Combination of Hedge Algebra and Type-2 Fuzzy System for Electrocardiogram Signal Recognition and Classification

Hoai Linh TRAN
Hanoi University of Science and Technology, Ha Noi, Vietnam
Corresponding author email: linh.tranhoai@hust.edu.vn

Abstract

The paper presents a new application of Hedge Algebra in Type 2 (HAT2) Fuzzy System (FS) for electrocardiographical signal recognition and classification. The HAT2 integrates the capability of fuzzy logic in modeling uncertainties of data and the linguistic variables to describe and manipulate the knowledge in a way closer to the way humans describe facts and rules. An adaptive learning algorithm will be also proposed to train the system's parameters for better fitting to the data samples. The proposed approach will be tested with the heart beat classification problem with different types of arrhythmias in the ECG signals taken from the Arrhythmia Databases of Massachusetts Institute of Technology and Boston's Beth Israel Hospital. The numerical results will be compared with other methods to show the high quality of proposed solution. The proposed solution is also shown being simple with low complexity of computation, which makes it suitable for use in IoT or portable devices for intelligent solutions used in modern healthcare systems.

Keywords: Type-2 fuzzy logic, hedge algebra, arrhythmia recognition.

1. Introduction

An electrocardiogram (ECG) is a simple way to track the patients' conditions. The most difficult problem in ECG signal analysis are the noises, which cause sometimes very large variation in the morphologies of waveforms. ECG signals themselves are non-stationary but there are different types of noises, for example, the power line interference (50 Hz or 60 Hz), the electrode contact noise, the patient's body movements, or even the mental conditions. The situation is even worse when the patients have cardiac arrhythmias. Due to this need, despite there are already many works related to the problem of heart beat recognition and classification, the interest in new models of arrhythmia recognition is still high.

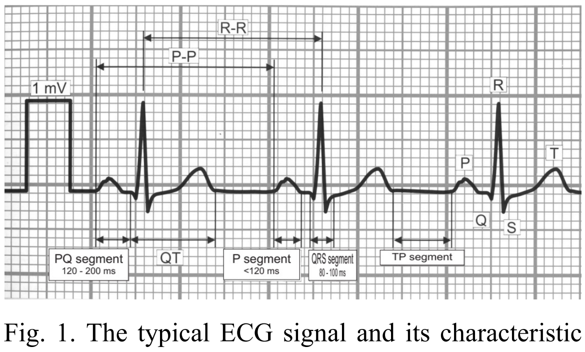

Fig. 1. The typical ECG signal and its characteristic peaks P-Q-R-S-T.

The research on high quality ECG signals recognition should be performed on various aspects including good feature extraction, effective noise filtering, training recognition model,... For a single patient and one type of arrhythmia the accuracy could be very high, even 98-99% was achieved, but for different patients and more than one type of arrhythmia, it's much more difficult to reach that level of accuracy [1, 2]. Also, another popular approach in many research is to classify all the arrhythmia beats as non-normal in a binary classification problem. This type of simplified approach usually allows to achieve high accuracy results [3-6].

For feature extraction, the models can use simple features based on characteristic points Q, R, S, T,. . . , which are seen in Fig. 1 [6], or may use some more sophisticated ones. Different 'good' features were proposed, such as based on Karhunen - Loeve functions, based on Hermite functions [6], based on wavelet transform and independent component analysis [7], or using the Kalman filter,... As classifier, various mathematical models were proposed like the Learning Vector Quantization, Support Vector Machine, Bayesian model [7],... In recent years, with the development of so-called deep learning networks, many works have been performed to test these new tools with the classic problems of ECG recognition. The main problems for deep learning-based networks are the big structures of the networks, which are capable to capture more features from the data, but at the same time require much more data to train them efficiently when ECG databases have not enough samples for all the types of arrythmias. Among these deep learning networks we can mention the more

ISSN: 2734-9373
https://doi.org/10.51316/jst.163.ssad.2023.33.1.4
Received: July 20, 2022; accepted: October 10, 2022

popular ones such as the Convolutional Neural Network (CNN), Deep Belief Network (DBN), and the Long Short-Term Memory (LSTM). The CNN networks were used in [1, 3] for individual class-based detection with accuracy as high as 99.95%, but for 4-class recognition, the F_1 score was dropped to 79% with testing data sets. In [4], DBN was applied for recognizing SVEB or VEB rhythms with accuracies of 97.5% for SVEB and 98.6% for VEB. In [2], the authors tested 3 different deep learning networks (CNN, SincNet, and CNN with entropy features) to recognize 5 classes of rhythms: normal, myocardial infarction, ST/T change, conduction disturbance, and hypertrophy beats. With the recognition of 5 classes, the CNN achieved 72.0%, SincNet achieved 73.0%, CNN with entropy features achieved 76.5% of accuracy. Another work [5] using CNN network for 5-class recognition could achieve the 97.41% of accuracy. More detailed reviews of ECG recognition solutions can be found in [7, 8].

There are different approaches to deal with the noises, from statistical-based techniques as the hidden Markov model to soft computing methods such as artificial neural networks and fuzzy rule-based classification systems. Among them, the contributions of fuzzy logic systems for the problem are highly recommended [9] because of many advantages, including the ability to flexibly represent objects by fuzzy sets, and the interpretability of the results.

The fuzzy logic was first presented by Zadeh in [10] to model human reasoning processes. In 1975, Zadeh extended the ideas of the original fuzzy logic theory and proposed the Type-2 fuzzy sets (T2-FS). T2-FS is a fuzzy set, whose membership grade for each element is a Type-1 fuzzy sets (T1-FS) in the interval [0,1] [10]. Recently, a new class of T2-FS, the so-called hedge algebraic T2-FS (HAT2-FS) has been proposed in [11]. HAT2-FS is a fuzzy set with linguistic membership grades, the calculation and inference on HAT2-FS are based on the characteristics of the linguistic truth values in hedge algebra. The HAT2-FS not only is closer to the human way of reasoning, but its set operations and inferring process are not too complicated; hence, the amount of calculation is manageable [11]. In this paper, we will present a new method of ECG signal classification using a HAT2-FS. The model will be tested with the same data from the MIT-BIH database [6] similar to some previous works [6, 12] for easier comparison of the performances.

2. Type-2 Fuzzy Logic Systems and Hedge Algebra

2.1. Type-2 Fuzzy Logic Systems

In most fuzzy logic systems [10, 13] (also known as the Type-1 fuzzy sets - T1-FS), the membership functions are crisp numbers. For example, to describe the truthfulness of "*x is around 4*" we define a *membership grade function* $\mu_{\approx 4}(x)$ and use a bell-type membership function like $\mu_{\approx 4}(x) = e^{-(x-4)^2}$. In this case for a given *x* there will be an exact (crisp) number to be the value of the membership function. For example, for *x* equal 3, the value of the membership is $\mu_{\approx 4}(3) = e^{-(3-4)^2} = 0.37$. But in reality, we usually have a situation in which we say "the truthfulness of $3 \approx 4$ is *about 0.37*" rather than "the truthfulness of $3 \approx 4$ is *exact 0.37*". Despite having a name which carries the connotation of uncertainty, research has shown that there are limitations in the ability of T1-FS to model the effect of uncertainties [9]. In order to extend the capability of a fuzzy system to deal with uncertain membership functions, Zadeh introduced Type-2 fuzzy sets (or T2-FS) as an expansion of T1-FS. Type-2 fuzzy sets are fuzzy sets whose membership function of an element is T1-FS, not a crisp number. For example instead of $\mu_{\approx 4}(3) = 0.37$ we have $\mu_{\approx 4}(3) \approx 0.37$, which in turn can be modeled by a bell-type function around the point 0.37 like $\mu_{\approx 4}(3) = f(u) = e^{-(u-0.37)^2}$ for $u \in [0,1]$. There are different ways to define the second fuzzy membership function $\mu_{\approx 4}(3) = f(u)$. Among these ways, the Interval Type-2 (IT2) FS [9], a special case of T2-FS, is currently the most widely used for its reduced computational cost. As given by name, in the IT2 model, the 2nd membership function is an interval, for example

$$\mu_{\approx 4}(3) = f(u) = \begin{cases} 1 & \textit{for} \quad u \in [0.14; 0.61] \\ 0 & \textit{for} \quad u \notin [0.14; 0.61] \end{cases} \tag{1}$$

With those intervals, we can see that an IT2-FS is bounded from the above and below by two T1-FS, $\underline{X}$ and $\overline{X}$, which are called upper MF (UMF) and lower MF (LMF), respectively as presented in Fig. 2.

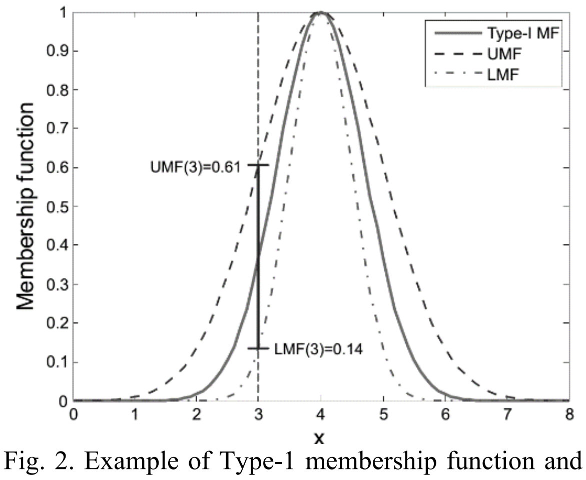

Fig. 2. Example of Type-1 membership function and its UMF and LMF in Type-2 fuzzy logic

Let an IT2-FLS work with vectors of N inputs, M rules and (for simplification) $K=1$ output. For an input $\mathbf{x}=\left[x_1,x_2,\ldots,x_N\right]$, a T2 rule of the system has the form:

$$\begin{aligned}
\textbf{if}\quad & \left(x_1 \; is \; \tilde{X}_{m,1}\right) \; and \\
& \left(x_2 \; is \; \tilde{X}_{m,2}\right) \; and \\
& \ldots \\
& \left(x_N \; is \; \tilde{X}_{m,N}\right) \\
\textbf{then}\quad & y \; is \; \tilde{Y}_m
\end{aligned} \tag{2}$$

(for $m=1,2,\ldots,M$) where $\tilde{X}_{m,n}=\left[\underline{X}_{m,n},\overline{X}_{m,n}\right]$, $(n=1,2,\ldots,N)$ are IT2-FSs, and $\tilde{Y}_m$ is an interval $\tilde{Y}_m=\left[\underline{y}_m,\overline{y}_m\right]$. We can have a crisp output by setting $\underline{y}_m=\overline{y}_m$. For each rule, the crisp output and the firing interval of the output are calculated as in [9].

If the IT2 rule in Eq. (2) is in training mode, the parameters of two functions UMF and LMF and the two interval limits $\underline{y}_m$, $\overline{y}_m$ are adapted [13].

2.2. Hedge Algebra

Hedge algebra is one of the approaches to manipulate on words both qualitatively and quantitatively [9]. Recently, a new class of T2-FS, the Hedge algebraic Type-2 fuzzy set (HaT2-FS) has been proposed in [11, 13]. In appearance, HaT2-FS is a fuzzy set with linguistic membership grades; however, the approach of HaT2-FS is radically different. Instead of traditional processing, the calculation and inference on HaT2-FS are operated based on the characteristics of the linguistic truth values in hedge algebra. Hedge algebra is an algebraic approach to represent and handle values of a linguistic variable based on their semantic order obtained by acting hedges into other elements of linguistic variables.

Formally a hedge algebra is a quatre $(AX,G,H,'\le')$, in which AX is the set of linguistic values, G is the set of generators, H is the set of hedges, and $'\le'$ is the ordered relationship among elements of hedge algebra. For example, with 2 basic terms '*High*' and '*Low*', let's define 4 so-called hedge operators (or hedges) '*Very*', '*More*', '*Approximately*', '*Less*'. By applying the hedges into the base terms we can generate new linguistic values, such as '*Very Very High*', '*Approximately Very Low*', etc. In this paper we will consider the linear symmetrical hedge algebra [11], in which:

- G has only two opposite generators, one of which has the "stronger" meaning, e.g., *High*, *True*, *Big*,... as a positive generator (denoted by c^+), and the other e.g. *Low*, *False*, *Small*,... as a negative generator (denoted by c^-).
- All the hedges in H are also ordered related to the operator $'\le'$.

Each hedge operator from H weakens or strengthens the meaning of generators from G and weakens or strengthens the meaning of other hedges. We use the SIG relation to represent the application of a hedge to other hedges or generators, $SIG: H\times\left(H\cup G\right)\rightarrow\left\{-1,1\right\}$:

- $SIG(h,k)=1$, if the hedge h strengthens the meaning of the term k,
- $SIG(h,k)=-1$, if h weakens the meaning of the term k.

Table 1 is an example of SIG relation for the above algebra.

Table 1. An example of SIG relation

SIG	*Very*	*More*	*Approx.*	*Less*	*c*
Very	1	1	-1	1	1
More	1	1	-1	1	1
Approx.	-1	-1	1	-1	-1
Less	-1	-1	1	-1	-1

For the above example, we can have the following linguistic values and their orders: *Very Low < More Low < Low < Approximately Low < Less Low < Less High < Approximately High < High < More High < Very High*. In linear symmetrical hedge algebras all elements are comparable, then we can construct a function which maps them onto values in the interval [0,1] without changing their orders. In [11] the notion of fuzziness measure and semantically quantifying mapping of linguistic values were proposed. For calculating these measures of each linguistic value we have to determine fuzziness measure *fm*(*c*) for all *c* ∈ *G* and fuzziness measure *fm*(*h*) for all *h* ∈ *H*. The parameters *fm*(*c*) and *fm*(*h*) are called fuzziness parameters of the hedge algebra.

Naturally, we can apply more than one hedge to the generator, for example, we have the value '*Very Very High*'. The number of hedges in a linguistic value will be called *the level* of the given linguistic value. The higher the level, the more linguistic values (and the associated intervals). With an HA of 2 generators ('*High*' and '*Low*') and 4 hedges ('*Very*', '*More*', '*Approximately*', '*Less*') and the maximum level equal *L* we may have

$$2+2\cdot4+\ldots+2\cdot4^L=\frac{2}{3}\left(4^{L+1}-1\right)$$

different hedged linguistic values. If all the basic terms and hedges are ordered then all the general linguistic terms are also ordered. For example from '*Low*' > '*Less Low*' it follows that '*Very Possible Low*' > '*Very Possible Less Low*',...

Hence, each linguistic value $\hat{x}$ is represented by a triple $\left(\left[\underline{fm}(\hat{x}),\overline{fm}(\hat{x})\right],v(\hat{x})\right)$ in which $\left[\underline{fm}(\hat{x}),\overline{fm}(\hat{x})\right]\subseteq[0,1]$ and $v(\hat{x})\in[0,1]$ are the fuzziness interval of $\hat{x}$ and the semantically quantifying value respectively [13]. In this paper the $v(\hat{x})$ is calculated using formula:

$$v(\hat{x})=\frac{\underline{fm}(\hat{x})+\overline{fm}(\hat{x})}{2} \tag{3}$$

With the interval $\left[\underline{fm}(\hat{x}),\overline{fm}(\hat{x})\right]\subseteq[0,1]$, the fuzziness of $\hat{x}$ is defined as the width of the interval:

$$fm(\hat{x})=\overline{fm}(\hat{x})-\underline{fm}(\hat{x}) \tag{4}$$

The fuzziness measure $fm: X\rightarrow[0,1]$ has following properties:

1. *fm* is a full measure, i.e.:
   - $\forall c\in G:\sum_{c} fm(c)=1;$ in case of symmetrical algebra, such that $G=\{c^{+},c^{-}\}$, we have $fm(c^{+})+fm(c^{-})=1;$
   - $\forall h\in H:\sum_{h} fm(h)=1;$
2. $\forall c\in G,h\in H: fm(h{\cdot}c)=fm(h){\cdot}fm(c).$
3. If $\hat{x}$ is crisp term, i.e. $\forall h\in H: h(\hat{x})=\{\hat{x}\}$ then $fm(\hat{x})=0.$

Example: Consider hedge algebra (AX, {*High*, *Low*}, {*Very*, *More*, *Approximately*, *Less*}, ≤) with the fuzziness parameters

fm(*High*) = 0.7,

fm(*Low*) = 0.3,

fm(*Very*) = 0.3,

fm(*More*) = 0.3,

fm(*Approximately*) = 0.2,

fm (*Less*) = 0.2.

The fuzziness measures satisfy condition 1:

$$fm(True)+fm(False)=1$$

and

$$1=fm(Very)+fm(More)+fm(Approximately)+fm(Less)$$

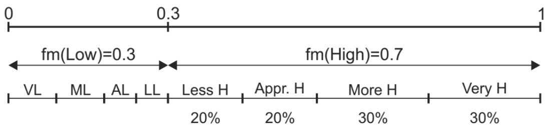

Fig. 3. An example of intervals for level 0 and level 1 hedged linguistic values.

In Fig. 3 there are the example intervals for linguistic values of basic terms and 1st level terms for the example algebra given above.

When using the average semantically quantifying value defined in (3), we can complete the triple for the linguistic values as:

- *True* → ([0.3, 1], 0.65),
- *Very True* → ([0.79, 1], 0.895),
- *More True* → ([0.58, 0.79], 0.685),
- *Approximate True* → ([0.44, 0.58], 0.51),
- *Less True* → ([0.3, 0.44], 0.37),
- *False* → ([0, 0.3], 0.15),...

When the intervals are disjointed then the average semantical quantifying operator is enough, but when intervals are not disjointed we usually use the "round to the nearest interval centre" method to quantify the linguistic values, and we can convert a crisp number to the nearest linguistic value. For the example in Fig. 3, all crisp value bigger than:

$$\frac{0.895+0.685}{2}=0.79$$

will be converted to '*Very High*' (or '*VH*'). And for the defuzzification process, a linguistic value would be converted into its semantical quantifying crisp value.

2.3. Hedge Algebra Type-2 Fuzzy Logic Rule

From the ideas of hedge algebra described above, in comparison with T2 reasoning rule given in (2), a set of M HAT2 reasoning rules can be defined with the form:

$$\begin{aligned}\textbf{if}\quad &\left(H_{m,1}c_1(x_1)\right)\ \textit{and}\\ &\left(H_{m,2}c_2(x_2)\right)\ \textit{and}\\ &\ldots\\ &\left(H_{m,N}c_N(x_N)\right)\\ \textbf{then}\quad & y\ is\ \tilde{Y}\end{aligned} \tag{5}$$

where $m=1,2,\ldots,M$ - rule number, $n=1,2,\ldots,N$ - the input dimension index, c_n - basic terms, x_n - input variables, L - the level (or the maximum length) of applied hedges, y - response of the rule and $H_{m,n}$ - the chain of hedges applied to the basic terms $h_{m,n,l}\in H$:

$$H_{m,n} = h_{m,n,L} \cdot h_{m,n,L-1} \cdot \ldots \cdot h_{m,n,1} = \prod_{l=1}^{L} h_{m,n,l} \tag{6}$$

As it can be seen when comparing with (2), for each dimension x_n, the fuzzifier was changed from the form $\left(x_n \ \ is \ \ \tilde{X}_{m,n}\right)$ with the interval membership grades equal $\left[\underline{X}_{m,n}, \overline{X}_{m,n}\right]$ to the form $\left(h_{m,n,L} h_{m,n,L-1} \cdots h_{m,n,1} c_n(x_n)\right)$ with the inverval membership equal

$$\overline{H}_{m,n}(x_n) - \underline{H}_{m,n}(x_n) = fm(c_n) \cdot \prod_{l=1}^{L} fm\left(h_{m,n,l}\right) \tag{7}$$

The antecedent of HAT2 reasoning rule can be adapted during the training process by changing the fuzziness measure of the basic term c_i or the fuzziness of the hedges. In this paper for simplification we use the constant, equal fuzziness measure for all the hedges (if we use 4 hedges, their fuzziness measure will be equal to 0.25).

As it can be easily seen in Fig. 3, for the positive basic term c^+, if we increase its fuzziness, all the intervals will be shifted to the left, which in turn decreases the semantical values of the variables. Analogically, for the negative basic term c^-, increasing its fuzziness will shift all the intervals to the right, which in turn increases the semantical values of the variables. We can also change the fuzziness measure of a term by changing the hedges. For example, if we want to have something smaller than '*Approximate High*', we can use '*Less High*' or '*Approximate Approximate High*' instead. If we want to have something smaller than '*Less High*', we can use '*Less Low*' for example.

3. Building a HAT2-FLS from a Data Set

The described above HAT2-FLS is a strongly nonlinear reasoning system. The parameters of this system will be estimated from a given set of data samples during a typical supervised learning process. In this paper, we propose the below scheme to initiate and tune a HAT2-FLS based on given data samples. The scheme contains two main phases:

- Phase 1 (steps 1 and 2): construct Type-1 reasoning rules as described in (2);
- Phase 2 (steps 3 and 4): convert Type-1 rules into HAT2 based rules as in (5).

The rules will be divided into a set of 'good' rules, which correctly classify some learning data, and a set of 'conflicted' rules, which incorrectly classify some learning data. The 'conflicted' rules are later further tuned to remove the conflicts.

Input: A data sets with p sample pairs $\{\mathbf{x}_i, d_i\}$, $i = 1, 2, \ldots, p$, where $\mathbf{x}_i \in \mathbb{R}^N$, $d_i \in \mathbb{R}$;

- Preselected $C_{\max}$ - number of membership functions to be generated for each input;
- Preselected L - maximum level of hedges.
- An empty initial set of rules and an empty initial set of 'conflicted' rules.

Output: Sets of rules HAT2-FLS

Step 1: For each input dimension number $n = 1, 2, \ldots, N,$ find the $C_{\max}$ rule centres $C_{n,1}, C_{n,2}, \ldots, C_{n,C_{\max}}$ of fuzzy membership functions using K-means algorithm [14]. Let these centres be sorted in ascending order. Generate the membership functions based on those centres as shown in Fig. 4 (the leftmost centre is associated with a left-open membership function, the rightmost centre is associated with a right-open membership function, and middle centres are associated with triangle membership functions). We also limit that for all centres $r = 1, 2, \ldots, C_{\max}$, the non-zero domain of membership function of centre $C_{n,r}$ should be limited between $C_{n,r-1}$ and $C_{n,r+1}$. This condition will help us to have that for all *x*, there would be a maximum of two consecutive membership functions, which are non-zero only.

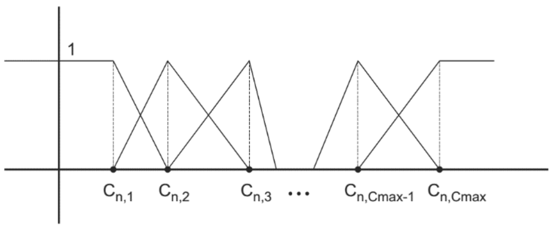

Fig. 4. The initial membership functions

Step 2: For each sample $\{\mathbf{x}_i, d_i\}$, $i = 1, 2, \ldots, p,$

- Construct a temporary rule

$$\begin{aligned} \textbf{if} \quad & \left(x_{i1} \ is \ \tilde{X}_{temp,1}\right) and \\ & \left(x_{i2} \ is \ \tilde{X}_{temp,2}\right) and \\ & \cdots \\ & \left(x_{iN} \ is \ \tilde{X}_{temp,N}\right) \\ \textbf{then} \quad & y \ is \ d_i \end{aligned} \tag{8}$$

whose the fuzzy sets $\tilde{X}_{temp,n}$ for each dimension n of the input $(n = 1, 2, \ldots, N)$, is around the centre $C_{n,r}$, which has the highest membership degree for the input $x_{i,n}$. The firing membership of the rule is equal to the product of membership grades of all components

$$\mu_{\tilde{\mathbf{X}}_{temp}}(\mathbf{x}_i) = \prod_{n=1}^{N} \mu_{\tilde{X}_{temp,n}}\left(x_{i,n}\right) \tag{9}$$

- If the antecedent of the just-generated temporary rule is not found then it's new and will be added to the working set. If there is already a rule previously generated with the same antecedent but a different consequent then the two rules (with the same antecedent part) are moved to the set of 'conflicted rules'.

When all the *p* sample pairs were considered, we check the conflicted rules, and for each antecedent part, among the conflicted rules we select the one with the highest membership grade and move this rule back to the set of 'good' rules. The others will be further tuned and updated in Step 4.

Step 3: Convert the set of rules of Type-1 to the rules of Type-2 format.

With an HA of 2 generators ('*High*' and '*Low'*), the hedges and the maximum level equal *L*, we have different hedged linguistic values, which have corresponding semantically quantifying values. For each rule, for each dimension, convert the membership degree (calculated in Step 2) of that dimension into the hedged linguistic value, which has the closest corresponding semantically quantifying value to the membership degree (as described in subsection 2.2).

Step 4: Adapt the rules set to improve the performance of the data samples set.

After Steps 2 and 3, we have two sets of rules: the set of correct rules and the set of 'conflicted' rules. The main purpose of Step 4 is to train and adapt the parameters of rules (from both sets) to improve the total performance on the learning set and the testing set (the number of misclassifications should be as low as possible). As mentioned earlier, a conflicted rule means there is a 'good' rule with the same antecedent but different consequent parts and the 'good' rule has a higher firing membership function.

In this paper, our proposed solution is to correct a conflicted rule to create a new, non-conflicted one, which can be used to classify the associated sample. The idea is presented in Fig. 5. Let the samples $\mathbf{x}_j$ and $\mathbf{x}_k$ belongs to the same rule number m, i.e. for all dimension n we have x_{jn} and x_{kn} belong to the same cluster $X_{n,r}$. It would be enough if we change the cluster's parameters, such that now there is one dimenesion x_{jn} and x_{kn} don't belong to the same cluster. We consider the dimension n, in which the differences between membership grades of $\mathbf{x}_j$ and $\mathbf{x}_k$ are the biggest (it means it would be easiest to separate the samples using that dimension):

$$\begin{aligned}&\mu\left(x_{jn} \approx c_{m,n,r}\right) - \mu\left(x_{kn} \approx c_{m,n,r}\right) = \\ &\max_{l=1,N}\left[\mu\left(x_{jl} \approx c_{m,l,r_l}\right) - \mu\left(x_{kl} \approx c_{m,l,r_l}\right)\right]\end{aligned} \tag{10}$$

The shapes of the membership function would be updated, such that after adaptation, the samples don't belong to the same rule as before. In order to achieve that, for the examples in Fig. 5b, the right border of the cluster number r and the left border of the cluster number $r+1$ would be "*shifted left*" so now the sample $\mathbf{x}_j$ belongs to cluster r, the sample $\mathbf{x}_k$ belongs to cluster $r+1$. This, in turn, will assign the two samples to two different rules.

The "shift left" operation in HAT2 FS can be done in two ways:

1. If the actual rule uses the positive basic term then increase the fuzzy measure of that basic term. If the actual rule uses the negative basic term then decrease the fuzzy measure of that basic term.
2. Use a weakening hedge if the maximum number of hedges was not reached.

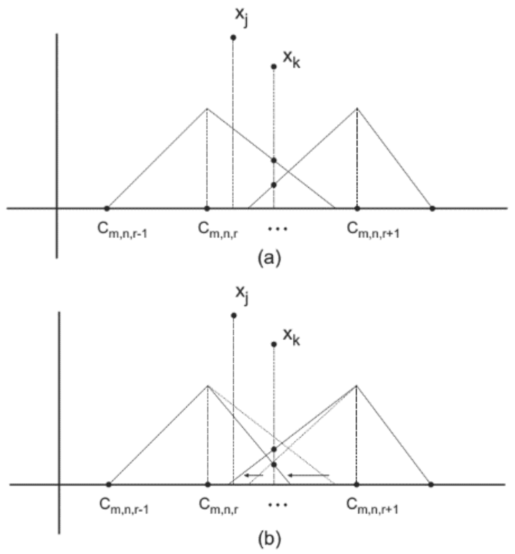

Fig. 5. Shifting the membership functions to move sample $\mathbf{x}_k$ from cluster $C_{m,n,r}$ (in (a)) to cluster $C_{m,n,r+1}$ (in (b)) while keeping sample $\mathbf{x}_j$ belong to cluster $C_{m,n,r}$

This "shifting" operation is simple, but it may cause problems because a changed rule may no longer work well with old samples. To deal with that, in this paper we also use the idea of the RPROP (*Resilient backPROPagation*) learning method [14]. The learning is done in batch mode, which means for each conflicted sample we count the decisions on which fuzzy measure (or borders) should be changed and how. When all the conflicted samples were considered, for all fuzzy measures (or borders), the final decision is the majority of the counted decisions. If there were more counts to move left, then we move the given linguistic term to the left once, if there are more counts to move right, then we move it to the right once. The step change was set to 1% of the actual given length of the intervals (the actual fuzziness of the linguistic value under training).

The batch mode is repeated many times (or so-called many epochs) until the number of conflicted samples cannot be decreased.

4. The ECG Signals and the Feature Extraction Method

In this paper, the above described HAT2-FLS is used as a classifier to analyse and recognize the ECG signals. This task is very popular and numerous works have been performed. However only a few models use Type-2 fuzzy logic like in [13] and we have not found any works using HA. In order to have an easy and direct comparison with previous works, we used the same data sets as in [6, 12] and for the detailed generations of features, use the description in [6]. In this paper, we just briefly describe the feature generation process.

The important step in building an efficient classifier system is diagnostic feature extraction. In our approach to the problem, we have applied the QRS complex decomposition into Hermite basis functions and used the decomposition coefficients as the features of the ECG signals.

The signals were taken from the popular MIT-BIH Arrhythmia Database [12]. The recognition of arrhythmia proceeded on the basis of the QRS segments of the registered ECG waveforms (only lead number 1 was used) of 19 patients [12] with six types of arrhythmia:

- left bundle branch block (L),
- right bundle branch block (R),
- atrial premature beat (A),
- ventricular premature beat (V),
- ventricular flutter wave (I),
- ventricular escape beat (E),
- and the waveforms corresponding to the normal sinus rhythm (N).

From a total of 6679 beats, an iterative method was used to split 3611 samples to the learning set and 3068 to the testing set. Table 2 presents the number of beat types used in learning and testing.

For each beat, we use the method described in [6] to extract the QRS complexes, whose lengths were 91 data points around the R peak (45 points before and 45 ones after). At the data sample rate of 360 Hz, this gives a window of 250ms.

The data has been also additionally zero-padded by adding 45 zeros to each end of the QRS segment to make the data look more similar to the Hermite basis functions. After that, we performed the decomposition by minimizing the approximation error:

$$E = \sum_i \left[s(t_i) - \sum_{n=0}^{N-1} c_n \phi_n(t_i, \sigma) \right]^2 \tag{11}$$

where c_n are the expansion coefficients, $\phi_n(t_i)$ - the Hermite basis functions of n-th order. These 16 coefficients c_n together with 2 classical time-based features: the instantaneous R-R interval of the beat (the time span between two consecutive R peaks) and the average R-R interval of 10 preceding beats, form the feature vector **x** applied to the input of the classifier.

Table 2. The number of learning and testing samples of each beat types

Beat type	Learn samples	Test samples
N	1065	935
L	639	561
R	515	485
A	504	398
V	549	451
I	271	201
E	68	37

5. The Numerical Experiments and Results

With the algorithm described in section 3, we have built the HAT2-FS for the data sets presented in section 4.

In step 1: For each of the 18 input dimensions, we generated only 3 clusters using the K-mean algorithm.

In step 2: From the 3611 learning samples, there were a total of 1697 rules generated, among which 1667 were 'good', and 30 'conflicted' rules.

In step 3, converting from Type-1 to HAT2: We used maximum length of hedges equal only 3, which means there are 170 intervals corresponded to 170 logic terms.

In step 4: After adapting the rules using the algorithm described in section 3, the number of conflicted rules was reduced to 3. After that, we test the trained HAT2-FLS with the new 3068 data samples.

Table 3 presents the confusion matrix for the testing data divided into beat types. The column presents the correct (desired) types, the row presents the classifier's results.

There were 65 samples wrongly classified, i.e. the classification testing error was

$$E_{test} = \frac{65}{3068} \cdot 100\% = 2.12\% \qquad (12)$$

Table 3. The detailed classifying results for 7 types of rhythms of testing data

	N	L	R	A	V	I	E
N	920	1	1	18	0	0	0
L	3	559	0	1	0	0	0
R	0	0	483	25	0	0	0
A	12	1	1	354	1	0	0
V	0	0	0	0	450	1	0
I	0	0	0	0	0	200	0
E	0	0	0	0	0	0	37
Total	935	561	485	398	451	201	37

Table 4. The testing errors of the proposed HAT2 and other neural networks MLP [12], TSK [12] and SVM [6] on the same training and testing data sets

Classifier system	Number of errors	% of errors
MLP	148	4.82%
TSK	100	3.26%
SVM	60	1.96%
HAT2	**65**	**2.12%**

To compare with other works, we selected the individual neural and fuzzy neural classifiers from [6, 12] because all these classifier networks have been first learned on the same learning data set and then tested on the same testing data set. The results are listed in Table 4. As can be seen, the HAT2 classifier shows better performance than the popular MLP and the type-I based TSK. Only the SVM networks achieved better performance with an error of 1.96%, but in fact, the SVM networks worked in the one-against-one mode [6], which means there were 21 subnets needed to train.

As mentioned in Section I, it's hard to compare the performance of a classifier with others when they are not trained and tested on the same data sets, but comparatively, we can note that the proposed method was tested with a 7-class recognition problem, including the types of ventricular flutter wave (denoted as type I) and ventricular escape beats (denoted as type E) with only 472 and 105 beats respectively. These types with a small number of samples available were usually skipped especially when deep learning networks were used. But our error of 2.12% for 7-class recognition problem is still very good compared with other works mentioned in Section I.

Table 5. The results of classifying ECG beats into normal and abnormal classes

	Normal	Abnormal
Normal (classified)	920	20
Abnormal (classified)	15	2113

We checked also the quality of the combined classifiers with the sensitivity and specificity statistics for binary classifiers. In order to do that we let all 6 types of arrhythmia be the "abnormal" type, so now the classifier task is to differ the normal beats from the abnormal ones. Also, let the abnormal beats be the positive cases and the normal beats be the negative cases. Table 5 presents the results of classifying beats into normal vs. abnormal class.

From the Table 5 we have

$$\begin{aligned} Sens &= \frac{TP}{TP+FN} = \frac{2113}{2113+20} = 99.06\% \\ Spec &= \frac{TN}{TN+FP} = \frac{920}{920+15} = 98.40\% \end{aligned} \qquad (13)$$

The above results, when compared to previous works mentioned in Section I, help to confirm the high quality of the proposed solution as an individual ECG signal classifier with high accuracy and simplicity to train.

6. Conclusion

An application of the hedge algebra type-2 fuzzy system to analyze and classify the ECG signal was presented in this paper. This tool combines the capability of fuzzy logic in modeling the noises as uncertainties of the data and the capability of hedge algebra in manipulating the linguistic terms to present knowledge. As a nonlinear classifier, the HAT2 FS is easy to initiate and train.

The classifier used the features consisted of coefficient of ECG signals decompositions using Hermite basis functions and the R-R peak-to-peak periods. The experiments performed for data from MIT-BIH AD with 7 arrhythmia types showed very good quality, where the classification error was 2.12%. It's expected that when combining this classifier with others using integration techniques, we could achieve better results, but we need simple classifier, especially for hardware implementation, then the HAT2 FS is a very good choice.

References

[1] Liu W., Huang Q., Chang Sh., Wang H., He J., Multiple-feature-branch convolutional neural network for myocardial infarction diagnosis using electrocardiogram, Biomedical Signal Processing and Control, vol. 45, pp. 22-32, 2018. https://doi.org/10.1016/j.bspc.2018.05.013

[2] Śmigiel S., Pałczyński K. and Ledziński D., ECG signal classification using deep learning techniques based on the PTB-XL dataset, Entropy (Basel). 2021 Aug 28;23(9):1121. https://doi.org/10.3390/e23091121

[3] Andreotti F., Carr O., Pimentel M., Mahdi A., De Vos M., Comparing feature-based classifiers and convolutional neural networks to detect arrhythmia from short segments of ECG, IEEE Computing in Cardiology (CINC), pp. 1-4, 2017. https://doi.org/10.22489/CinC.2017.360-239

[4] Sayantan G., Kien P.T., Kadambari K.V., Classification of ECG beats using deep belief network and active learning, Medical & Biological Engineering & Computing, 56 (10) pp. 1887-1898, 2018. https://doi.org/10.1007/s11517-018-1815-2

[5] Wu M., Lu Y., Yang W. and Wong S.Y., A study on arrhythmia via ECG signal classification using the convolutional neural network, Front. Comput. Neurosci. 14:564015, 2021. https://doi.org/10.3389/fncom.2020.564015

[6] Osowski S., Linh T.H., Markiewicz T., Support vector machine based expert system for reliable heart beat recognition, IEEE Transactions on Biomedical Engineering, vol. 51, no. 4, pp. 582-589, 2004. https://doi.org/10.1109/TBME.2004.824138

[7] Odinaka I., Po-Hsiang Lai, Kaplan A. D., O'Sullivan J. A., Sirevaag E. J., Rohrbaugh J. W., ECG biometric recognition: a comparative analysis, IEEE Transactions on Information Forensics and Security, vol. 7, no. 6, pp. 1812-1824, 2012. https://doi.org/10.1109/TIFS.2012.2215324

[8] Zahra Ebrahimi, Mohammad Loni, Masoud Daneshtalab and Arash Gharehbaghi, A review on deep learning methods for ECG arrhythmia classification, Expert Systems with Applications: X, Volume 7, 100033, 2020. https://doi.org/10.1016/j.eswax.2020.100033

[9] Castillo O. and Melin P., Type-2 Fuzzy Logic - Theory and Applications, Springer-Verlag, Berlin Heidelberg, 2008. https://doi.org/10.1007/978-3-540-76284-3

[10] Zadeh L.A., The conception of a linguistic variable and its application in approximate reasoning - I, Information Science, vol. 8, pp. 199-249, 1975. https://doi.org/10.1016/0020-0255(75)90036-5

[11] Ho N. C. and Wechler W., Extended hedge algebras and their application to fuzzy logic, Fuzzy Sets and Systems, vol. 52, pp. 259-281, 1992. https://doi.org/10.1016/0165-0114(92)90237-X

[12] Linh T.H., Nam P.V. and Nam V.H., Multiple neural networks integration using binary decision tree to improve the ecg signal recognition accuracy, Applied Mathematics and Computer Science, vol. 24, no. 3, pp. 647-655, 2014. https://doi.org/10.2478/amcs-2014-0047

[13] Khang T. D., Phong P. A., Dong D. K. and Trang C.M., Hedge algebraic Type 2 fuzzy sets, in Proc. FUZZ-IEEE 2010, in conjunction with the WCCI 2010, Spain, pp. 1850-1857, 2010. https://doi.org/10.1109/FUZZY.2010.5584108

[14] Haykin S., Neural Networks: A Comprehensive Foundation, 2nd ed., Prentice-Hall, Englewood Cliffs, NJ, 1999.