Springer us, 2017.

[8] Grossberg, Stephen. "Adaptive Resonance Theory: How a brain learns to consciously attend, learn, and recognize a changing world." Neural Networks 37 (2013): 1-47.

[9] Carpenter, Gail A., Stephen Grossberg, and John H. Reynolds. "ARTMAP: Supervised real-time learning and classification of nonstationary data by a self-organizing neural network." Neural networks 4, no. 5 (1991): 565-588.

[10] Carpenter, Gail A., Stephen Grossberg, Natalya Markuzon, John H. Reynolds, and David B. Rosen. "Fuzzy ARTMAP: A neural network architecture for incremental supervised learning of analog multidimensional maps." IEEE Transactions on neural networks 3, no. 5 (1992): 698-713.

[11] Williamson, James R. "Gaussian ARTMAP: A neural network for fast incremental learning of noisy multidimensional maps." Neural networks 9, no. 5 (1996): 881-897.

[12] Carpenter, Gail. Default artmap. Boston University Center for Adaptive Systems and Department of Cognitive and Neural Systems, 2003.

[13] https://github.com/kaidb/cn8112