Random Fuzzy-Rule Foams for Explainable AI

Akash Kumar Panda and Bart Kosko

Abstract A random foam trains several fuzzy-rule-foam function approximators and then combines them into a single rule-based approximator. The foam systems train independently on bootstrapped random samples from a trained neural classifier. The foam systems convert the neural black box into an interpretable set of rules. The fuzzy rule-based systems have an underlying probability mixture structure that gives rise to an interpretable Bayesian posterior over the rules for each input. A rule foam also measures the uncertainty in its outputs through the conditional variance of the generalized probability mixture. A random foam combines the learned additive fuzzy systems by averaging their throughputs or rule structure. A random foam is also interpretable in terms of its rules, its posterior of its rules, and its conditional variance. Thirty 1000-rule foams trained on random subsets of the MNIST digit data set. Each such foam system had about 93.5% classification accuracy. The random foam that averaged throughputs achieved 96.80% accuracy while the random foam that averaged only their outputs achieved 96.06% accuracy. The throughput-averaged random foam also slightly outperformed a standard random forest that output-averaged 30 classification trees. Thirty 1000-rule foams also trained on a deep neural classifier that had 96.26% accuracy. The random foam that averaged these foam throughputs was itself 96.14% accurate. The random foam that averaged their outputs was just 95.6% accurate. The appendix proves a Gaussian combined-foam version of the fuzzy approximation theorem for additive systems.

1 Explaining Neural Black Boxes with Fuzzy Rule Foams

Deep neural classifiers are black-box function approximators that map pattern inputs to class labels through multiply layers of neurons. They do not explain their predictions or the inner workings of their black box. So they do not yield explainable AI or XAI systems [1, 5, 6, 18]. Nor do they give a confidence measure for their input-output predictions. Yet some predictions should be more credible than others given the nature of the pattern inputs and given what the neural system has learned or forgotten in prior bouts of training. These XAI problems tend to get only worse as the network gets deeper.

We recently showed that an additive fuzzy rule-based approximator can learn from a trained neural classifier and thereby produce an interpretable system [16]. The rules' if-part sets vary in size in analogy with bubbles in a foam as in Figure 1.

Department of Electrical and Computer Engineering
University of Southern California,
Los Angeles, California e-mail: kosko@usc.edu

Varying the size of the rules controls the granularity of the approximators and helps mitigate the fuzzy system's inherent exponential rule explosion.

Rule foams have a natural representation as a generalized probability mixture $p(y|x)$ [12]. A set of m fuzzy if-then rules $R_{A_1 \to B_1}, \dots, R_{A_m \to B_m}$ defines such a generalized probability mixture $p(y|x) = p_1(x)\ p_{B_1}(y|x) + \cdots + p_m(x)\ p_{B_m}(y|x)$ so long as the fuzzy system $F : \mathbb{R}^n \to \mathbb{R}^p$ is additive [12]. So a rule foam converts a sampled neural network into an explanatory probabilistic system by way of the rules. The mixture's likelihood functions $p_{B_1}(y|x)$ absorb the structure of the fuzzy rules. The foam's mixture gives rise to a Bayes theorem that in turn gives a complete posterior probability description of the rule firings. This Bayesian posterior gives an input-by-input description of the inner workings of the fuzzy-system proxy for the sampled deep neural network. The mixture's second moment further gives a conditional variance that describes the uncertainty or confidence of the foam system's output for each given pattern input. So the rule foam defines an XAI system that is modular and self-descriptive and does so completely in probabilistic terms.

This paper introduces a *random* rule foam in rough analogy to a random forest of decision trees [3] that arise from sampling with replacement from a data set. Figure 1 shows the random-foam training process. A deep neural classifier first trains on a set of pattern data. Then we form several rule foams by randomly sampling *with replacement* from the trained neural classifier and thus perform bootstrap sampling from the neural network itself (or more generally from any data set). We then combine the rule foams by directly combining their rules or their throughput structure because the foams are additive fuzzy systems. Simulations show that such throughput combination tends to outperform simply combing the foam outputs.

The random construction tends to reduce the output conditional variance. Figure 2 shows how this output condition variance helps with the classification process. The random foam maintains its mixture structure because mixing mixture gives a mixture. The additive fuzzy structure goes further and gives a simple double-sum form for the overall mixture. This sum structure allows throughput combination in addition to the simpler output combination.

The random foams substantially outperformed the individual rule foams that comprised them. Simulations showed that random foams slightly outperformed random forests for the same number of combined systems. Random forests lack the self-explanation structure of the random foams. The random foam closely approximated the neural network that it trained on and achieved a similar classification accuracy.

Figure 3 shows the Bayesian posterior over the rules for a given input image and observed classification output. This information also helps explain the classification process. Figure 1 shows the bubble-like detail of rule foam. Figure 4 shows how classification advantage of using random foams. The appendix presents a multi-foam Gaussian generalization of the original fuzzy approximation theorem for additive systems [10, 11, 13]. The next section presents the pertinent mixture structure of additive fuzzy systems.

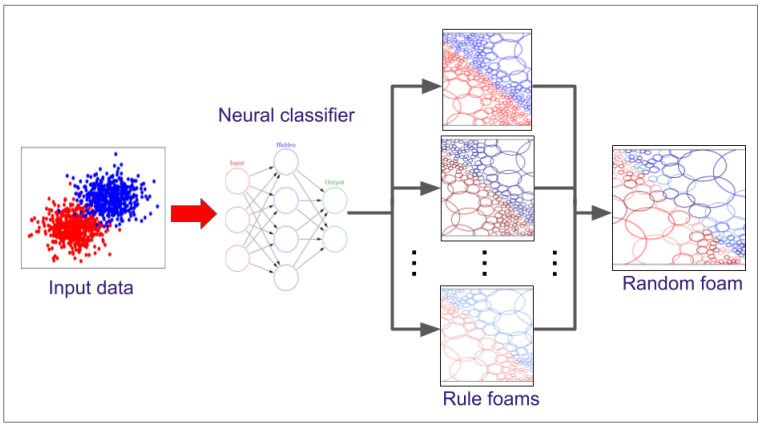

Fig. 1 Random foam structure: The deep neural classifier first learns from the data. Then several rule foams randomly sample from the trained network and independently learn from those random samples. The random foam combines these foams by additively combining or mixing rules.

2 Standard Additive Model (SAM) Fuzzy Systems

A standard additive model (SAM) fuzzy system approximates the function $f : \mathbb{R}^d \to \mathbb{R}$ using m "if-then" fuzzy rules. The j-th rule has the fuzzy if-part set $A_j \subset \mathbb{R}^d$ and the fuzzy then-part set $B_j \subset \mathbb{R}$. The fuzzy sets A_j and B_j have their respective multi-valued indicator functions $a_j : \mathbb{R}^d \to [0, 1]$ and $b_j : \mathbb{R} \to [0, 1]$ such that $a_j(x) = \text{Degree}(x \in A_j)$ and $b_j(y) = \text{Degree}(y \in B_j)$. The jth fired then-part set $B_j(x)$ has set function $b_j(y|x) = a_j(x)b_j(y)$ because the system is standard. The system's then-part set $B(y|x)$ sums the rule then-part sets. Then the fuzzy rules define a generalized probability mixture [12]:

$$p(y|x) = \frac{b(y|x)}{\int b(y|x)dy} = \sum_{j=1}^{m} \frac{w_j a_j(x)V_j}{\sum_{j=1}^{m} w_j a_j(x)V_j} \frac{b_j(y)}{V_j} = \sum_{j=1}^{m} p_j(x)p_{B_j}(y) \tag{1}$$

where a_j and b_j are the jth rule's respective if-part and then-part fuzzy set functions. The jth rule weight is w_j and $V_j = \int b_j(y)dy$ is the volume of then-part set B_j. The SAM structure $b_j(y|x) = a_j(x)b_j(y)$ gives the likelihood as $b_j(y|x) = b_j(y)$.

The function approximation is the conditional expectation of the mixture:

$$F(x) = E[Y|X = x] = \int y\ p(y|x)\ dy = \sum_{j=1}^{m} p_j(x)c_j \tag{2}$$

where c_j is the j-th then-part centroid.

2.1 Fuzzy Approximation Theorem for Gaussian SAMs

A SAM fuzzy system is an universal function approximator [10]. It can uniformly approximate any continuous function f defined on a compact set using finite number

of rules m. This approximation may need m that grows exponentially with the number of dimensions d leading to rule explosion.

Consider a SAM with m Gaussian rules that approximates the continuous function f on a compact set. Let ε be the approximation error. The function f is uniformly continuous. So we can find some positive δ so that

$$||x - \hat{x}_j|| < \delta \implies |f(x) - f(\hat{x}_j)| < \varepsilon/2. \tag{3}$$

Theorem 1 in the appendix shows that the Gaussian SAM can uniformly approximate f with approximation error ε and with m rules if

$$m \propto \frac{1}{\delta^d} \ . \tag{4}$$

The fuzzy rule foam mitigates this rule explosion by controlling the bubble size.

2.2 Conditional Variance

The SAM measures the uncertainty in its output through the conditional variance

$$V[Y|X = x] = \sum_{j=1}^{m} p_j(x)\sigma_{B_j}^2 + \sum_{j=1}^{m} p_j(x)(c_j - F(x))^2 \tag{5}$$

where σ_{B_j} is the j-th rule's then-part dispersion. The second term in equation (5) imposes an interpolation penalty on the system for guessing with respect to the given set of rules. We are confident in the system's output if the variance is low and we do not trust the system's output when the variance is high.

Consider a SAM that approximates the simple function f that represents an ideal classifier. A simple function maps an input space to a finite number of values [17]. The classifier's output has lower conditional variance in the class interior and has higher conditional variance at the class boundary. The misclassification rate is higher in regions of high conditional variance. Neural classifiers lack a conditional variance or indeed any intrinsic measure of their system uncertainty. SAMs that train on a neural network's output can measure the uncertainty in the network's classification.

Figure 2 shows the conditional variance of a 2-class fuzzy rule based classifier. The conditional variance is high near the class boundary. The misclassification rate is also high close to the class boundary.

2.3 Bayesian Posteriors of Subsystems and Rules

The generalized mixture $p(y|x)$ in (1) has the same form as the elementary theorem on total probability. The mixing weights $p_j(x)$ define generalized rule priors and the $p_{B_j}(y)$ define generalized rule likelihoods. This total-probability structure gives at once a posterior probability density over the rules from Bayes theorem:

$$p(j|y,x) = \frac{P(Z = j, Y = y|X = x)}{p(y|x)} = \frac{p_j(x)p_{B_j}(y)}{p(y|x)} = \frac{p_j(x)p_{B_j}(y)}{\sum_{j=1}^{m} p_j(x)p_{B_j}(y)} \tag{6}$$

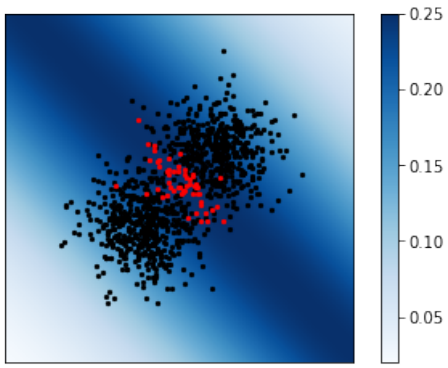

Fig. 2 Output uncertainty of the rule-foam classifier. The correctly classified points are in black. The misclassified points are in red. The background color shows the conditional variance. The color bar gives the value of the conditional variance $V[Y|X = x]$ that corresponds to the color: It is highest where the pattern classes overlap.

for the j-th rule firing given observed input x and output y. These posteriors also give the contribution of each rule to the final output.

Figure 3 shows a snapshot of the rule posteriors for a CIFAR-10 image input. The x-axis lists the rule numbers that correspond to the 7 highest posteriors. The 621st rule contributed most to the classification of the input image.

Consider a SAM classifier that misclassified the input x. The SAM identifies the rule responsible for the mistake. This makes the fuzzy rule-based system interpretable. A fuzzy system that learns from a neural classifier can interpret the neural black box. Users can also use these posteriors to later prune or modify the rule base.

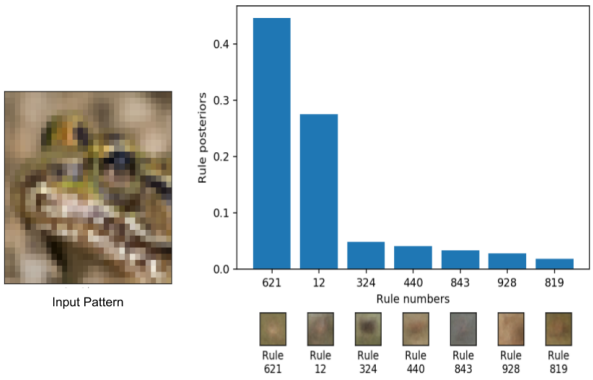

Fig. 3 Bayesian posterior $p(j|y, x)$ over the rule firings when input x produces output y. The input image is on the left. The 7 largest rule posteriors for the input are on the right. The if-part centroids of these rules are also on the right. The rule foam had 1000 rules. The x-axis lists the rule number. The y-axis lists the corresponding posterior probability of rule firing. The posterior distribution when the foam correctly classified an input from class 'Frog'. The 621st rule ($j = 621$) contributed most to the classification of this input pattern.

2.4 SAM Combination of Throughput Rules

SAM systems combine by taking sum of their then part sets. Consider q SAMs that each approximate the function f. The k-th SAM uses m_k rules and has the weight v^k. This helps us combine knowledge from multiple experts and also allows us to combine closed-form equations [19] with the softer knowledge of rules. The rules of the SAM combination also give the underlying mixture density:

$$p(y|x) = \sum_{k=1}^{q} \sum_{j=1}^{m_k} \frac{v^k w_j^k a_j^k(x) V_j^k}{\sum_{k=1}^{q} \sum_{j=1}^{m_k} v^k w_j^k a_j^k(x) V_j^k} \frac{b_j^k(y)}{V_j^k} = \sum_{k=1}^{q} \sum_{j=1}^{m_k} p_j^k(x) p_{B_j^k}(y). \quad (7)$$

The function approximation is just the conditional expectation of the mixture:

$$F(x) = E[Y|X = x] = \int y \; p(y|x) \; dy = \sum_{k=1}^{q} \sum_{j=1}^{m_k} p_j^k(x) c_j^k. \quad (8)$$

Here c_j^k is the k-th SAM's j-th then-part centroid. This SAM combination also measures the uncertainty in its output through its conditional variance $V[Y|X = x]$:

$$V[Y|X = x] = \sum_{k=1}^{q} \sum_{j=1}^{m_k} p_j^k(x) \sigma_{B_j^k}^2 + \sum_{k=1}^{q} \sum_{j=1}^{m_k} p_j^k(x) (c_j^k - F(x))^2 \quad (9)$$

where $\sigma_{B_j^k}$ is the dispersion of k-th SAM's j-th then-part fuzzy set B_j^k. The generalized mixture $p(y|x)$ in (7) also gives a finer-grained Bayesian posterior

$$p(j, k|y, x) = \frac{p_j^k(x) p_{B_j^k}(y)}{\sum_{k=1}^{q} \sum_{j=1}^{m_k} p_j^k(x) p_{B_j^k}(y)} \quad (10)$$

of the k-th SAM's j-th rule firing. Summing over m gives the posterior over the constitutive foams.

3 Fuzzy Rule Foam

A SAM fuzzy system's rules resemble bubbles in the input-output product space. The bubbles or rule patches cover the graph of the function they approximate. The SAM's graph-cover structure leads to rule explosion. Rule foam mitigates rule explosion and allows fuzzy systems to approximate high-dimensional pattern classifiers.

A classifier foam's rules do not cover the input space equally. The rule foam concentrates its rules at the class boundaries. There are a few large rules covering the class interior and a lot of smaller rules covering the class boundary. This defines a foam like structure of the rule if-part set bubbles [16]. Figure 1 shows this structure. The circles represent the if-part sets of the rule foam. The radius of the circle represents the dispersion of the if-part set. The if-part dispersion characterizes the size of the rule. Rules are smaller close to the class boundary and are larger away from the boundary. The foam also avoids covering empty regions of input space through Adaptive Vector Quantization (AVQ).

3.1 Training with Adaptive Vector Quantization (AVQ)

Adaptive vector quantization (AVQ) is a sample-based scheme for estimating an unknown data distribution [2]. We use reinforcement version of AVQ to distribute rule if-part sets. AVQ is a from of k-means clustering [7, 15] or competetive learning [4] or self-organizing maps [8]. AVQ gives Quantization Vectors (QVs) $\{\hat{x}_j\}_{j=1}^m$ whose distribution approximates that of the data set $\{x_n\}_{n=1}^N$. The AVQ algorithm cycles through the data set every epoch. It finds the QV $\hat{x}_j$ closest to the input vector x and either rewards it or punishes it by moving it either towards or away from x. AVQ moves $\hat{x}_j$ closer to the x if they belong to the same class and moves $\hat{x}_j$ away from the x if they belong to different classes.

Let $\hat{x}_j^{(t)}$ denote the j-th QV after the t-th iteration. Let $\hat{x}_j^{(t)}$ be the closest QV to the data point x_n. Let $\hat{x}_j^{(t)}$ belong to the class C_j. AVQ updates $\hat{x}_j^{(t)}$ as

$$\hat{x}_j^{(t+1)} = \hat{x}_j^{(t)} + \eta_t(x_n - \hat{x}_j^{(t)})r_j(x_n) \tag{11}$$

for decreasing learning rates η_t. The bipolar reinforcement function r_j is the indicator difference [9]

$$r_j(x_n) = \mathbb{I}_{C_j}(x_n) - \sum_{C \neq C_j} \mathbb{I}_C(x_n). \tag{12}$$

It gives $r_j(x_n) = +1$ if $x_n \in C_j$ and $r_j(x_n) = -1$ otherwise. So the jth QV looks a little more like the current sample point x_n if $r_j(x_n) = 1$ and a little less like it if $r_j(x_n) = -1$.

The distribution $\hat{x}_j$'s approximates the distribution of the input vectors. So there are no $\hat{x}_j$'s in empty regions of input space. We center the foam's if-part sets around $\hat{x}_j$'s and avoid covering the empty input space.

4 Random Foams

A random foam combines several fuzzy rule foams that train on random subsets of a data set. This method resembles how random forest combines the output of several trees [3]. The random foam performs better than the individual constituent foams.

We can combine foams by simply combining or averaging their outputs or by combining their rules or the throughputs. The additive structure of SAM systems permits this richer form of combination through (7). Random forests do not not permit such throughput combination.

Let the k-th foam F_k with m_k rule approximate f. Then (2) gives F_k as

$$F_k(x) = \sum_{j=1}^{m_k} \frac{w_j^k a_j^k(x) V_j^k}{\sum_{j=1}^{m_k} w_j^k a_j^k(x) V_j^k} c_j^k. \tag{13}$$

Averaging the foam *outputs* gives the random foam output $F_{avg}(x)$ as

$$F_{avg}(x) = \frac{1}{q} \sum_{k=1}^{q} F_k(x) \ . \tag{14}$$

We combine the throughputs of the q SAMs using equation (7). We choose weight each SAM equally by $u^k = 1/q$. Then the random foam output is

$$F_{com}(x) = \sum_{k=1}^{q} \sum_{j=1}^{m_k} p_j^k(x) c_j^k \tag{15}$$

where

$$p_j^k(x) = \frac{u^k w_j^k a_j^k(x) V_j^k}{\sum_{k=1}^{q} \sum_{j=1}^{m_k} u^k w_j^k a_j^k(x) V_j^k} = \frac{w_j^k a_j^k(x) V_j^k}{\sum_{k=1}^{q} \sum_{j=1}^{m_k} w_j^k a_j^k(x) V_j^k}. \tag{16}$$

5 Experiments on MNIST Data

We tested the random foam on the MNIST data set [14]. The MNIST data set consists of 60,000 28×28 gray-scale images of handwritten digits from 0 to 9. The random foam trained 30 rule foams on random subsets of MNIST data with bootstrap resampling. Each subset had 10,000 MNIST images. Each individual foam used 1000 rules was about 93.5% accurate. The random foam then combined these foams in both ways.

The output-averaged random foam was 96.06% accurate while the throughput-averaged random was was 96.80% accurate. We trained a random forest with 30 trees for comparison. The random forest was 96.55% accurate and thus less accurate than the throughput-combined random foam trained on the same MNIST data. Figure 4 compares the foam accuracies against the number of foams in the random foams. It also shows the performance of the random forest against the number of trees.

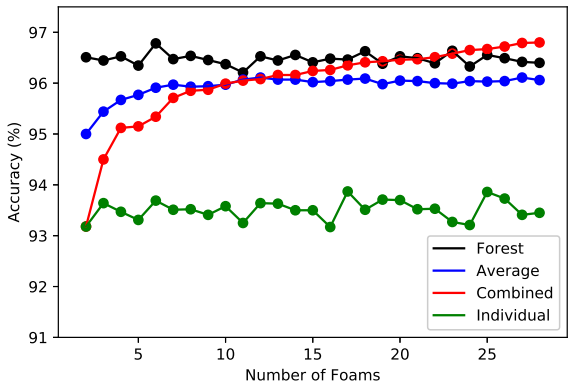

Fig. 4 Accuracy of the random foams: Random foam with throughput averaging performed best. Accuracy of the throughput-combined foam F_{com} in red. Accuracy of the output-averaged foam F_{avg} in blue. The accuracy of the individual foams F_k in green. Plot of random forest accuracy against number of trees in black.

The random foam trained by sampling from a multilayer neural classifier that itself trained on the MNIST data. The neural network had 784 identity input neurons and 10 softmax output neurons. It had two hidden layers with 256 rectified-linear-unit (ReLU) neurons each. The network trainsed on the MNIST dataset for 100 epochs and had 96.26% accuracy. The 30 1000-rule foams trained by randomly sampling 10,000 input-output pairs each from this trained neural classifier. Each individual foam was about 94.47% accurate. The random foam then averaged their outputs and their throughputs. The throughput-averaged random foam was 96.14% accurate while the output-averaged random foam was only 95.6% accurate.

Figure 5 shows the accuracy of the output-averaged random foam $F_{avg}(x)$ and the throughput-averaged random foam $F_{com}(x)$ against the number of their constituent foams. It also shows the accuracy of the neural classifier that trained the random foam and shows the accuracy of an individual foam. Figure 5 shows that the random foam performed significantly better than its constituent foams. It also shows that averaging throughputs performed better than averaging outputs. The throughput-averaged random foam was nearly as accurate as the neural classifier that trained it.

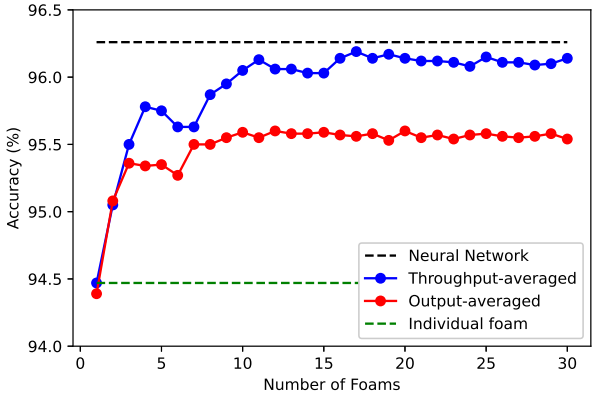

Fig. 5 Random foam trained on a neural classifier. The throughput-averaged random foam performed almost as good as the neural network that trained it. The accuracy of the neural network is in black. The accuracy of the throughput averaged random foam is in blue. The accuracy of the output-averaged random foam is in red. The accuracy of an individual constituent foam is in green.

6 Conclusions

Fuzzy rule foams are granular function approxiimators with interpretable rule structures and output uncertainties. They suffer from exponential rule explosion because their if-part fuzzy sets must cover the input pattern space. A random foam combines several independent fuzzy rule-foam approximators and performs better than the individual rule-foam approximators. The random foam acts as type of variance reduction because of the independent random sampling involved in the foam training. The random foam interprets its rules though its Bayesian posterior over the rules

and measures its output uncertainty with the conditional variance of its underlying generalized mixture. Simulations showed that random foams performed comparably on the MINIST test set when it trains on random samples from a trained deep neural classifier. The random foam that trained on the MNIST data itself had slightly better accuracy than the neural classifier. Future random foams may find more efficient if-part rule covers that mitigate rule explosion as they achieve higher accuracy.

References

1. A. Adadi and M. Berrada. Peeking Inside the Black-Box: A Survey on Explainable Artificial Intelligence (XAI). *IEEE Access*, 6:52138–52160, 2018.
2. A Buzo, RM Gray, et al. An algorithm for vector quantizer design. *IEEE Transactions on Communications*, 28(1):84–95, 1980.
3. Bradley Efron and Trevor Hastie. *Computer age statistical inference*, volume 5. Cambridge University Press, 2016.
4. Stephen Grossberg. Competitive learning: From interactive activation to adaptive resonance. *Cognitive science*, 11(1):23–63, 1987.
5. Roger Guimerà, Ignasi Reichardt, Antoni Aguilar-Mogas, Francesco A Massucci, Manuel Miranda, Jordi Pallarès, and Marta Sales-Pardo. A Bayesian machine scientist to aid in the solution of challenging scientific problems. *Science Advances*, 6(5), 2020.
6. Fabian Horst, Sebastian Lapuschkin, Wojciech Samek, Klaus-Robert Müller, and Wolfgang I Schöllhorn. Explaining the unique nature of individual gait patterns with deep learning. *Scientific Reports*, 9(1):1–13, 2019.
7. Anil K Jain. Data clustering: 50 years beyond K-means. *Pattern Recognition Letters*, 31(8):651–666, 2010.
8. Teuvo Kohonen. The self-organizing map. *Proceedings of the IEEE*, 78(9):1464–1480, 1990.
9. B. Kosko. Stochastic competitive learning. *IEEE Transactions on Neural Networks*, 2(5):522–529, 1991.
10. B. Kosko. Fuzzy Systems as Universal Approximators. *IEEE Transactions on Computers*, 43(11):1329–1333, November 1994.
11. Bart Kosko. *Fuzzy Engineering*. Prentice-Hall, 1996.
12. Bart Kosko. Additive Fuzzy Systems: From Generalized Mixtures to Rule Continua. *International Journal of Intelligent Systems*, 33(8):1573–1623, 2018.
13. Vladik Kreinovich, George C Mouzouris, and Hung T Nguyen. Fuzzy rule based modeling as a universal approximation tool. In *Fuzzy Systems*, pages 135–195. Springer, 1998.
14. Yann LeCun and Corinna Cortes. MNIST handwritten digit database. 2010.
15. JB MacQueen. Some methods for classification and analysis of multivariate observations. Western Manage-ment Sci. Inst. Univ. Technical report, of California Working Paper, 1966.
16. Akash Kumar Panda and Bart Kosko. Converting neural networks to rule foam. In *Proceedings of 2019 International Conference on Computational Science and Computational Intelligence (CSCI)*, pages 519–525. IEEE, 2019.
17. Walter Rudin. *Real and complex analysis*. McGraw-hill education, 2006.
18. Wojciech Samek. *Explainable AI: interpreting, explaining and visualizing deep learning*, volume 11700. Springer Nature, 2019.
19. Fred Watkins. The representation problem for additive fuzzy systems. In *Proceedings of the International Conference on Fuzzy Systems (IEEE FUZZ-95)*, pages 117–122, 1995.

7 Appendix : Fuzzy Approximation Theorem for a Combination of Gaussian SAMs

Theorem 1 extends a version of the fuzzy approximation theorem [10] to the case of combined additive systems. The Gaussian structure of the if-part sets gives insight

into the uniform approximation. The approximation may need ab exponentially growing total number of rules mq with respect to the number of dimensions n.

Theorem 1 *A combination of q Gaussian SAMs can uniformly approximate a continuous function f over a compact space.*

Proof We combine q SAMs to approximate $f(x)$ with $F(x)$ by mixing their then-part sets. We want to show that the approximation error obeys $|f(x) - F(x)| < \varepsilon$ for any $\varepsilon > 0$ using a finite number of rules. We use Gaussian if-part sets with set functions $a_j^k(x) = \exp\{-||x-\hat{x}_j^k||^2/(d_j^k)^2\}$. We assign equal weights, equal volumes, and equal dispersions to all the rules: $v^k = v$, $w_j^k = w$, $V_j^k = V$, and $d_j^k = d$. This reduces the generalized mixing weights $p_j(x)$ to softmax activation functions:

$$p_j^k(x) = \frac{vwa_j^k(x)V}{\sum_{k=1}^q \sum_{j=1}^m vwa_j^k(x)V} = \frac{\exp\{-||x - \hat{x}_j^k||^2/d^2\}}{\sum_{k=1}^q \sum_{j=1}^m \exp\{-||x - \hat{x}_j^k||^2/d^2\}}. \tag{17}$$

Multiply $f(x)$ by $\sum_{k=1}^q \sum_{j=1}^m p_j^k(x) = 1$ and then use (8):

$$|f(x) - F(x)| = \left| \left(\sum_{k=1}^q \sum_{j=1}^m p_j^k(x) \right) f(x) - \sum_{k=1}^q \sum_{j=1}^m p_j^k(x) c_j^k \right|. \tag{18}$$

Rearrange the terms and use the triangle inequality:

$$|f(x) - F(x)| \leq \sum_{k=1}^q \sum_{j=1}^m p_j^k(x) \left| f(x) - c_j^k \right| = \sum_{k=1}^q \sum_{j=1}^m p_j^k(x) \left| f(x) - f(\hat{x}_j^k) \right| \tag{19}$$

because we center the then-part set around $f(\hat{x}_j^k)$. So $c_j^k = f(\hat{x}_j^k)$. The target function f is uniformly continuous because it is continuous and X is compact. So for all $\varepsilon > 0$ there is a $\delta > 0$ such that $|f(x) - f(\hat{x}_j^k)| < \varepsilon/2$ if $||x - \hat{x}_j^k|| < \delta$. This splits the rules into the two sets $J_1 = \{(j,k) : ||x - \hat{x}_j^k|| < \delta\}$ and $J_2 = \{(j,k) : ||x - \hat{x}_j^k|| \geq \delta\}$. We assume that $J_1 \neq \emptyset$. Then (19) becomes

$$|f(x) - F(x)| \leq \sum_{(j,k) \in J_1} p_j^k(x) \left| (f(x) - f(\hat{x}_j^k)) \right| + \sum_{(j,k) \in J_2} p_j^k(x) \left| (f(x) - f(\hat{x}_j^k)) \right|. \tag{20}$$

Consider the first term on the right side of (20). The absolute differences obey $\left| (f(x) - f(\hat{x}_j^k)) \right| < \varepsilon/2$ because $(j,k) \in J_1$implies that $||x - \hat{x}_j^k|| < \delta$. Add these terms over all $(j,k) \in J_1$:

$$\sum_{(j,k) \in J_1} p_j^k(x) \left| (f(x) - f(\hat{x}_j^k)) \right| < \sum_{(j,k) \in J_1} p_j^k(x) \frac{\varepsilon}{2} < \frac{\varepsilon}{2} \tag{21}$$

because $\sum_{(j,k) \in J_1} p_j^k(x) \leq \sum_{k=1}^q \sum_{j=1}^m p_j^k(x) = 1$. Now pick a real number C large enough that $|f(x) - f(\hat{x}_j^k)| < C$ for all j and k. Pick $(j,k) \in J_2$ and pick $(j_1, k_1) \in J_1$. Then $||x - \hat{x}_j^k|| \geq \delta$ and $||x - \hat{x}_{j_1}^{k_1}|| < \delta$. Pick a real number D small enough that $||x - \hat{x}_j^k||^2 - ||x - \hat{x}_{j_1}^{k_1}||^2 \geq D > 0$. Choose the rule if-part dispersion d small enough

that $\exp(D/d^2) > 2mqC/\varepsilon$. Then the definition of D gives

$$\frac{\exp\left(-||x-\hat{x}_{j_1}^{k_1}||^2/d^2\right)}{\exp\left(-||x-\hat{x}_j^k||^2/d^2\right)} = \exp\left(\frac{||x-\hat{x}_j^k||^2 - ||x-\hat{x}_{j_1}^{k_1}||^2}{d^2}\right) > \frac{2mqC}{\varepsilon}. \tag{22}$$

Add $\sum_{i\neq j_1}\sum_{l\neq k_1}\frac{\exp\left(-||x-\hat{x}_i^l||^2/d^2\right)}{\exp\left(-||x-\hat{x}_j^k||^2/d^2\right)}$ both sides:

$$\frac{\sum_{j_1=1}^{m}\sum_{k_1=1}^{q}\exp\left(-||x-\hat{x}_{j_1}^{k_1}||^2/d^2\right)}{\exp\left(-||x-\hat{x}_j^k||^2/d^2\right)} > \frac{2mqC}{\varepsilon} + \frac{\sum_{i\neq j_1}\sum_{l\neq k_1}\exp\left(-||x-\hat{x}_i^l||^2/d^2\right)}{\exp\left(-||x-\hat{x}_j^k||^2/d^2\right)} > \frac{2mqC}{\varepsilon} \tag{23}$$

because $\sum_{i\neq j_1}\sum_{l\neq k_1}\frac{\exp\left(-||x-\hat{x}_i^l||^2/d^2\right)}{\exp\left(-||x-\hat{x}_j^k||^2/d^2\right)} > 0$. Equation (17) and the definition of C give

$$p_j^k(x)\left|f(x) - f(\hat{x}_j^k)\right| < \frac{\varepsilon}{2mqC}\left|f(x) - f(\hat{x}_j)\right| < \frac{\varepsilon}{2mq}. \tag{24}$$

Sum both sides over all $(j,k) \in J_2$:

$$\sum_{(j,k)\in J_2} p_j^k(x)|f(x) - f(\hat{x}_j^k)| < \sum_{(j,k)\in J_2}\frac{\varepsilon}{2mq} \leq mq\frac{\varepsilon}{2mq} = \frac{\varepsilon}{2} \tag{25}$$

because J_2 has fewer than mq elements. Inequalities (21) and (25) add to give the approximation error $|f(x) - F(x)| \leq \varepsilon/2 + \varepsilon/2 < \varepsilon$. So the combined SAM with mq rules uniformly approximates f. □

The proof above assumes that $J_1 \neq \emptyset$. So for all $x \in X$ there is a $\hat{x}_j^k$ such that $||x - \hat{x}_j|| < \delta$. Let $N_\delta(\hat{x}_j^k) = \{x : ||x - \hat{x}_j^k|| < \delta\}$ denote the δ-neighborhood around $\hat{x}_j^k$. Then $\forall\, x \in X :\ \exists\, \hat{x}_j^k : x \in N_\delta(\hat{x}_j^k)$. This gives

$$X \subseteq \bigcup_{k=1}^{q}\bigcup_{j=1}^{m} N_\delta(\hat{x}_j^k). \tag{26}$$

X is bounded because X is compact by hypothesis. Let an n-dimensional sphere of radius R contain X. The volume of this sphere is cR^n for some constant c. Each $N_\delta(\hat{x}_j^k)$'s is an n-dimensional sphere with radius δ and volume $c\delta^n$. Use the δ-neighborhoods to cover the R-sphere that contains X. The $N_\delta(\hat{x}_j^k)$'s are not disjoint. So the volume of the R-sphere is less than the volume of the mq $N_\delta(\hat{x}_j^k)$'s combined: $cR^n \leq mqc\delta^n$. Then $mq \geq (R/\delta)^n$. This inequality shows that the required total number of rules grows exponentially with the dimension n.