Neuro-fuzzy logic for parts-based reasoning about complex scenes in remotely sensed data

Blake Ruprecht[a], Charlie Veal[a], Al Cannaday[a,b], Derek T. Anderson[a], Fred Petry[c], James Keller[a], Grant Scott[a], Curt Davis[a,b], Charles Norsworthy[c], Paul Elmore[d], Kristen Nock[e], and Elizabeth Gilmour[e]

[a]Electrical Engineering and Computer Science Dept, University of Missouri, Columbia, MO
[b]Center for Geospatial Intelligence, University of Missouri, Columbia, MO
[c]U.S. Naval Research Laboratory, Stennis Space Center, Stennis, MS
[d]Johns Hopkins Applied Physics Laboratory, Laurel, MD
[e]U.S. Naval Research Laboratory, Washington, DC

ABSTRACT

In this article, we explore the role and usefulness of neuro-fuzzy logic in the context of automatically reasoning under uncertainty about complex scenes in remotely sensed data. Specifically, we consider a first order Takagi-Sugeno-Kang (TSK) adaptive neuro-fuzzy inference system (ANFIS). First, we explore the idea of embedding an experts knowledge into ANFIS. Second, we explore the augmentation of this knowledge via optimization relative to training data. The aim is to explore the possibility of transferring then improving domain performance on tedious but important and challenging tasks. This route was selected, versus the popular modern thinking of learning a neural solution from scratch in an attempt to maintain interpretability and explainability of the resultant solution. An additional objective is to observe if the machine learns anything that can be returned to the human to improve their individual performance. To this end, we explore the task of detecting construction sites, an abstract concept that has a large amount of inner class variation. Our experiments show the usefulness of the proposed methodology and it sheds light onto future directions for neuro-fuzzy computing, both with respect to performance, but also with respect to glass box solutions.

Keywords: neural network, adaptive neuro-fuzzy inference system, anfis, fuzzy logic, remote sensing

Acknowledgement: Petry, Norsworthy, Nock and Gilmour would like to thank the Naval Research Laboratory's Base Program for sponsoring this research.

1. INTRODUCTION

In the last decade, we have witnessed exponential growth of research and disparate applications of machine learning (ML) and artificial intelligence (AI). However, this advancement, primarily in performance according to standard measures like F1, has arose off the back of black box AI-ML, to varying degrees. While a success in and of itself, this arguably generates more problems than solutions. One conundrum is trust and accountability in these machines derived from data and the decisions they make, which falls under the umbrella of explainable AI (XAI). Another topic is domains where expert knowledge exists, perhaps exacerbated by low data volume and/or variety. In such a scenario, it might not be possible to learn a quality data-driven alone solution. In this article, we explore the concept of encoding expert human knowledge into a neural logic system. This allows for knowledge transfer, it maintains explainability and interpretability, and it provides a starting point for augmenting the solution via optimizing with respect to data. The question then pivots to do we see improvement and what logic was responsible for that gain? As a final payoff, any knowledge discovery can be returned to the expert to improve their knowledge and performance on a domain.

The contributions of this article are as follows. First, we discuss high-level benefits and drawbacks, with respect to XAI, of an adaptive neuro-fuzzy inference system (ANFIS) to machine reason about remotely sensed

Further author information: (Send correspondence to Blake Ruprecht)
Blake Ruprecht: E-mail: bcrf53@mail.missouri.edu

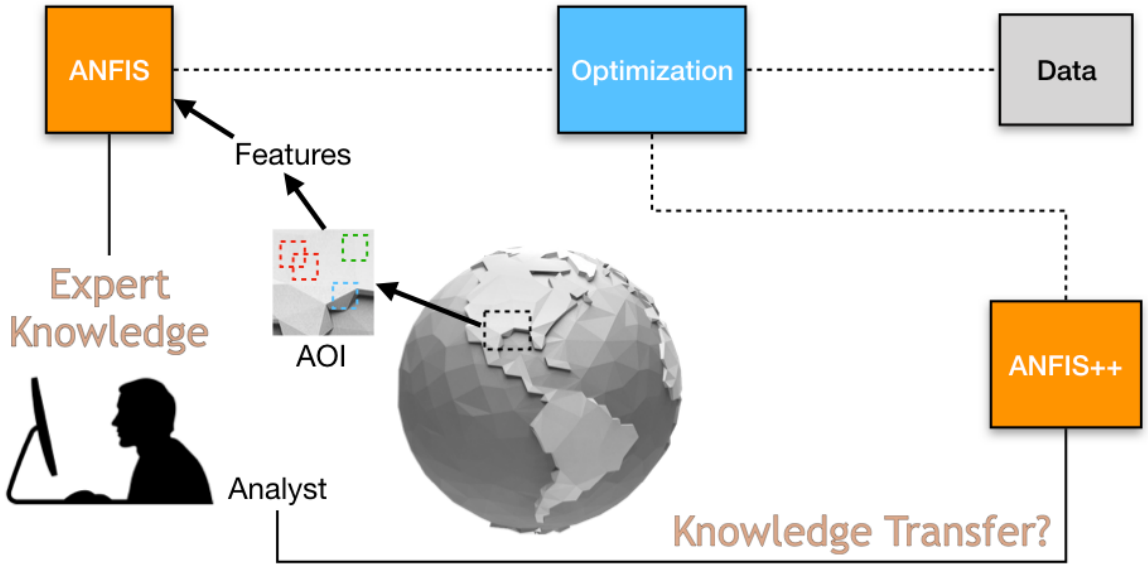

Figure 1. High-level illustration of this article. First, expert knowledge is transferred into an adaptive neuro-fuzzy inference system (ANFIS) for sake of automating some process, e.g., object detection or land classification in remote sensing. Next, data is used and the solution is optimized to produce an augmented ANFIS, "ANFIS++". The ANFIS++ is used in place of the expert and it is analyzed to determine differences for the sake of discovering new domain specific logic that might be of interest to the expert and/or analyzed for validation of the machine learned model.

data. Second, we discuss how to import human knowledge into ANFIS. Third, we outline its optimization for augmented reasoning, which we call Human Augmentation. Fourth, we explore and learn from these ideas on a specific application, classification of complex and vague constructions sites. While a contribution in and of itself, this work sets us up to seek better neuro-fuzzy solutions and human centered ML-AI.

The remainder of this article is organized as follows. In Section 2, we discuss traditional ANFIS. Section 3 discusses initialization of ANFIS, Section 4 is different ANFIS membership functions, Section 5 is "types" of rules that can be learned, Section 6 is the construction site application and experiments, followed by a summary and future work. Figure 1 illustrates the main concepts of our article and their connectivity.

2. ADAPTIVE NEURO-FUZZY INFERENCE SYSTEM (ANFIS)

Jang introduced ANFIS in,[1] which was initially based on TSK type fuzzy inference.[2] Figure 2 illustrates the flow of data of ANFIS. Let a training dataset have dimensionality, $N \times K$. While it is possible to support batch and mini-batch processing in contexts like gradient descent-based optimization, herein we focus on sample-by-sample processing for sake of readability. Let $\mathbf{x}_n(k)$ denote the k-th feature of sample n, let ANFIS consist of R rules, and let $y_n \in \Re$ be the output of ANFIS. ANFIS performs three steps on $\mathbf{x}_n$ to determine y_n. The first two steps, the Antecedent Firing and the Consequent Component Building, can be run in parallel, and they produce an antecedent vector, $\mathbf{w}_n$, and consequent weighting vector, $\mathbf{z}_n$, respectively. The third step, Aggregation, takes $\mathbf{w}_n$ and $\mathbf{z}_n$, performs a weighted sum, and it produces the final output, y_n. The lengths of $\mathbf{w}_n$ and $\mathbf{z}_n$ are R. The rules in ANFIS follow the familiar *IF-THEN* format, each of which has an antecedent and consequent clause.

2.1 ANFIS Equations

The following are the equations that delineate the forward ANFIS system.

Antecedent Firing: consists of calculating the firing strength of all the antecedent clauses of each rule, w_n^r, which uses a t-norm (herein, we use the product operator) of the membership values, $A_k^r(\cdot)$, of each input feature, i.e.,

$$w_n^r = \prod_{k=1}^{K} A_k^r(\mathbf{x}_n(k)). \tag{1}$$

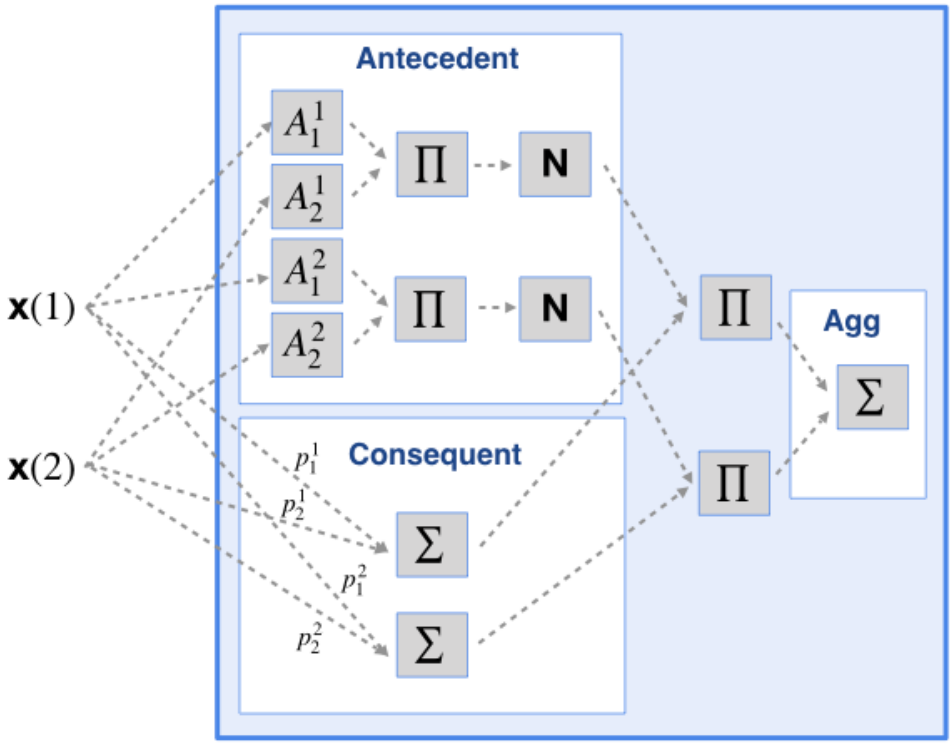

Figure 2. This figure illustrates the flow of data in a first order TSK ANFIS for the case of two inputs and two rules.

The membership functions $A_k^r(\cdot)$ are unique to each rule and feature, and can be learned. This step can be thought of as how well the input vector matches each individual rule.

Consequent Component Building: consists of calculating the components of the consequent clause of each rule, z_n^r, which is based on the summation of each input feature and weight p_k^r, plus a bias term, p_{bias}^r, i.e.,

$$z_n^r = \left(\sum_{k=1}^{K} \mathbf{x}_n(k) p_k^r \right) + p_{bias}^r. \tag{2}$$

The input feature weight p_k^r is a unique scalar to each rule and feature, which can be learned. This step can be thought of as the output each rule would make for the input vector that perfectly fires the antecedent clauses.

Aggregation Step: consists of calculating the weighted aggregation of the consequent clauses, $\mathbf{z}_n$, using the antecedent clauses, $\mathbf{w}_n$, as weights, to produce the output scalar y_n, i.e.,

$$y_n = \frac{\sum_{r=1}^{R} z_n^r w_n^r}{\sum_{r=1}^{R} w_n^r}. \tag{3}$$

This step can be thought of as the combination of the logical decisions from each rule based on how well the input vector matched each rule.

2.2 Optimization

Table 1 are the partial derivatives for ANFIS, based on a first order TSK model. Backpropagation based on the partial derivatives can be performed to update or learn the ANFIS parameters. Different methods can be used to perform learning updates; herein we use PyTorch's automatic differentiation. We have open source PyTorch codes made available at https://github.com/Blake-Ruprecht/Fuzzy-Fusion.

Table 1. ANFIS Derivatives for Gradient Descent

$$\frac{\partial y_n}{\partial z_n^r} = \frac{w_n^r}{\sum_{i=1}^R w_n^i}$$

$$\frac{\partial y_n}{\partial w_n^r} = \frac{\sum_{j=1, j\neq r}^R w_n^j (z_n^r - z_n^j)}{(\sum_{i=1}^R w_n^i)^2}$$

$$\frac{\partial y_n}{\partial p_k^r} = \frac{\partial y_n}{\partial z_n^r} \cdot \mathbf{x}_n(k) = \frac{w_n^r}{\sum_{i=1}^R w_n^i} \mathbf{x}_n(k)$$

$$\frac{\partial y_n}{\partial A_k^r(\cdot)} = \Big(\frac{\partial y_n}{\partial w_n^r} \Big) \Big(\prod_{j=1, j\neq k}^K A_j^r(\mathbf{x}_n(j)) \Big)$$

$$\frac{\partial y_n}{\partial \Theta_m^{r,k}} = \Big(\frac{\partial y_n}{\partial A_k^r(\cdot)} \Big) \Big(\frac{\partial A_k^r(\cdot)}{\partial \Theta_m^{r,k}} \Big)$$

$$\frac{\partial N(\cdot)}{\partial \mu} = exp\Big(-\frac{(x-\mu)^2}{2\sigma^2} \Big) \Big(\frac{x-\mu}{\sigma^2} \Big)$$

$$\frac{\partial N(\cdot)}{\partial \sigma} = exp\Big(-\frac{(x-\mu)^2}{2\sigma^2} \Big) \Big(\frac{(x-\mu)^2}{\sigma^3} \Big)$$

3. INITIALIZATION

The parameters of the Antecedent Firing step, namely the membership functions $A_k^r(\cdot)$ must be determined during initialization. The parameters for the Consequent Component Building, namely p_k^r, must be initialized.

3.1 Antecedent Parameters

Due to the dead gradient problem with ANFIS, as discussed in,[3] antecedent parameter initialization is crucial to the performance of the algorithm. Different methods may be used, including random initialization, expert-determined, and clustering to inform where potential membership functions should instantiate. We have used K-means and SP1M clustering to estimate the initial parameters for the membership functions. As shown in,[3] SP1M clustering creates the most accurate membership function estimations. Expert knowledge can also be used to determine the initial position of the membership functions, as discussed in section 6.

3.2 Consequent Parameters

Different methods may be used to initialize the consequent parameters, including random initialization, and Least-Means Squared (LMS) estimation. Herein, we use LMS, since the TSK system is linear, and it provides a good starting point for learning.

4. MEMBERSHIP FUNCTIONS

The choice of membership functions shapes not only the learning behavior of ANFIS, but the semantic interpretation too. Factors such as differentiability, simplicity, nonlinearity, fuzzy support and core, and more must be considered when choosing membership functions in ANFIS.

4.1 Gaussian Membership Function

By far, the most commonly utilized membership function for ANFIS is the Gaussian,

$$N(\mathbf{x}_n(k); \mu_k, \sigma_k) = e^{\Big(-\frac{1}{2} \frac{(\mathbf{x}_n(k) - \mu_k)^2}{\sigma_k^2} \Big)}. \tag{4}$$

While desirable, e.g., for differentiation, simplicity to work with, nonlinear, and infinite support (theoretically), a semantic limitation is that the standard Gaussian can only have a single element with membership value

one. From a modeling standpoint, where one actually cares about the membership values and their qualitative interpretations, this can prove to be overly restrictive. Furthermore, Gaussian functions are symmetric about the mean, meaning any type of skew cannot be learned. This restricts the membership function further.

4.2 Trapezoidal Membership Function

While there are numerous functions to remedy this shortcoming, herein we focus on the trapezoidal function,

$$T(\mathbf{x}_n(k);\boldsymbol{\Theta}) = \begin{cases} 0 & (\mathbf{x}_n(k) < \Theta_1^k) \text{ or } (\mathbf{x}_n(k) > \Theta_4^k) \\ 1 & \Theta_2^k \leq \mathbf{x}_n(k) \leq \Theta_3^k \\ \frac{\mathbf{x}_n(k)-\Theta_1^k}{\Theta_2^k-\Theta_1^k} & \Theta_1^k \leq \mathbf{x}_n(k) < \Theta_2^k \\ \frac{\Theta_4^k-\mathbf{x}_n(k)}{\Theta_4^k-\Theta_3^k} & \Theta_3^k < \mathbf{x}_n(k) \leq \Theta_4^k. \end{cases} \tag{5}$$

The trapezoidal function remedies the semantic limitations of the Gaussian through the inclusion of many elements that may have membership value one, and not necessarily infinite support. The trapezoidal function also allows asymmetric behavior to be learned, increasing the quality of interpretations.

5. DIFFERENT "TYPES" OF RULES

In the pursuit of explainable and interpretable solutions, it is not enough to say that ANFIS is automatically understandable simply because it consists of rules. This story might hold if we just encode human knowledge into ANFIS. But, if we learn an ANFIS from scratch and/or if we augment—start with the human and optimized from there—is the resultant solution understandable? To this end, we contemplate some of the scenarios that could arise in the context of a (potentially partially) machine learned ANFIS solution. Note, we discuss different types of rules that ANFIS can discover, but we do not consider below any post processing procedures to identify and/or possibility remediate their existence. We mention this because we do not want to discredit such a possibility, but we do not want to consider one as well since we are unaware of how it would be conducted.

Exception Rules: These rules are rare or they are not present in the data. In the case of the latter, they can be added after the fact by a human based on expert knowledge. In the case of the prior, there is a low probability that ANFIS will naturally discover them. Nevertheless, they are important as they present real logic that the system needs to learn.

Noise Rules: These rules correspond to noise in the underlying data and they may arise from factors like initialization and/or rule over specification. The problem is these rules are not real and they can lead to over fitting. From a learning standpoint, they are not desirable but they may be otherwise indistinguishable by the machine from exception rules.

Counter Rule: Counter rules are cancel one another. In theory, these rules should have similar antecedents, they just have opposite effects (consequences). These rules are problematic because they lead to wasted computation and from an XAI standpoint they complicate matters.

Duplicate Rules: Similar to counter rules, duplicate rules consist of multiple rules covering more-or-less the same antecedent space. If the consequent components are similar then this leads to increased weighting of an underlying rule during aggregation. In a sense, this is ANFIS gaming the system, as its a way to high jack the rule aggregation process and improve the importance of a rule.

Split Rules: These types of rules occur when multiple rules subdivide a rule and work together, versus a more compact and succinct single rule solution. In terms of XAI, this equates to an "overly wordy" machine.

Dead Rule: These are rules that are not associated with any underlying data samples (clusters or noise). Instead, they never fire and they pose a real risk, e.g., ANFIS not generalizing to new data. The reader can refer to our recent work,[3] which proves their existence due to initialization and a dead gradient problem.

Good Rules: We could not end this section without discussing the most positive outcome of ANFIS. A good rule is a real relationship that exists and we have learned. Technically speaking, a good rule would be a rule whose antecedents accurately capture the underlying uncertainty in the antecedents and well approximate the consequence. These are the underlying generative structure that we desire ANFIS to learn and wish to learn and explain to a user.

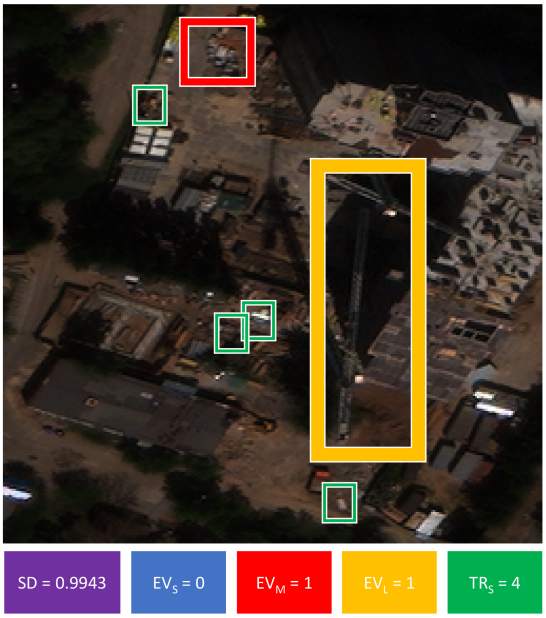

Figure 3. This figure illustrates an example of our construction site application. We show some of the features based on their parts-detector categories and their resultant outputs. Note, the entire image (region of interest in the context of broad area scanning) is analyzed for SD, the construction site detector confidence.

6. CASE STUDY: PARTS-BASED CONSTRUCTION SITE REASONING

Construction site detection from satellite imagery is difficult from a machine learning perspective. There is much variation in the different features and amounts of those features that make up construction sites. Due to this variation, parts-based detectors can be used to classify the features of a construction site to simplify the problem. These simpler detectors can be used as the input to a neuro-fuzzy system, which will then build fuzzy rules and TSK linear relations to fuse the parts into a decision.

The dataset used is from the "DIUx xView 2018 Detection Challenge".[4] The dataset consists of a series of large image scenes with sets of feature/object bounding boxes. xView is one of the largest and most diverse publicly available satellite imagery object-detection datasets. The initial results used in this paper, i.e. the human expert, were provided by Cannaday et al.[5]

The construction site problem consists of identifying construction sites from satellite imagery based on parts-detection. Our usage of the xView dataset involves four different parts-detectors: small, medium, and large Engineering Vehicles (EV_S, EV_M, and EV_L, respectively), small Trucks (TR_S), and the inference output from the ProxylessNAS DNN trained model (SD), detailed in [4].[5] An example of a construction site image with some example parts-detectors is shown in Figure 3.

6.1 Experiment 1: Human Replication

Parts-based detection can be done by ANFIS with the goal of replicating a human expert's knowledge. Since the rules used by a human in this task typically look like threshold based IF-THEN rules (see Figure 4. These rules can be manually instantiated in the antecedent firing step of ANFIS, allowing the system to replicate the logical rules used by an expert. The trapezoidal membership function can be made to perform like a step-function when the Θ_1^k and Θ_2^k parameters are equal (creating a vertical line), and the Θ_3^k parameter is suitably large enough to create a core encompassing all possible input values greater than the vertical threshold. The consequent component building step must be instantiated to reflect a step-function, which we accomplished using a linear

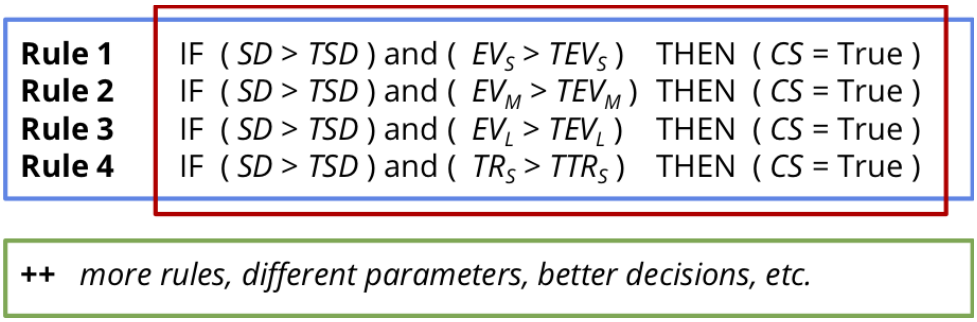

Figure 4. An example of the four rules used by our expert. The variables with a T in front are thresholds. These four rules were instantiated in Experiment 1 to replicate human knowledge (the blue box). In Experiment 2, ANFIS from scratch learned new rules and parameters in an attempt to make better decisions. Experiment 3 (the red box), shows what happens when we combine the two different strategies to utilize the benefits of both.

equation solver. Note, for the sake of this experiment, we turned off the learning feature of ANFIS to ensure the rules did not improve, and only replicated expert knowledge. Under these restrictions and instantiations, ANFIS was able to perfectly replicate the expert-informed rules, leading to the same exact performance and interpretability.

6.2 Experiment 2: ANFIS from Scratch

In our second experiment, we instantiated ANFIS using SP1M clustering and least-means squared for the antecedent and the consequent parameters, respectively. The purpose of this experiment was to determine if smart instantiation of ANFIS from scratch could approach the performance of ANFIS with expert knowledge, while maintaining interpretability. In these experiments, learning was turned back on to allow ANFIS to refine the instantiation methods over time. ANFIS mostly learned the consequent parameters, and the initialized antecedent parameters tended to not move due to the diminishing/dead gradient problem. This led to an ANFIS solution from scratch the performed about as well as the human expert did on the task, but with worse interpretability. The rules ANFIS learned are able to be analyzed, and look similar to the human's IF-THEN structure, albeit with more fuzziness than the crisp thresholds used by the human. The rules are more difficult to analyze with respect to expected results are, losing some interpretability compared to the human rules. Learning ANFIS from scratch proves to be an okay method in terms of both performance and interpretability, but it doesn't transfer expert knowledge in any way, which could improve performance on this domain while maintaining interpretability.

6.3 Experiment 3: Human Augmentation

After the expert rules were replicated, we turned learning on to allow ANFIS to optimize the parameters of the antecedent firing and the consequent component building. Ideally, this method can utilize the learning ability of ANFIS along with the transference of expert knowledge to create solution that performs better and maintains interpretability. Based on the expert rules, ANFIS was able to optimize the parameters of the consequent step, resulting in better F1-scores, while maintaining the interpretability of the human rules in the antecedent step. This improved performance was accomplished using backpropgation of error, creating a better TSK linear regressor. The antecedent step continued to use interpretable thresholds, and the increase in performance can be interpreted as better consequent component parameters being learned to combine the antecedent firings in the aggregation step. Basically, the linear equation can learn how important the different number of vehicles are in comparison. Since the expert knowledge was transferred to this system, ANFIS did not have to learn from scratch, which prevented some of the known issues of ANFIS (see Section 5) from hampering learning.

All three methods offer the same level of explainability that is inherent in ANFIS. In each case, the main way to analyze the decisions made by the algorithm consist in looking at the core and support of the antecedent firing functions. In Experiment 1, it is easy to see that the core and support form a step-function antecedent for each rule. This is by design, since we implemented the antecedent initialization to function like this, and didn't turn on learning to change it. Experiment 3 hardly changes the antecedent parameters (due to previously mentioned problems), but in this case, explainability is maintained because the decision thresholds are still clear.

Table 2. Experimental Results

EXPERIMENT	TP	FP	F1-SCORE
Expert Rules	341	37	90.93%
1) Human Replication	341	37	90.93%
2) ANFIS from Scratch	344	44	90.53%
3) Human Augmentation	335	16	92.67%

These step-function antecedents are interpretable as such: if the input value is greater than the rule threshold, the antecedent firing strength for that rule is high (normally 1). If the antecedent firing strength is high, the consequent component is "listened to" during aggregation. Similarly, Experiment 3 has interpretable antecedent parameters. Since the trapezoidal function is defined by four parameters, we can view the parameters for each rule to determine the core and support for each rule. In our experimentation, we discovered that ANFIS from scratch did not create step threshold functions, which is not surprising due to the initialization strategy, but did create explainable rules. It is easy to see that the core and support of the ANFIS from scratch rules define what types of values each rule fires for. In ANFIS, the consequent component step creates a linear logical decision; the explainability of the system resides within the antecedent firing parameters.

7. SUMMARY AND FUTURE WORK

Herein, we explored the role of neuro-fuzzy logic on a real-world problem, parts-based reasoning of complex scenes for remote sensing. First, we showed a way to encode expert knowledge into a machine. Next, we showed that ANFIS could optimize this starting logic and find a better solution, which makes sense because it is hard for a user to optimize their logic. The final machine logic could be understood and explained back to the expert. However, the weakest part of the explanations are the rule consequences, which are likely due to a type-one TSK inference formulation. Last, we also showed that it is possible to learn the fuzzy logic system from scratch—even in light of the fact that ANFIS is hindered by diminishing to dead gradients that lessen its ability to learn—and we found a similar performing solution but with different logic. In summary, we demonstrated a way to take a good explanation, from the human, to encode it into the machine and to improve on that logic. This process is a good example of human-machine teaming and the answer was better than the machine learned only solution likely due to the domain knowledge of the human.

In light of the above case study, a number of weaknesses were identified. First, another neuro-fuzzy inference system beyond ANFIS needs to be used or ANFIS needs to be improved. There are severe hindrances that restrict ANFIS from achieving desired logic. Next, we spent a lot of time trying to get "good explanations". However, what is a good explanation? That needs to be defined and likely folded into the learning process if the goal is to find high quality answers that are also high quality explanations to the expert. Last, we took the output from a set of parts-based detectors. In future work it would be more beneficial if the machine could extract its own features, perhaps based on aspects like spatial relationships between evidence in the scene. However, our concern is that the quality degree of an explanation largely depends on the interpretability of the input; e.g., black box in, black box out.

REFERENCES

[1] Jang, J. . R., "Anfis: adaptive-network-based fuzzy inference system," *IEEE Transactions on SMC* **23**, 665–685 (May 1993).

[2] Sugeno, M., [*Industrial Applications of Fuzzy Control*], Elsevier Science Inc., New York, NY, USA (1985).

[3] Ruprecht, B., Veal, C., Murray, B., Islam, M., Anderson, D., Petry, F., Keller, J., Scott, G., and Davis, C., "Possibilistic clustering enabled neuro fuzzy logic," in [*FUZZ-IEEE*], (2020, under review).

[4] Lam, D., Kuzma, R., McGee, K., Dooley, S., Laielli, M., Klaric, M., Bulatov, Y., and McCord, B., "xview: Objects in context in overhead imagery," (2018).

[5] Cannaday II, A. B., Chastain, R. L., Hurt, J. A., Davis, C. H., Scott, G. J., and Maltenfort, A. J., "Decision-level fusion of dnn outputs for improving feature detection performance on large-scale remote sensing image datasets," in [*2019 IEEE International Conference on Big Data (Big Data)*], 5428–5436 (Dec 2019).