Bayesian Pruned Random Rule Foams for XAI

Akash Kumar Panda
Signal and Image Processing Institute
Department of Electrical and Computer Engineering
University of Southern California
Los Angeles, California

Bart Kosko
Signal and Image Processing Institute
Department of Electrical and Computer Engineering
University of Southern California
Los Angeles, California
kosko@usc.edu

***Abstract*—A random rule foam grows and combines several independent fuzzy rule-based systems by randomly sampling input-output data from a trained deep neural classifier. The random rule foam defines an interpretable proxy system for the sampled black-box classifier. The random foam gives the complete Bayesian posterior probabilities over the foam subsystems that contribute to the proxy system's output for a given pattern input. It also gives the Bayesian posterior over the if-then fuzzy rules in each of these constituent foams. The random foam also computes a conditional variance that describes the uncertainty in its predicted output given the random foam's learned rule structure. The mixture structure leads to bootstrap confidence intervals around the output. Using the Bayesian posterior probabilities to prune or discard low-probability sub-foams improves the system's classification accuracy. Simulations used the MNIST image data set of 60,000 gray-scale images of ten hand-written digits. Dropping the lowest-probability foams per input pattern brought the pruned random foam's classification accuracy nearly to that of the neural classifier. Posterior pruning outperformed simple accuracy pruning of a random foam and outperformed a random forest trained on the same neural classifier.**

***Index Terms*—XAI, additive fuzzy systems, rule foam, generalized mixtures, Bayesian rule posteriors**

I. XAI with Random Rule Foams

We show how to use the inherent Bayesian posterior probabilities of combined fuzzy rule-based systems to improve classification accuracy. This approach gives an explainable system that approximates a sampled neural classifier. An additive fuzzy system F sums and averages its m fired if-then rules $R_{A_1 \to B_1}, \ldots, R_{A_m \to B_m}$ for each vector input x. The m rules define a probability mixture $p(y|x)$ that mixes m likelihoods [16]: $p(y|x) = p_1(x)p_{B_1}(y|x) + \cdots + p_m(x)p_{B_m}(y|x)$ in accord with (1) - (2) below.

The first moment of the mixture $p(y|x)$ gives the fuzzy system itself as $F(x) = E[Y|X = x]$ per (3). The mixture structure gives each fuzzy subsystem its own output conditional variance as well as a total system conditional variance. The mixture also defines bootstrap confidence intervals around the system output that further describe its uncertainty. The result is a new and modular form of explainable AI (XAI) [1], [4], [5], [8], [20], [22], [26], [29], [30].

Each rule foam system is an additive fuzzy system F of m if-then rules $R_{A_j \to B_j}$ and maps pattern inputs x to output pattern classifications $F(x)$ [13], [14], [17]. The system F is again just the first moment (3) of the mixture $p(y|x)$. A random foam combines several independent foams by randomly sampling with replacement (bootstrap sampling) from a trained deep neural classifier. Adaptive vector quantization forms and tunes the rules of each foam. The additive structure allows the random foam to combine the constituent foams by combining each foam's rules rather than just by combining each foam's output. The random rule foam acts as an interpretable *proxy* system for the sampled classifier.

Figure 1 shows an accuracy-pruned random foam that approximates a deep neural classifier. The neural classifier trained on the MNIST dataset of the ten handwritten digits $0, 1, \ldots, 9$. It had a 96.62% classification accuracy. The random foam trained by randomly drawing bootstrap input-output pairs from the neural classifier without replacement. The resampling-trained random foam had 24 constituent foams and was 95.95% accurate. Accuracy-based pruning permanently removed the 13 foams that had the lowest accuracy. The pruned random foam with the remaining 11 foams was 96.11% accurate.

Posterior-based pruning performed better than accuracy-based pruning. Posterior pruning removed the 13 sub-foams with the smallest posterior probabilities from the combined foam and did so for *each input*. It did not permanently remove these sub-foams as with accuracy pruning. It removed them only for the image-pattern input x and then tested on x. This pruned random foam was 96.27% accurate. Its average accuracy rose to 96.52% when it pruned the 20 smallest-posterior-value foams per iteration.

II. Additive Rule Foams

A rule foam's rules resemble bubbles in the input-output product space that cover the graph of the function they approximate [23]. 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 avoids covering empty regions of the input space through Adaptive Vector Quantization (AVQ).

The SAM's graph covering structure leads to rule explosion. Rule foam mitigates rule explosion and allows fuzzy systems to approximate high-dimensional pattern classifiers. The 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

978-1-6654-4407-1/21/$31.00 © 2021 IEEE

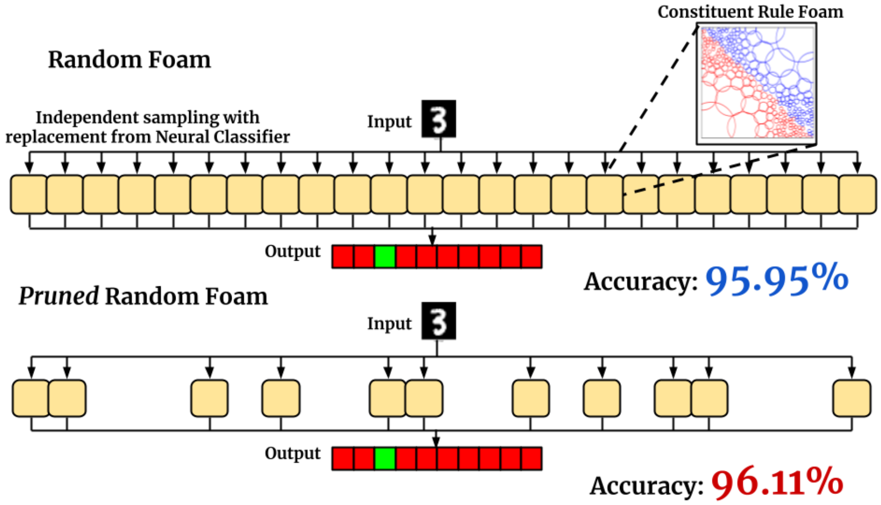

Fig. 1. Random foam pruning. The top image shows a random foam that consists of 24 independent constituent foams. Each constituent foam trains by random sampling with replacement from a neural classifier. The image on the right zooms in on one of the constituent foams and shows the foam-like structure of its if-part fuzzy sets. This random foam classified the input MNIST pattern with 95.95% accuracy. The bottom image shows a pruned random foam. This algorithm pruned 13 foams based on their accuracy. This pruned random foam was 96.11% accurate.

smaller rules covering the class boundary. This defines a foam like structure of the rule if-part set bubbles.

A generalized mixture underlies rule foam. This allows rule foam to measure the uncertainty in its output through conditional variance. This also give a Bayesian posterior distribution over the rules that shows the contribution of each rule to the output. These posteriors may also be used to prune rules. Foams can also combine to give random foams.

A. Generalized Mixture $p(y|x)$ for a SAM Fuzzy System

Standard Additive Model (SAM) fuzzy system $F : \mathbb{R}^d \rightarrow \mathbb{R}$ takes input pattern vector x and then sums and averages its m if-then fuzzy rules $R_{A_1 \rightarrow B_1}, \ldots, R_{A_m \rightarrow B_m}$ to produce output $F(x)$ [14], [17]. The jth rule $R_{A_j \rightarrow B_j}$ has fuzzy if-part set $A_j \subset \mathbb{R}^d$ and fuzzy then-part set $B_j \subset \mathbb{R}$ with respective multi-valued indicator functions $a_j : \mathbb{R}^d \rightarrow [0, 1]$ and $b_j : \mathbb{R} \rightarrow [0, 1]$ 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 the set function $b_j(y|x) = a_j(x)b_j(y)$ because the system is standard. The total firing set $B(y|x)$ sums and weights the fired rules to give $b(y|x) = w_1 b_1(y|x) + \cdots + w_m b_m(y|x)$ for rule weights $w_j \geq 0$. Then the SAM's rules define the generalized probability mixture [16]

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

$$= \sum_{j=1}^{m} p_j(x) p_{B_j}(y) \quad (2)$$

where $V_j = \int b_j(y)dy$ is the finite volume of then-part set B_j. This mixture structure does *not* arise from the min-max structure of earlier non-additive systems [27], [32].

The fuzzy system F arises naturally as the first non-central moment $E[Y|X = x]$ of the mixture $p(y|x)$:

$$F(x) = E[Y|X = x] = \int y \, p(y|x)dy = \sum_{j=1}^{m} p_j(x)c_j \quad (3)$$

where c_j is the centroid of the jth then-part set B_j: $c_j = \int y \, p_{B_j}(y)dy$. The output conditional variance $V[Y|X = x]$ in (7) is just the second moment of $p(y|x)$.

An additive fuzzy system is an universal function approximator [13], [14], [18]. It can uniformly approximate any continuous function f on a compact domain using finite rules even though this may involve exponential rule explosion [15].

B. Adaptive Vector Quantization (AVQ)

Adaptive Vector Quantization (AVQ) is a sample based scheme for estimating an unknown data distribution [3]. We use reinforcement version of AVQ to distribute rule if-part sets. AVQ is a from of k-means clustering [10], [21] or competitive learning [9] or self-organizing maps [11]. 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 jth QV after the tth 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

$$\hat{x}_j^{(t+1)} = \hat{x}_j^{(t)} + \eta_t(x_n - \hat{x}_j^{(t)})r_j(x_n) \quad (4)$$

for decreasing learning rates η_t. The bipolar reinforcement function r_j is the indicator difference [12]

$$r_j(x_n) = \mathbb{I}_{C_j}(x_n) - \sum_{C \neq C_j} \mathbb{I}_C(x_n). \tag{5}$$

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.

C. Rule Importance through Bayesian Posteriors

The generalised mixture in equation (2) is a form of total probability. The mixing weights $p_j(x)$ define prior probabilities and the $p_{B_j}(y)$ define likelihoods. Then the theorem on total probability gives the Bayes posteriors

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

of the jth rule firing. These posteriors also give the contribution of each rule to the final output. Figure 2 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.

Suppose a foam misclassifies the input x. The foam still shows the rule most responsible for the mistake. This makes the fuzzy rule-based system interpretable. Users can also use these posteriors to later prune or modify the rule base.

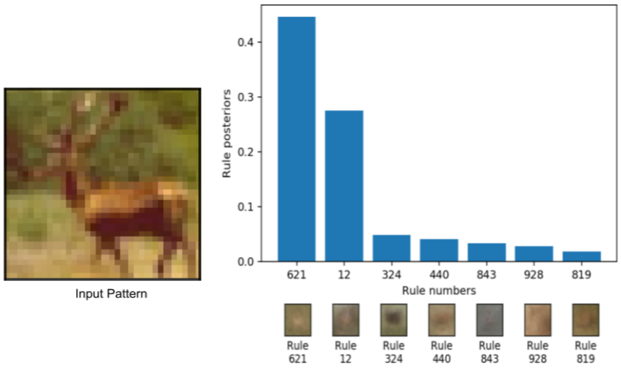

Fig. 2. Bayesian posterior $p(j|y,x)$ over the fired rules when input pattern x produced output y. The input image is on the left. The 7 largest rule-posterior values for the input are on the right. The if-part set centroids of these rules are also on the right. The rule foam had 1000 rules. The x-axis lists the rule number. The y-axis lists the corresponding posterior probability of rule firing. The histogram shows the posterior density when the foam correctly classified an input from class 'Deer'. The 621st rule ($j = 621$) contributed the most to classifying this input pattern.

D. System Confidence though Conditional variance

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

$$V[Y|X = x] = \sum_{j=1}^m p_j(x) \sigma_{B_j}^2 + \sum_{j=1}^m p_j(x)(c_j - F(x))^2 \tag{7}$$

where σ_{B_j} is the jth rule's then-part dispersion. The second term in (7) 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 foam that approximates the simple function f representing a classifier. A simple function maps a space to a finite number of values [25]. 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 also higher in regions of high conditional variance. Conditional variance is a good measure of system confidence.

This feature is absent in the neural classifiers. Foams that train on a neural network's output can measure the uncertainty in the network's classification.

Figure 3 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.

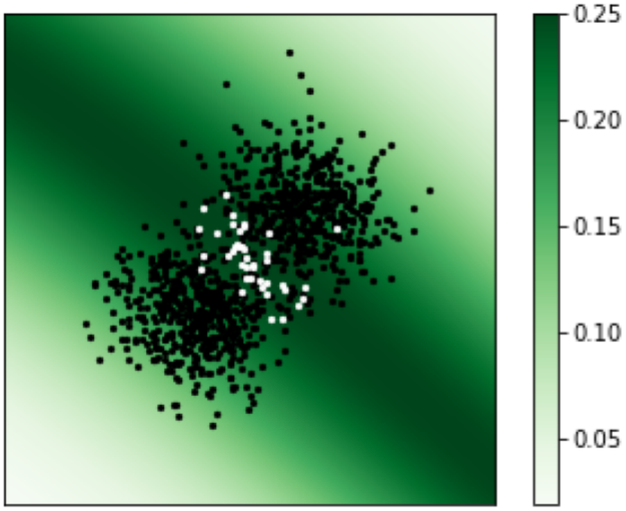

Fig. 3. Output uncertainty of the rule-foam classifier. The correctly classified points are in black. The misclassified points are in white. The background color shows the conditional variance. The color bar gives the value of $V[Y|X = x]$ that corresponds to the color: The variance is highest where the pattern classes overlap.

III. Random Foams

A random foam combines several fuzzy rule foams that train on random subsets of a data set. This method resembles how a random forest combines the output of several trees [2], [7], [28] but the combination technique differs because it combines throughput rules or mixtures. The random foam also performs better than its constituent foams.

A. *Foam Combination*

Foams combine by taking the sum of their then-part sets. Consider q foams that each approximate the function f. The kth SAM uses m_k rules and has weight v^k. Then we can combine knowledge from multiple experts and also combine closed form knowledge [16], [31] with the soft knowledge. The additive structure of a SAM also give a governing generalized mixture $p(y|x)$:

$$p(y|x) = \sum_{k=1}^{q} \sum_{j=1}^{m_k} \frac{v^k w_j^k a_j^k(x) V_j^k}{\sum_{k=1}^{q} \sum_{j=1}^{m_k} v^k w_j^k a_j^k(x) V_j^k} \frac{b_j^k(y)}{V_j^k} \tag{8}$$

$$= \sum_{k=1}^{q} \sum_{j=1}^{m_k} p_j^k(x) p_{B_j^k}(y). \tag{9}$$

The first moment of $p(y|x)$ again gives the system F:

$$F(x) = E[Y|X=x] = \sum_{k=1}^{q} \sum_{j=1}^{m_k} p_j^k(x) c_j^k \tag{10}$$

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

$$V[Y|X=x] = \sum_{k=1}^{q} \sum_{j=1}^{m_k} p_j^k(x) \sigma_{B_j^k}^2 + \sum_{k=1}^{q} \sum_{j=1}^{m_k} p_j^k(x)(c_j^k - F(x))^2 \tag{11}$$

where $\sigma_{B_j^k}^2$ is the variance of the kth SAM's jth then-part set. This mixture $p(y|x)$ gives the telescoped Bayesian posterior

$$p(j,k|y,x) = \frac{p_j^k(x) p_{B_j^k}(y)}{\sum_{k=1}^{q} \sum_{j=1}^{m_k} p_j^k(x) p_{B_j^k}(y)} \tag{12}$$

of the kth SAM's jth rule firing.

We combine the foams in two ways. The first and older way combines their outputs. The second way combines the rules or *throughputs* using (9). Random forests do not allow such throughput combination because of their tree structure.

B. *Combining Outputs and Throughputs*

Let the k-th foam approximate f with F_k using m_k rules. Then equation (3) gives the function approximation F_k:

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

Then the average of the individual foam outputs gives the random foam output $F_{avg}(x)$:

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

We combine the throughputs of the q SAMs using equation (9). We choose weight each SAM equally by $u^k = 1/q$. So 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

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

$$= \frac{w_j^k a_j^k(x) V_j^k}{\sum_{k=1}^{q} \sum_{j=1}^{m_k} w_j^k a_j^k(x) V_j^k}. \tag{17}$$

C. *Telescoping posteriors and variance*

A random foam trains and combines several independent foams. It measures its uncertainty through its conditional variance in (11) and perhaps through other higher-order moments. It also measures the confidence of each constituent foam in their outputs through (7). Random foam gives the contribution of all the rules through the Bayesian posteriors in (12). It also measures the contribution of each constituent foam through a posterior distribution over all the foams:

$$p(k|y,x) = \sum_{j=1}^{m_k} p(j,k|y,x) = \frac{\sum_{j=1}^{m_k} p_j^k(x) p_{B_j^k}(y)}{\sum_{k=1}^{q} \sum_{j=1}^{m_k} p_j^k(x) p_{B_j^k}(y)} \tag{18}$$

Figure 4 shows an example of these telescoping posteriors for a 10-foam random foam trained on the MNIST dataset.

D. *Bootstrap Confidence Intervals for Random Foam Outputs*

The random foam's mixture $p(y|x)$ can also describe the uncertainty in the scalar output y through a bootstrap confidence interval for a given input x. Independently sample $\{y_i\}_{i=1}^n$ from $p(y|x)$ for a given x. Sort these samples to get the increasing sequence of order statistics

$$y_{(1)},\ y_{(2)},\ \cdots,\ y_{(n)} \tag{19}$$

where $y_{(i)}$ is the ith smallest sample. Then the $(1-\alpha)\%$ bootstrap confidence interval is

$$(y_{(n\alpha/2)},\ y_{(n(1-\alpha/2))}). \tag{20}$$

We are $(1-\alpha)\%$ confident that $f(x)$ lies inside this interval.

A K-dimensional vector function f has K scalar component functions f_k. K random foams can respectively approximate its K component scalar functions. So we can give a bootstrap confidence interval for each component y_k of the vector output y.

IV. Pruning Foams

A random foam combines the throughputs of several independent foams. The foams do not contribute equally to a given output. Some foams tend to harm the performance. They tend to have low accuracy on the dataset. They may also have a low foam-posterior for the given input-output pair (x,y). Pruning such foams reduces the number of parameters and may also increase the performance.

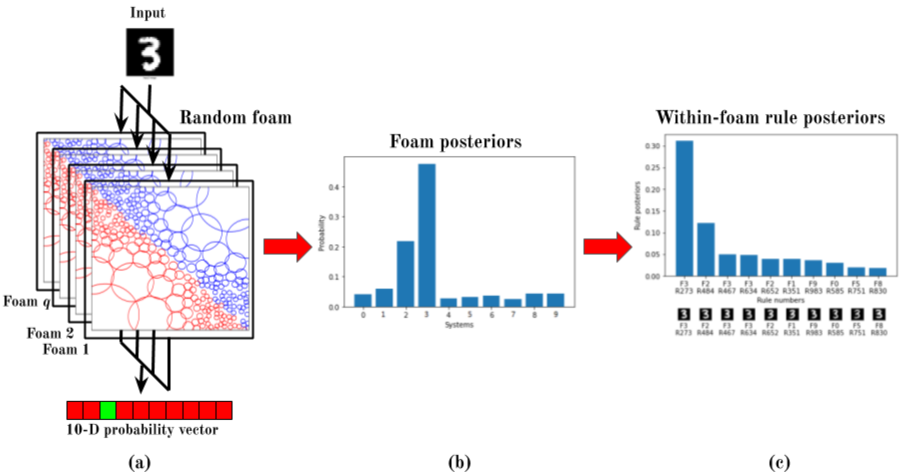

Fig. 4. Telescoping Bayesian posteriors in a random foam. (a) The random foams trained 10 independent foams and then combined them. The random foam then correctly classified the input pattern from class '3'. (b) The random foam give a Bayesian posterior over all its constituent foams for this input. The foam numbers lie along x-axis and their probabilities lie along the y-axis. Foam 3 contributed most to the classification. (c) The random foam also gave a posterior density over the rules present inside the foams. The image shows the 10 rules with the highest posterior probabilities. The x-axis lists the foam numbers and the rule numbers. The y-axis lists their probability. 'F3 R273' refers to the 273rd rule in the 3rd foam. This rule contributed the most to the classification among all the Foam-3 rules. The rule if-part centroids appear as the images below their posteriors.

A. *Accuracy based pruning*

We sort the constituent foams based on their accuracy and then prune the least accurate foams. This process is similar to the overproduce-and-choose technique in ensemble learning [24]. Slight pruning gets rid of the inaccurate foams and slightly boosts the accuracy. So the random foam accuracy increases with slight pruning and then falls as the pruning continues.

B. *Bayesian Posterior based Pruning*

A constituent foam's contribution to the output varies with each input. The foam posterior in (12) gives this contribution. A foam that does not contribute much to the classification of input x_1 may contribute a lot to classification of a different input x_2. Each input x gives a different list of foams to be pruned based on the Bayesian posteriors. Posterior-based pruning prunes the random foam differently for each input pattern while accuracy-based pruning removes the same set of foams for every input. This process is similar to the *dynamic* overproduce-and-choose technique in ensemble learning [6]. Random forests do not allow this kind of pruning because they do not have a posterior structure over the trees [7].

Posterior-based pruning algorithm calculates the foam posteriors for each input. It prunes the foams with lowest posteriors for each input. This method does not reduce the number of parameters because each foam may be used for some input pattern. But this method does increase the accuracy beyond accuracy-based pruning.

V. Experiments with MNIST dataset

We tested the random foam on the MNIST data set [19]. 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 5 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.

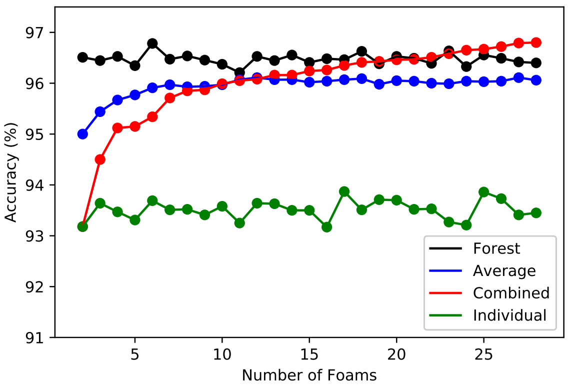

Fig. 5. Accuracy of the random foams: Random foam with *throughput* averaging performed best. Accuracy of the throughput-combined foam F_{com} in red. Accuracy of the output-averaged foam F_{avg} in blue. The accuracy of the individual foams F_k in green. The accuracy of the random forest in black.

The same 96.62% neural classifier trained a 95.90% accurate random foam with 24 constituent foams. The accuracy rose to 96.03% when the five least accurate foams were pruned. Accuracy decreased with further pruning. Figure 6 shows this trend in accuracy upon pruning.

The pruned foam's accuracy continued to rise with posterior-based pruning. The accuracy rose to 96.52% when 20 foams with lowest posteriors were dropped for each input. Posterior-based pruning brought the pruned foam's accuracy close to the neural classifier that the random foam trained on.

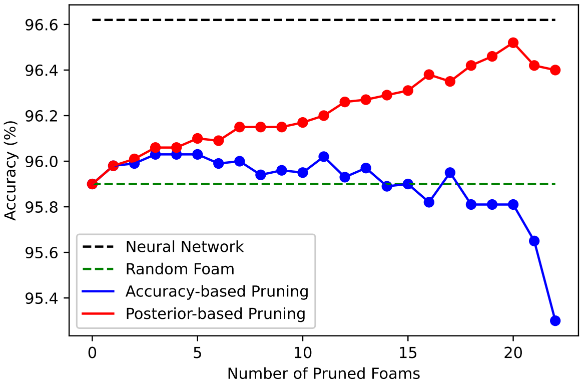

Fig. 6. Pruning effect on random-foam accuracy. The x-axis lists the number of pruned foams. The y-axis lists the accuracy of the corresponding pruned random foam. Slight pruning increased accuracy but too much pruning decreased accuracy. The red dotted line shows the accuracy of the random foam before pruning.

VI. Conclusion

Careful pruning can improve a random rule foam because it consists of so many constituent foam systems. A large-scale random foam can combine hundreds or thousands of these rule-foam subsystems. The Bayesian posteriors in each random foam gives a natural and sample-by-sample measure of the relative importance of the constituent foam systems and of their rules. Dropping low-posterior foams for a given pattern input increased the pruned random foam's accuracy and brought it closer to the accuracy of the underlying classifier. Such Bayesian posterior pruning also outperformed permanently pruning the random foam based on just the accuracy of each foam subsystem. Future pruning schemes can combine this posterior information with output variances and other foam or classifier properties.

References

[1] Franz Baader, Stefan Borgwardt, and Rafael Penaloza. Decidability and complexity of fuzzy description logics. *KI-Künstliche Intelligenz*, 31(1):85–90, 2017.

[2] Piero Bonissone, José M Cadenas, M Carmen Garrido, and R Andrés Díaz-Valladares. A fuzzy random forest. *International Journal of Approximate Reasoning*, 51(7):729–747, 2010.

[3] A Buzo, RM Gray, et al. An algorithm for vector quantizer design. *IEEE Transactions on Communications*, 28(1):84–95, 1980.

[4] Oana Cocarascu, K Cyras, and F Toni. Explanatory predictions with artificial neural networks and argumentation. 2018.

[5] Faiyaz Doctor, Hani Hagras, Victor Callaghan, and Antonio Lopez. An adaptive fuzzy learning mechanism for intelligent agents in ubiquitous computing environments. In *Proceedings World Automation Congress, 2004.*, volume 16, pages 101–106. IEEE, 2004.

[6] Eulanda M Dos Santos, Robert Sabourin, and Patrick Maupin. A dynamic overproduce-and-choose strategy for the selection of classifier ensembles. *Pattern recognition*, 41(10):2993–3009, 2008.

[7] Bradley Efron and Trevor Hastie. *Computer age statistical inference*, volume 5. Cambridge University Press, 2016.

[8] André S Fialho, Uzay Kaymak, Rui Jorge Almeida, Federico Cismondi, Susana M Vieira, Shane R Reti, João MC Sousa, and Stan N Finkelstein. Probabilistic fuzzy prediction of mortality in intensive care units. In *2012 IEEE International Conference on Fuzzy Systems*, pages 1–8. IEEE, 2012.

[9] Stephen Grossberg. Competitive learning: From interactive activation to adaptive resonance. *Cognitive science*, 11(1):23–63, 1987.

[10] Anil K Jain. Data clustering: 50 years beyond K-means. *Pattern Recognition Letters*, 31(8):651–666, 2010.

[11] Teuvo Kohonen. The self-organizing map. *Proceedings of the IEEE*, 78(9):1464–1480, 1990.

[12] B. Kosko. Stochastic competitive learning. *IEEE Transactions on Neural Networks*, 2(5):522–529, 1991.

[13] B. Kosko. Fuzzy Systems as Universal Approximators. *IEEE Transactions on Computers*, 43(11):1329–1333, November 1994.

[14] Bart Kosko. *Neural Networks and Fuzzy Systems*. Prentice-Hall, 1991.

[15] Bart Kosko. Optimal fuzzy rules cover extrema. *International Journal of Intelligent Systems*, 10(2):249–255, 1995.

[16] Bart Kosko. Additive Fuzzy Systems: From Generalized Mixtures to Rule Continua. *International Journal of Intelligent Systems*, 33(8):1573–1623, 2018.

[17] Bart Kosko. Convergence of generalized probability mixtures that describe adaptive fuzzy rule-based systems. In *2020 IEEE International Conference on Fuzzy Systems (FUZZ-IEEE)*, pages 1–8. IEEE, 2020.

[18] 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.

[19] Yann LeCun and Corinna Cortes. MNIST handwritten digit database. 2010.

[20] Thomas Lukasiewicz. Probabilistic logic programming. In *ECAI*, pages 388–392, 1998.

[21] JB MacQueen. Some methods for classification and analysis of multivariate observations. Western Manage-ment Sci. Inst. Univ. Technical report, of California Working Paper, 1966.

[22] Miguel A Olivares-Mendez, Somasundar Kannan, and Holger Voos. Vision based fuzzy control autonomous landing with uavs: From v-rep to real experiments. In *2015 23rd Mediterranean Conference on Control and Automation (MED)*, pages 14–21. IEEE, 2015.

[23] 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.

[24] Derek Partridge and William B Yates. Engineering multiversion neural-net systems. *Neural computation*, 8(4):869–893, 1996.

[25] Walter Rudin. *Real and complex analysis*. McGraw-hill education, 2006.

[26] Cátia M Salgado, Carlos S Azevedo, Jonathan Garibaldi, and Susana M Vieira. Ensemble fuzzy classifiers design using weighted aggregation criteria. In *2015 IEEE International Conference on Fuzzy Systems (FUZZ-IEEE)*, pages 1–5. IEEE, 2015.

[27] Toshiro Terano, Kiyoji Asai, and Michio Sugeno. *Fuzzy systems theory and its applications*. Academic Press Professional, Inc., 1992.

[28] Krzysztof Trawiński, Oscar Cordon, and Arnaud Quirin. On designing fuzzy rule-based multiclassification systems by combining furia with bagging and feature selection. *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, 19(04):589–633, 2011.

[29] Jasper van der Waa, Elisabeth Nieuwburg, Anita Cremers, and Mark Neerincx. Evaluating XAI: A comparison of rule-based and example-based explanations. *Artificial Intelligence*, 291:103404, 2021.

[30] Christian Wagner and Hani Hagras. Uncertainty and type-2 fuzzy sets and systems. In *2010 UK Workshop on Computational Intelligence (UKCI)*, pages 1–5. IEEE, 2010.

[31] 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.

[32] Hans-Jürgen Zimmermann. *Fuzzy set theory—and its applications*. Springer Science & Business Media, 2011.