Two strategies for Rough Set Classifier in Data Stream

Li Cheng, Luo Jiehao, Qu Shijun, Wei Yidong

1.Introduction

In data mining, the extraction of decision rules is an important topic. Given condition attributes, we want to know which combinations of them will lead to which decisions. For example, what symptoms allow doctors to make a definite diagnosis, what kinds of credit records will make a client eligible for a loan and what patterns of sunspots could cause solar flares. To solve problems of this type, rough set theory[1] has been widely used, which mines decision rules by calculating lower and upper approximations. However, a majority of rough set applications focus on dealing with static data, which becomes more and more impractical as the volume of data is increasing significantly. When the data is a stream instead of being stored on disks, we need a technique to effectively capture each batch of it and to dynamically update the mined rules. This is our work, a system that combines rough set rule mining and data streams processing.

Rough set algorithms suffer from the situations where attributes are numerical instead of nominal. We usually have to discretize them before mining, and it gets worse with data streams as each batch has its own intervals of discretization. For example, there are 2 intervals in 2 batches, namely [1, 10] and [2, 11]. Although they are mostly overlapped, if a decision rule says the former will lead to a decision, the latter will be abandoned completely, which influences the accuracy of prediction. To deal with this problem, we develop a version using fuzzy rough set, which allows us to skip the discretization step and therefore generates better outcomes.

Furthermore, rough set algorithms are very sensitive to outliers. However, in our system, by separating a data stream into batches, it is improved to some extent. The reason is that when we mine all data as a whole, the outliers affect the entire mining process, but if we divide it into batches, outliers will only influence the mining process within the batch to which they belong. After the whole mining is done, we used a “voting” mechanism to prune the rules with low support.

Last but not least, existing algorithms dealing with dynamic data tend to update the results simply incrementally, which ignores the phenomenon of “concept drift”. Some rules’ effectiveness decreases over time, and we don’t want to treat them equally with the more recent ones. Instead, it would be better to set a time fading factor to give newly mined rules more weights. This is also implemented as a model in our system.

To summarize, we dealt with data streams using the 3 models of window sliding[13], landmark[12] and time fading[12], divided data into batches, mined each of them separately using either rough set or LEM2 algorithm, predicted test data and examined the prediction accuracies.

Regarding our contributions, generally speaking, we combined rough set theory and data streams mining. Since data are separated into batches, methods needed to integrate together the prediction results from each batch, thus we proposed 2 strategies and one of them also improves the problems caused by outliers. By applying time fading model, one of our models reflects “concept drift”. And since the conditional attributes in the decision table could be numeric, we also proposed a variant of fuzzy rough set that skips the discretization step.

The structure of this report is as follows: section 2 will introduce some background; section 3 will list related work and their disadvantages; section 4 will be the details of our proposed system. Section 5 will cover our proposed fuzzy rough set variant. Section 6 will be our experiment and the comparison with other existing models. Section 7 will conclude our paper and outline future work.

2. Background

In this section, we briefly review some background that are related to our project, including some definition in rough set theory, in fuzzy rough set theory and some related algorithms.

2.1 Rough set theory

2.1.1 An Example of Rough set system

Proposed by Polish computer scientist Zdzisław I. Pawlak in the early 1980s, rough set is a widely used technique to deal with uncertainty and to mine from imperfect knowledge. The resulting rules can tell us what combinations of conditions will definitely or probably lead to what decisions.

In rough set theory, objects characterized by the same information are considered indistinguishable. Consider we have some objects and want to classify them by colors, then we may put them into two categories, black and white, but those with the same color are indistinguishable between each other. If we introduce the concept of shape, we can divide them further based on if they are black and square, black and round, white and square or white and round, but those with both the same color and shape are still indistinguishable. As more and more information gets involved, we can differentiate them more and more precisely. However, in real life, we can only consider a limited amount of knowledge, which means that there are always some objects that cannot be divided, and in this case, we say they belong to the same elementary set, which is a basic atom of knowledge about the problem domain.

When using rough set theory to mine rules, we divide the universe into two categories: condition attributes and decision attributes. Table 1 is an example proposed by Zdzisław Pawlak in [1], in which “Coal”, “Sulfur” and “Phosphorus” are condition attributes and “Cracks” is the decision attribute. It is used to predict what combinations of the three chemical elements will lead to cracks on pipes.

Pipe	Coal	Sulfur	Phosphorus	Cracks
1	high	high	low	yes
2	avg.	high	low	no
3	avg.	high	low	yes
4	low	low	low	no
5	avg.	low	high	no
6	high	low	high	yes

Table 1

Firstly, we decide which condition attributes to consider. We could choose all three of them, but we could also choose a subset of them like “Coal” and “Sulfur” or “Sulfur” and “Phosphorus”. Then we calculate all elementary sets based on the chosen condition attributes. For example, if we pick them all, then {1}, {2, 3}, {4}, {5}, {6} are the elementary sets.

Then we pick a class of the decision attribute that we want to predict. In the example above, we could pick “yes” or “no” in the “Cracks” column. And the next step is to form a set of all records with the decision attribute of the chosen class, which is called “sample subset”. If we have chosen “yes” in Table 1, the sample subset will be {1, 3, 6}.

Then we want to compare each elementary set with the sample subset. If an elementary set is a subset of the sample subset, it belongs to “the lower approximation”, which means its condition attributes will **definitely** lead to the decision; if an elementary set has a non-empty intersection set with the sample subset, it belongs to “the upper approximation”, which means its condition attributes will **probably** lead to the decision. In the example above, the lower approximation is {1, 6} and the upper approximation is {1, 2, 3, 6}.

2.1.2 Preliminaries in Rough Set theory

In this subsection we will only mention some definitions related to our models in rough set theory, including equivalence relation, approximations, and discernibility. Interested reader could consult[1,3] for more details

Let $DT = (U, A \cup D)$ be a decision table with nominal conditional attribute A, decision value set $D = \{d\}$, and $U = \{x_1, x_2, ...x_n\}$.

Definition 1:(equivalence relation) An equivalence relation defined by attribute subset $B \subseteq A$ is called B-indiscernibility relation, denoted by IND(B), is defined by $IND(B) = \{(x, y) \in U \times U : a(x) = a(y), \forall a \in B\}$

Definition 2:(indiscernible set) For any object $x \in U$, $[x]B = \{y \in U : a(x) = a(y), \forall a \in B\}$

Definition 3:(approximation) Given an attribute subset B ⊆ A, the lower approximation of B, B↓, and upper approximation of B, B↑, regarding concept set X is defined as:
$B\downarrow(X) = \{x : [x]B \in X\}$, those x such that all the elements in $[x]_B$ is in X.
$B\uparrow(X) = \{x : [x]B \cap X \neq \emptyset\}$, those x such that at least one element in $[x]_B$ is in X

Definition 4: Suppose $U = \{x_1, x_2, ...x_n\}$, a n × n matrix $M(c_{ij})$, is defined as a discernibility matrix of $DT = (U, A \cup D)$ if:

$$c_{ij} = \{a \in A : a(x_i) \neq a(x_j)\} \text{ if } d(x_i) = d(x_j) \text{ and } c_{ij} = \emptyset \text{ otherwise}$$

Definition 5: The core in decision table is defined as: $Core_D = \{a : a \in c_{ij}; \, for\ all\ c_{ij} \neq \emptyset \}$

Definition 6: The discernibility of attribute a is defined as $Dis(a) = \{(x_i, x_j) \in U \times U : a \in c_{ij}\}$ and the discernibility of A is: $DIS(A) = \bigcup_{a \in A} Dis(a)$

Definition 7: an attribute subset $Red \subseteq A$ that preserves the same discernibility as A, is called relative reduct and it satisfies: $\bigcup_{a \in Red} Dis(a) = Dis(A)$ in the context of discernibility matrix

2.2 fuzzy rough set

Fuzzy rough set[cite] is a variant of rough set which treat the real value as a member in fuzzy set to preserve the relationship among real value. As in the project we also proposed a variant of fuzzy rough set to handle attribute selection and decision rules induction in the environment of dynamic data. This subsection will introduce some relevant definitions in fuzzy rough set.

Definition 8: (fuzzy equivalence relation) For each condition attribute a ∈ A, a binary relation R_a , which is called a fuzzy equivalence relation if R_a is reflexive, (R_a(x, x) = 1), symmetric, (R_a(x, y) = R_a(y, x)) and sup-min transitive(R_a(x,y) <= R_a(x,z) + R_a(z,y)) .

Definition 9: (approximation in fuzzy rough set) the lower and upper approximate of attribute set B, B ⊆ A, regarding the concept set X is defined as:

$$(B{\downarrow}X)(x) = inf_{u \in U} max\{1 - R_B(x,u), X(u)\}$$

$$(B \uparrow X)(x) = sup_{u \in U} min\{R_B(x,u), X(u)\}$$

Here, $X(u)$ denote the membership degree of u to a fuzzy set X and $R_B(x,u)$ denote the fuzzy equivariance relation between x and u regarding attribute set B. And $R_B(x,u) = T_{a \in B}(R_a(x,u))$, where T is a t-norm aggregation function in fuzzy set theory.

One would notice that the definitions of lower and upper approximate is different from the one in the crisp rough set. These definitions are representing the membership degree of x to such lower and upper approximate instead of representing a set of elements.

In the contest of decision table, we mostly concern about the lower and upper approximate membership degree of each $x \in U$, regard to its decision class. That is:

$$(A{\downarrow}[x]_D)(x) = inf_{u \in U} max\{1 - R_A(x,u), [x]_D(u)\}$$

$$(A \uparrow [x]_D)(x) = sup_{u \in U} min\{R_A(x,u), [x]_D(u)\}$$

In the assumption of our models, decision attribute is always crisp (nominal), then the membership of u to $[x]_D$ becomes: $[x]_D(u) = 1 \ if \ d(x) = d(u) \ and \ [x]_D(u) = 0 \ if \ d(x) \neq d(u)$. And the definition could be refined as:

$$(A{\downarrow}[x]_D)(x) = inf_{u \notin [x]_D} max\{1 - R_A(x,u)\}$$

$$(A \uparrow [x]_D)(x) = sup_{u \in [x]_D} min\{R_A(x,u)\}$$

The refined lower approximate membership function will be used in our membership rough set models to detect the membership degree of object belong to its decision class's lower approximate and to making predictions, whereas the binary fuzzy relation is replaced by similarity measures between attributes, which will be covered in section 5.

2.3 relevant algorithm:

In this subsection we will give a brief introduction of some existing algorithms that are included in our project, which includes K-MEANS clustering for discretization, attribute reduction by discernibility matrix[3], and LEM2[2]

2.3.1 K-Means clustering for dicertization

The traditional rough set system could only work on ordinary attributes, if there are numeric attributes in the decision table, we firstly use K-means clustering techniques to discretize the numeric attribute into different bins.

2.3.2 Attribute reduction by discernibility matrix

Discernibility matrix in definition 4 preserving all the information about the discernibility of each attribute. Based on the discernibility matrix, one reduct of the decision table could be obtained by using a hill-climbing strategy.

```
Procedure attribute reduction by discernibility matrix
input: a decision table DT
output: the reduct in the decision table
Step 1: based on definition 4, construct the discernibility matrix
Step 2: find $Dis(a), \forall a \in A$, and $DIS(A) = \bigcup_{a \in A} Dis(a)$
Step 3: Let $Core = \emptyset$ and compute $Dis(A - \{a\}), \forall a \in A$;
Step 4: select those $a_0$ such that $Dis(A - \{a_0\}) \neq DIS(A)$ and add them to $Core$
Step 5: Let $Red = Core, and\, Dis(Red) = \bigcup_{a \in Red} Dis(a)$
Step 6:For $\forall a \notin Red, Dis(a) = Dis(a)/Dis(Red)$
Step 7: $While\, Dis(red) \neq Dis(A)$:
                    Compute $Dis(Red + \{a\}),\ for\ a \notin (A - Red)$
                    Select $a_0$ such that $Dis(Red + \{a_0\})$ is maximum
                    Let $Dis(Red) = Dis(Red + \{a_0\})$ and $Red = Red + \{a_0\}$ ,
                    $\forall a \notin Red, Dis(a) = Dis(a)/Dis(Red)$
```

2.3.3 LEM2 Algorithm

Although the lower and upper approximations are based on the relative reduct, the gained rules are still not minimal. The reduct only trims unnecessary conditions, but parts of classes of the remaining conditions are just sufficient to cause a decision. Thus, to predict more precisely, we want to keep only those important classes instead of the whole conditions.

LEM2 (Learning from Examples Module, version 2), proposed by Jerzy W. Grzymala-Busse, is an algorithm for learning minimal rules from examples. In the pseudocode shown below [2], B is a nonempty lower or upper approximation of an information system, T is a set of attribute-value pairs (a, v) and 𝕋 is local covering of B.

```
Procedure LEM2
input: a set B,
output: a single local covering T of set B
begin
G := B;
𝕋 := Ø;
while G ≠ Ø
        begin
        T := Ø;
        T(G) := {t | [t] ∩ G ≠ Ø};
        while T = Ø or [T] ⊊ B
                begin
                        select a pair t ∈ T(G) with the highest attribute priority, if a tie occurs, select a pair t ∈
                        T(G) such that |[t] ∩ G| is maximum; if another tie occurs, select a pair t ∈ T(G) with the
                        smallest cardinality of [t]; if a further tie occurs, select first pair;
                        T := T ∪ {t};
                        G := [t] ∩ G;
                        T(G) := {t | [t] ∩ G ≠ Ø};
                        T(G) := T(G) – T;
                end {while}
        for each t in T do
                if [T – {t}] ⊆ B then T := T – {t};
        𝕋 := 𝕋 ∪ {T};
        G := B – $\bigcup_{T\in��}$ [T];
end {while};
for each T in 𝕋 do
if $\bigcup_{S\in��-\{T\}}$ [S] = B then 𝕋 := 𝕋– {T};
end {procedure}.
```

2.3.4 Fuzzy Rough Nearest Neighbor(FRNN)

Fuzzy rough nearest neighbor(FRNN) was proposed by Richard Jensen and Chris Cornell[15], as a classification method based on the definition of fuzzy rough set' upper approximation and lower approximation. It firstly calculate the K nearest neighbors of the target object then using those found nearest neighbors as rule to make predictions on target object by definition of lower and upper approximation.

```
precedure FRNN
input: $X$ the training data set, $D$ the decision class, and $y$ the object need to be classified
output: classification for y
begin
        $N \leftarrow$ the top K nearest neighbor of y
        $\gamma = 0, Class \leftarrow \emptyset$
        for each $d \in D$ do :
            if $(R\downarrow D)(y) + (R\uparrow D)(y) > \gamma$ :
                $Class \leftarrow d$,
                $\gamma = (R\downarrow D)(y) + (R\uparrow D)(y)$
            endif
        endfor
        output Class
end
```
----

3. Related work

There are plenty of work using rough set theory to solve problems. Salvatore Greco, Benedetto Matarazzo and Roman Slowinski applied rough set theory to evaluate bankruptcy risk, and the resulting rules provided guidance about financing newly established firms[4]. Seungkoo Lee and George Vachtsevanos investigated a method[5] to identify defects on automotive glass and therefore helped manufacturers improve financial loss. G. Lambert-Torres published an article[6] about assisting power system operators to take a decision about an operation based on mined rules. Those are a fraction of the typical applications of rough set theory.However, they all have a limitation, which is being unable to deal with data streams. When the data is flowing instead of being preserved on disks, we need other manners to capture and mine it dynamically.

Existing methods developed for data streams mining include: in [7], the authors divided the information system into subspaces, updated the equivalence feature matrix within each subspace and renewed the approximations incrementally; [8] investigated an incremental attribute reduction based on information entropy; in [9], the authors proposed to dynamically update reduct based on the positive and negative region of the decision attribute and [10] introduced RRIA, a tree based incremental features selection method in a rough set system. Although they can process data streams, the manners mentioned in the last paragraph have another disadvantage, which is that their way to update the information system is simply incremental, meaning they treat old data and new data equally. Nevertheless, old data could "go bad" and become less and less representative over time. For example, it is not likely for data about income levels from decades ago to be able to guide nowadays' policy makers, because the economy has grown so much. If we don't apply a mechanism to reflect it, we cannot capture the phenomenon of "concept drift". Since stale data is still preserved in the system and play a role in the mining of new rules, the accuracy of prediction may be influenced. Furthermore, it has been observed that when mining using rough set, the outcome is sensitive to outliers[11]. Outlier could significantly affect the processes of discretization, attribute reduction, and induction of decision rules. All works above did not deal with outliers properly, and outliers will play a role in the whole process of mining.

The issues described above are meant to be improved in our system, which uses rough set to mine data streams. The three models of data streams mining it uses are as follows.

- Window Sliding Model[13]:

  In this model, we set a fixed capacity for our mining. At anytime, we only capture and mine data that fulfill this capacity. And when new data flows in, old data will be abandoned, which it is like a window sliding through the data stream.

- Landmark Model[12]:

  Unlike window sliding model which goes through all data but only mines a certain amount of it, landmark model receives only a part of the whole data, usually starting from a point of time towards the end, but mines it all.

- Time Fading Model[12]:

  Unlike the 2 models mentioned above which mine all the data equally, this model put a weight to different batches of data. Because some data goes bad over time, we want to set a factor to reduce the significance of stale data and to weaken its impact on the process of rules mining.

4. Proposed Models

4.1 Our proposed fuzzy rough set prediction method

This section will cover the details of our proposed fuzzy rough set variant. The motivation for creating this variant is that in rough set system, real value must be categorized or discretized before processed and the equivariance relation is too rigorous, resulting in loss of similarity and connectivity among real value attribute. On the other hand, one must decide the number of clusters for dicertization.

4.1.1 discernibility matrix construction and attribute reduction

We adopt the fuzzy discernibility matrix technique to find the reduct attributes.The similarity measures we could use for numeric attributes in our models are gaussian kernel similarity or min-max scale similarity which is defined as:

$$sim_a(x,y) = exp(-(x-y)^2/(2*\sigma_a^2)) \quad (1)$$

$$sim_a(x,y) = \frac{|a(x)-a(y|)}{|a_{max}-a_{min}|} \quad (2)$$

The similarity aggregation method used in our model is following the general practice :

$$sim_A(x,y) = min_{a\in A}\{sim_a(x,y)\}$$

4.1.2 Rule mining in FRS

As the numeric attributes are not discretized in FRS,it is impossible to mine a specific rule like what we did in rough set system. To make prediction in FRS, one could compare the predicted object with nearest objects exist in the decision table and find its membership degree to each decision class's lower approximation and upper approximate, and pick the decision that with the maximum sum[15]. However, as the dataset gets larger and larger, such prediction strategy will be time consuming in the context of big data or data stream, since it requires to make comparisons with every instance to find nearest objects. As a result, we proposed a method to select the representative object in the FRS as ruling objects and keep them static in the system. And find the similarity of the predicted object with ruling object instead of all the objects in the decision table. The proposed method might take time in attribute reduction step, however, in the long run environment, it could save unnecessary time finding nearest neighbor.

To find the ruling objects in the decision table, we use K-means/medoids clustering algorithm to find preassigned N percent number of centroids/medoids in each decision class and the found centroids/medoids are acting as rules when we make predictions to new objects.

```
--------------------------------------------------------------------------------
procedure for finding rule objects in decision table
input: decision table DT, maximum ratio of rule object for each decision N
output: |$[x]_D$|*N rule objects in each decision class d
Step 1: let rule_objects be an empty dictionary
Step 2: for each decision d: using k-means clustering algorithm to find |$[x]_D$|*N cluster centers $centers$
Step 3: rule_objects[d] = $centers$
--------------------------------------------------------------------------------
```

4.1.3 prediction in FRS

Analogy to FRNN[15], which is introduced in background , we use the same mechanism to make predictions on new target. But instead of comparing the new targets with its nearest neighbors, we compare the target with the rule we mined in subsection 4.1.2.

```
--------------------------------------------------------------------------------
procedure for finding rule objects in decision table
input: target object X, rule objects
output: predicted decision for X
Step 1: $best\_decision = \emptyset$ , $m_{max}$ = 0
Step 2: for each decision $d$:
        let $rules\_d$ be the rules for decision $d$.
        using $rules\_d$ calculate the membership degree of X to decision $d$'s upper approximation $m_{up}$ by
        definition 9, but only regard to reduct attribute

        let $rules\_d_0$ be the rules for decision $d_0, \forall\ d_0 \neq d$:.
        using $rules\_d_0$ calculate the membership degree of X to decision $d_0$'s lower approximation $m_{low}$
        by definition 9. but only regard to reduct attribute
        if $m_{max}$ < $m_{low} + m_{up}$ :
            $m_{max} = m_{low} + m_{up}$
            $best\_decision = d$
--------------------------------------------------------------------------------
```

4.2 Proposed model for stream mining in RS and MRS

4.2.1 Batch-basis strategy

Batch-basis strategy refers to making prediction in each batch locally and combined the decision in each batch to making final decision. Within each batch, LEM2 is used to mine decision rules and decision is made based on rules.

```
--------------------------------------------------------------------------------
procedure for combining decisions from each batch
input: object X that need to be predicted
output: prediction for object X
Step 1: initialize vote count and decision support for each decision as 0
Step 2: for each batch b:
            predict(X) by local decision rule and return local decision support $sup$ and local decision $d$
            (all vote count multiply by fading factor if in time fading model)
            (all decision  support multiply by fading factor if in time fading)
            vote count for $d$ increase by 1
            decision support for $d$ increase by $sup$
Step 3: choosing the decision with most vote count as final decision, if tie occur, choose the one with highest
decision support
--------------------------------------------------------------------------------
```

This procedure is also displayed in chart 1, we mined each batch, make predictions locally in each batch, return the ‘vote’ and support from each batch and select the decision with the highest support. Since we did this batch by batch, and that is why we call it “Batch-basis”. The value of this strategy is that it is outlier-bearable and it does well in capturing those concepts that drift gradually.

For example, there is a record R in test data and 5 batches in training data. After mining, we found that each of Batch 1 - 4 gave us 2 rules leading to decision “yes” and record R meets all of them. Batch 5 gave us 100 rules leading to decision “no” and record R meets all of them as well. By this strategy, since Batch 1 - 4 all predict R to be “yes” while only batch 5 predicts R to be “no”, it is 4:1, so the final decision of R will be “yes”. (If this is time fading model, we need to consider each decision’s weight.) However, if we do the prediction at once instead of on batch-basis, we will get another result. There are 108 rules in total, 8 of them say R should be “yes” while 100 of them say the opposite, thus, the final decision of R will be “no” (this is the logic of our next strategy called “Aggregated”).

It is clear that Batch 5 is an abrupt drift in the context of concept drift, or an outlier in a general sense. It generates an excess of rules to dominate the prediction process, which might not be desirable in some real world applications. With Batch-basis strategy, this effect is suppressed because no matter how many rules generated by a single batch, they are isolated within this batch and will not have impacts on other batches’ prediction process.

4.2.2 Aggregated strategy

Aggregated strategy refers to aggregate the mined rules in each batch to become a larger global set of decision rules and make predictions based on the aggregated rules. Identical to Batch-basis, each batch also use the LEM2 algorithm to induce decision rules.

```
--------------------------------------------------------------------------------
procedure for aggregate decision rule from each batch
Step 1: initialize an empty dictionary $dict$ to hold rules for different decision
Step 2: for each batch b:
        for each rule in b:
                dict[b].confidence = $\frac{dict[b].support * \beta * dict[b].confidence + b.confidenc * b.support}{dict[b].support * \beta + b.support}$
                dict[b].support = dict[b].support* β + b.support
Here β refers to the time fading factor, and β = 1 if it is in landmark model or sliding window models.
--------------------------------------------------------------------------------------------------------------------------------------------
```

One benefit of aggregated strategy over batch-basis strategy is that identical rules from different batch can be merged together hence more memory space will be saved.However, for sliding window in aggregated strategy, the above procedure cannot be used since some batch will be deleted once batch size meet reach the window size. As a result, we keep all the rules in a list and aggregate it whenever we need to make predictions, as concerning for efficiency, we use dynamic catch to catch the such aggregated rules and dispatch them only if there are new updates in the system.

As Chart 2 demonstrates, contrary to the last strategy, here we no longer predict on batch-basis. Instead, we predict the whole test dataset using rules mined from all batches **at once**. To sum up, we firstly induce the decision rules as well as their corresponding support in each batch, and aggregate the identical rules by adding up their support. Whenever the predicted object arrive, we use the aggregated rule to make predictions.

For example, if there are 3 batches in training data and assume each batch induces 2 rules:

Batch 1:
rule 1:if A = 1 then decision = 0; support = 2
rule 2:if A = 0 then decision = 1; support = 1
Batch 2:
rule 1: if B = 0 then decision = 0; support = 3
rule 2: if A = 1 then decision = 1; support = 1
Batch 3:
rule 1: if B = 0 then decision = 0; support = 2
rule 2: if A = 0 then decision = 1; support = 3

By aggregating the rules in each batch: the rules in our models now become:
Aggregated rules in landmark model:
rule 1: if A = 1, then decision = 0, support = 2
rule 2: if A = 1, then decision = 1; support = 1
rule 3: if A = 0, then decision = 1; support = 4
rule 4: if B = 0, then decision = 0, support = 5

Aggregated rules in sliding window model with window size 2:
rule 1: if B = 0 then decision = 0; support = 5
rule 2: if A = 0 then decision = 1; support = 3
rule 3: if A = 1 then decision = 1; support = 1

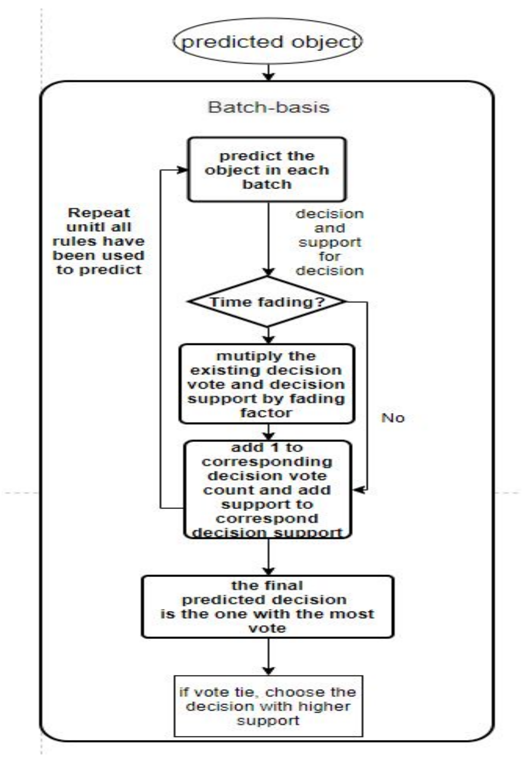
Aggregated rules in time fading model with fading factor 0.5:
rule 1: if A = 1, then decision = 0, support = 0.5
rule 2: if A = 1, then decision = 1; support = 0.5
rule 3: if A = 0, then decision = 1; support = 3.25
rule 4: if B = 0, then decision = 0, support = 2.75

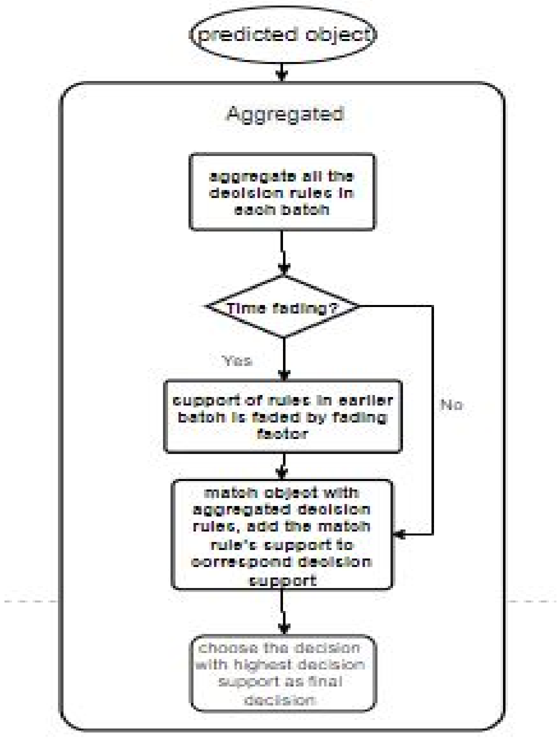
When an object X = (A:0, B:0) need to be predicted, the rules match in landmark model and sliding window are rule 3 and rule 4, and since rule 4 have higher support than rule 3, we will say X have decision 0; in the time fading model, the support of rule 3 is higher than rule 4, then we will predict X have decision 1.

The aggregated strategy, unlike the batch-basis strategy, it might not be able to capture the gradual concept dfit, whereas it tends to perform well in capturing abrupt concept drift.

chart 1
Batch-basis strategy

chart 2
Aggregated strategy


predicted object
Batch-basis
predict the object in each batch
decision and support for decision
Repeat unitl all rules have been used to predict
Time fading?
mutiply the existing decision vote and decision support by fading factor
No
add 1 to corresponding decision vote count and add support to correspond decision support
the final predicted decision is the one with the most vote
if vote tie, choose the decision with higher support


predicted object
Aggregated
aggregate all the decision rules in each batch
Time fading?
Yes
No
support of rules in earlier batch is faded by fading factor
match object with aggregated decision rules, add the match rule's support to correspond decision support
choose the decision with highest decision support as final decision

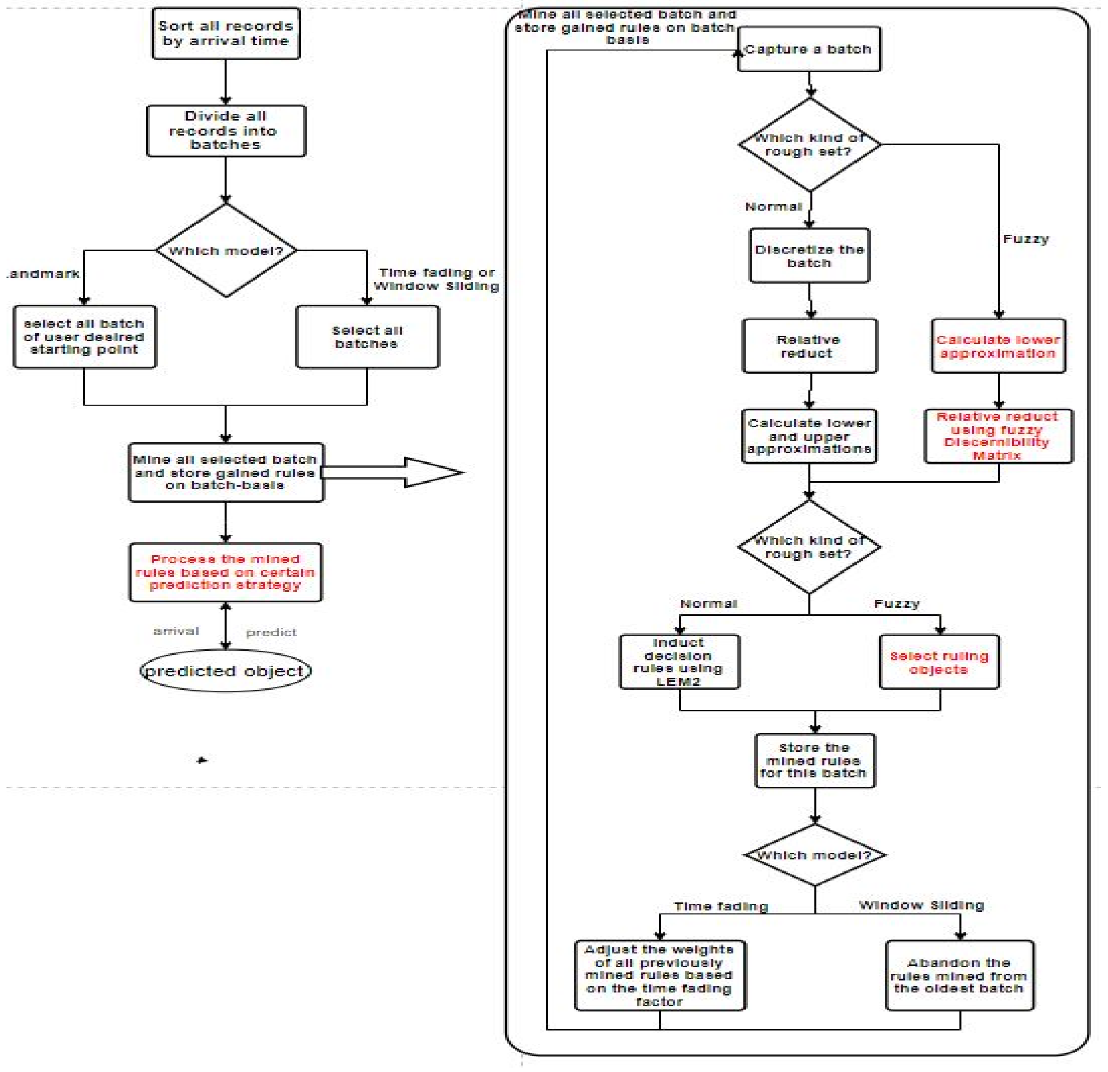
4.3 summary of proposed models as whole system

The whole process of proposed stream mining framework can be summarized in the following chart 3. The steps in red in the Chart 3, including membership rough set and two strategies for integrating prediction results generated by each batch, are explained in the last two sections, The others are common applications of corresponding techniques mentioned in section 2, interested reader could read the original publication for details.

In words, we use the traditional rough set and the proposed fuzzy rough set variant; we used 3 models for data streams mining, namely window sliding, landmark and time fading, the last is to capture “concept drift”; we have 2 stategies to integrate prediction results from each batch, one is batch-basis which improves the problem caused by outliers, the other is not batch-basis and we call it “aggregated”. In short, the whole procedure is separate the data into batches, mine rules using rough set theory and predict test data using our customized strategies.

For the convenience in the discussion of experiment, we here would like to give name for each of the proposed combination. For rough set theory with batch-strategy, we will call it incremental vote rough set,(IVRS), sliding window vote rough set(SwVRS), and time fading vote rough set(TFVRS). For our proposed membership rough set, they would be incremental vote membership rough set,(IVMRS), sliding window vote membership rough set(SwVMRS), and time fading vote membership rough set(TFVMRS). For the aggregated strategy with rough set theory, we use incremental cumulative rough set,(ICRS), sliding window cumulative rough set(SwCRS), and time fading cumulative rough set(TFCRS).Unfortunately, for the proposed membership rough set model, since each rule/ruling object is tightly connected to its local batch, we did not combine it with the aggregate strategy.

chart 3. the whole system



Sort all records by arrival time
Divide all records into batches
Which model?
Landmark
Time fading or Window Sliding
select all batch of user desired starting point
Select all batches
Mine all selected batch and store gained rules on batch-basis
Process the mined rules based on certain prediction strategy
arrival
predict
predicted object
Mine all selected batch and store gained rules on batch basis
Capture a batch
Which kind of rough set?
Normal
Fuzzy
Discretize the batch
Relative reduct
Calculate lower approximation
Calculate lower and upper approximations
Relative reduct using fuzzy Discernibility Matrix
Which kind of rough set?
Normal
Fuzzy
Induct decision rules using LEM2
Select ruling objects
Store the mined rules for this batch
Which model?
Time fading
Window Sliding
Adjust the weights of all previously mined rules based on the time fading factor
Abandon the rules mined from the oldest batch

5. Experiment

5.1 test data description

5.1.1 experiment data for proposed fuzzy rough set rule mining and predictions

To show that the proposed rule mining and prediction strategy in fuzzy rough set preserves most of the accuracy as KRNN, we use 5 datasets[14] to make comparisons to KRNN algorithm, the validation approach is identical to the authors did in KRNN, use 2 $\times$ 10 fold cross validation:

Table1. experiment data for proposed fuzzy rough set classification

	#class	#attr	#instance	#norm	#cont
wine	3	13	178	0	13
iris	3	4	150	0	4
pima	2	8	768	0	8
ionosphere	2	34	351	0	34
glass	6	9	214	0	9

5.1.2 experiment data for stream mining model

To show the scalability and performance of stream mining models. We run experiments on several stream data sets. The following table will be the detailed description about the dataset we use for experiment.

Table2. experiment data for proposed stream mining model

data name	# of numeric attribute	# of nominal attribute	# of ordinal attribute	# class	#skewness	size	initial train size	update frequency	total test size
electricity	6	0	1	2	balance	10000	2000	200	8000
rotated hyperplane 10	10	0	0	2	balance	10000	2000	200	8000
rotated hyperplane 20	10	0	0	2	balance	10000	2000	200	8000
moving RBF	10	0	0	5	semi-skew	10000	2000	50	8000
weather	8	0	0	2	balance	10000	2000	200	8000
covtype	10	44	0	7	semi-skew	50000	2000	60	48000

Since we focus on making predictions in the context of data stream and update new information as data arrival, the traditional cross validation model is not appropriate for us. Instead, we will use predict-reveal-update approach. That is, we initialize our model with the first 2000 objects/row, use the fitted models to predict the next 200 instances, record the prediction performance, then update our model by these 200 instances and make another 200 predictions. For comparison of performance, we use traditional rough set system with incremental attribute reduction combined with LEM2 algorithm, which make 1 prediction then incremental update the whole decision rules by the real decision of the predicted instance.

5.1.3 experiment data for sensitivity to concept drift type

in section 4, we claimed that the proposed batch-basis strategy is sensitive to gradual concept drift while the aggregated strategy is sensitive to abrupt concept drift. To prove our argument, we also run experiments on some artificial concept drift data[15,16,17].

Table 4. concept dfit data

	drift type	size	# class	# attr
Circles	gradual	100000	2	2
LED	gradual	100000	10	24
Sine	abrupt	100000	2	2
Stagger	abrupt	100000	2	2

5.2 experiment result

5.2.1The accuracy of our proposed prediction method in the fuzzy rough set

Table 3. comparison of proposed method to FRNN

	proposed method	FRNN
wine	90.9%	95.77%
iris	92%	96%
pima	70.39%	70.76%
ionosphere	84.77%	91.31%
glass	69.51%	71.02%

5.2.2The accuracy of our proposed model in the stream dataset

Table 4. accuracy of the proposed models in data stream environment

	IVRS	SwVRS	TFVRS	ICRS	SwCRS	TFCRS	IVMRS	SwVMRS	TFVMRS	RS+LEM2
electricity	83.33%	83.3%	83.2%	83.68%	83.95%	83.86%	82.5%	82.22%	83%	70.2%

rotated hyperplane 10	74.99%	73.25%	74.46%	76.05%	74.23%	75.94%	84.61%	84.65%	85.125%	45%
rotated hyperplane 20	56.35%	65.15%	64%	52.6%	65.15%	61.63%	55.6%	65.68%	62.2%	36.56%
movingRBF	31.7%	44.51%	43.35%	32.19%	40.58%	43.075%	45.33%	70.01%	74.775%	25.88%
covtype	69.33%	66.55%	71.6%	68.7935%	67.64%	72%	N/A	N/A	N/A	55%
weather	73.78%	73.6%	73.9%	72.475%	73.24%	72.83%	75.78%	75.26%	75.88%	62.7%

As the result shown above, our models outperform their counterparts in all of the test data.The 3 proposed membership rough set models also tend to have better accuracy when attributes are numeric, which prove our concept that using the similarity measurement could preserve more intra-relationship than discretization, thus lead to better accuracy. Furthermore, the performance of sliding window and time fading modes tend to have better performance in some data than the landmark one when the starting point is the same, which implies that the concept in the data really changes and our models could detect them well.

5.2.3 scalability of proposed models

For proving the proposed models are scalable in the context of data stream mining, Table 4 shows the total running time of the proposed models to finish the experiment above. And Table 3 is the description of experiment environment. As for fair comparison, all of the proposed rough set models use the same implementation of discernibility matrix method for attribute reduction in each batch and using the same implementation of LEM2 algorithm for decisions rule induction.

Table 3. experiment environment

implemented language	Python
cpu	intel(R) Core(TM) i7-6700 CPU @ 3.40GHz 3.40GHz
Operating System	Window 10
RAM	16GB

Table 5.Running Time of the proposed models vs incremental rough set

	IVRS	SwVRS	TFVRS	ICRS	SwCRS	TFCRS	IVMRS	SwVMRS	TFVMRS	RS+LEM2
electricity	11.31sec	10.62sec	10.34sec	8.458sec	8.982sec	8.418sec	~5mins	~5min	~5min	>40mins
rotated hyperplane 10	34.67sec	24.07sec	26..91sec	27.69sec	21.55sec	24.71sec	~9mins	~9mins	~9mins	>40mins
rotated hyperplane 20	65.62sec	57.93sec	65.07sec	51.77sec	47.25sec	54.95sec	~20mins	~20mins	~20mins	>40mins
movingRBF	43.04sec	38.98sec	37.91sec	41.27sec	34.75sec	38.93sec				>40mins
covtype										>40mins
weather	19.71sec	18.42sec	20.37sec	16.71sec	16.69sec	18.42sec				>40mins

According to the running time result, with the same core algorithms- discernibility matrix and LEM2, the proposed models perform much better than the traditional counterpart. Even though the algorithm we choose and implement might not be efficient enough, please keep in mind that any benefit of faster algorithms to traditional rough set would also apply to our models..

5.2.4 robustness to outlier of proposed models

To check the concept that our models are more robust to outliers than traditional rough set system, even in the static dataset, we use the hyperplane10 dataset with several levels of manual impose noise to make the comparison. As among our models, the LVRS and LCRS given equal weight to each batch of data, it could be used in the environment of static data, so we select them for the comparison. And the comparison result is shown in Table 5.

train size	test size
2000	500

Table 6. Accuracy of static dataset with different noise level imposed

	RS+LEM2	LVRS	LCRS
hyperplane 10 noise = 0.01	43%	64%	64%
hyperplane 10 noise = 0.02	44.8%	64%	64.8%
hyperplane 10 noise = 0.05	39.6%	62%	61.2%

hyperplane 10 noise = 0.1	37.8%	63.2%	64.6%
hyperplane 10 noise = 0.2	31.8%	57.2%	59.6%

As the above table shows, our models are more robust to outliers than the traditional rough set system.

5.2.5 experiment for gradual concept drift and abrupt concept drift

In this subsection, we run experiments to verify our argument in section 4 that the batch-basis strategy is sensitive to gradually concept drift while the aggregated strategy is sensitive to abrupt concept drift.

batch size	window size	fading factor
50	20	0.95

Table 6. comparison of proposed models in different kind of concept drift

	IVRS	SwVRS	TFVRS	ICRS	SwCRS	TFCRS
Circles	71.44%	85.13%	86.35%	74.17%	83.43%	84.31%
Stagger	67.11%	81.34%	80.96%	68.17%	94.49%	94.16%
LED	85.37%	81.86%	82.95%	85.35%	81.67%	81.67%
Sine	54.70%	81.06%	84.14%	55%	83.86%	80.50%

As the experiment result shows, only the “Stagger” dataset obviously support our argument, even though the “Circles” data shows batch-basis do slightly better in gradual concept drift. So we could not make conclusion about our argument at this point. And we leave this as an open question and in the hope that we could figure it out in future works.

6.conclusion and future work

6.1 Future work

Our promising future works will be : (1) improving the performance of the proposed fuzzy rough set classifier, including finding better similarity measurements, finding better attribute reduction algorithms; since now we using K means clustering approach to find rule objects, as K means clustering only works on numeric attributes, we will develop a better approach that could also work on the nominal attribute;(2)developing better rule induction algorithm in rough set models, taking the rule’s confidence and strength into account in addition to just the support, improving the efficiency by using faster technique instead of just the basis algorithm; (3) developing multi-levels batch models and test whether they also fit for data stream. (4) instead of classification, we will also try to apply the proposed technique for online stream data clustering, online stream outlier detection.

6.2 Conclusion

to be concluded

Reference:

[1] Pawlak, Zdzislaw. "Rough sets and data mining." *Proceedings of the Australiasia-Pacific Forum on*. 1997.

[2] Grzymala-Busse, J.W. (2003). A Comparison of Three Strategies to Rule Induction from Data with Numerical Attributes. *Electr. Notes Theor. Comput. Sci., 82*, 132-140.

[3] Skowron A., Rauszer C. (1992) The Discernibility Matrices and Functions in Information Systems. In: Słowiński R. (eds) Intelligent Decision Support. Theory and Decision Library (Series D: System Theory, Knowledge Engineering and Problem Solving), vol 11. Springer, Dordrecht

[4] Greco S., Matarazzo B., Slowinski R. (1998) A New Rough Set Approach to Evaluation of Bankruptcy Risk. In: Zopounidis C. (eds) Operational Tools in the Management of Financial Risks. Springer, Boston, MA

[5] Lee, S., & Vachtsevanos, G. (2002). An application of rough set theory to defect detection of automotive glass. Proceedings of the Institution of Mechanical Engineers, Part B: Journal of Engineering Manufacture, 216(4), 637–641.

[6] G. Lambert-Torres, "Application of rough sets in power system control center data mining," *2002 IEEE Power Engineering Society Winter Meeting. Conference Proceedings (Cat. No.02CH37309)*, New York, NY, USA, 2002, pp. 627-631 vol.1.

[7] H. Chen, T. Li, C. Luo, S. Horng and G. Wang, "A Decision-Theoretic Rough Set Approach for Dynamic Data Mining," in *IEEE Transactions on Fuzzy Systems*, vol. 23, no. 6, pp. 1958-1970, Dec. 2015.

[8] Liang, Jiye, et al. "A group incremental approach to feature selection applying rough set technique." *IEEE Transactions on Knowledge and Data Engineering* 26.2 (2012): 294-308.

[9] Hu, Feng, et al. "Incremental attribute reduction based on elementary sets." *International Workshop on Rough Sets, Fuzzy Sets, Data Mining, and Granular-Soft Computing*. Springer, Berlin, Heidelberg, 2005.

[10] Zheng, Zheng, and Guoyin Wang. "RRIA: a rough set and rule tree based incremental knowledge acquisition algorithm." *Fundamenta Informaticae* 59.2-3 (2004): 299-313.

[12] C.K. Leung, A. Cuzzocrea, and F. Jiang. Discovering frequent patterns from uncertain data streams with time-fading and landmark models. LNCS TLDKS, VIII: 174{196, 2013.

[13] Q.M. Rahman, A. Fariha, A. Mandal, C.F. Ahmed, and C.K. Leung. A sliding window-based algorithm for detecting leaders from social network action streams. In Proceedings of the IEEE/WIC/ACM WI-IAT 2015, Vol. 1, pp. 133{136. IEEE.

[14] D. N. A. Asuncion, “UCI machine learning repository,” 2007. [Online]. Available: http://www.ics.uci.edu/~mlearn/MLRepository.html

[15] R. Jensen and C. Cornelis.“Fuzzy-rough nearest neighbour classification and prediction,” Theoretical Computer Science, vol. 412, pp. 5871–5884, 2011.

[16] Pesaranghader, Ali, et al. "Reservoir of Diverse Adaptive Learners and Stacking Fast Hoeffding Drift Detection Methods for Evolving Data Streams", *Machine Learning Journal*, 2018.Pre-print available at: https://arxiv.org/abs/1709.02457, DOI: https://doi.org/10.1007/s10994-018-5719-z

[17] Pesaranghader, Ali, et al. "Fast Hoeffding Drift Detection Method for Evolving Data Streams", *European Conference on Machine Learning*, 2016.Pre-print available at: http://iwera.ir/~ali/papers/ecml2016.pdf, DOI: https://doi.org/10.1007/978-3-319-46227-1_7

[18]Pesaranghader, Ali, et al. "McDiarmid Drift Detection Methods for Evolving Data Streams", *International Joint Conference on Neural Networks*, 2018.Pre-print available at: https://arxiv.org/abs/1710.02030