A Rough Set Classifier System in Data Stream

Wei Yidong, Luo Jiehao, Qu Shijun, Li Cheng

Abstract

In the era of big data, focusing on static data is becoming more and more impractical as the volume of data is increasing significantly. Although rough set theory has been widely used as a framework to mine decision rules from information system, most of its applications are unable to process streaming data. To improve this, we present a system based on rough set theory to mine decision rules from streaming data using three models. It turns out that both prediction accuracy and performance could be improved in our system.

Key words: data streams, classification, decision rules, rough set

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, most rough set applications focus on 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 when it comes to stream data as new data might exceed the range of current intervals. Also, deciding the number of discrete intervals requires prior knowledge of the data and might not be easy. Further, discretization could lose the similarity and connectivity among real values, especially values belong to different intervals but very close to each other. To deal with these problems, we developed a version using fuzzy rough set, which allowed us to skip the discretization step and therefore to generate 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, the few existing algorithms that deal 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 or an attention window to give newly mined rules more weights or more attention. In our system, we achieved this by using a time fading model.

To summarize, we dealt with data streams using 3 models, namely 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 we chose to separate data into batches, we needed methods to integrate together the predicted results from each batch, thus we proposed 2 strategies, and one of them also improves the problems caused by outliers. By applying the time fading model and sliding window model, our models are able to reflect “concept drift”. On the other hand, the landmark model could also apply on static dataset in favor of its batch structure which could suppress the effect of outliers and more stable than traditional rough set. Further, 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 and directly mining rules from data.

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 reviewed background that are related to our project, including definitions in rough set theory, fuzzy rough set theory and 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 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.

Table 1. example of rough set information system

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

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 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. Readers 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:(discernibility matrix) 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:

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

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

Definition 6:(discernibility) 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:(reduct) 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 [21] is a variant of rough set which treats real value as a member in fuzzy set to preserve the relationship among real values. As we 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 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 approximations of attribute set B, B ⊆ A, regarding the concept set X are defined as:

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

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

where, $X(u)$ denotes the membership degree of u to a fuzzy set X and $R_B(x,u)$ denotes 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 may notice that the definitions of lower and upper approximation are different from that in crisp rough set. These definitions represent the membership degree of x to such lower and upper approximations instead of representing a set of elements.

In the context of decision table, we are mostly concerned about the lower and upper approximation membership degree of each $x \in U$, regarding 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 attributes are 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 can 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 and upper approximation membership function will be used in our membership rough set models to detect the membership degree of objects belonging to its decision class' lower and upper approximations and to make predictions, whereas the binary fuzzy relation is replaced by similarity measures between attributes, which will be covered in section 4.

Definition 10:(discernibility matrix in fuzzy rough set): According to [11], A fuzzy discernibility matrix $M(c_{ij})$ with size n × n is defined as:

$$c_{ij} = \{a \in A : 1 - R_a(x_i, x_j) \geq \lambda(x_i) \text{ if } d(x_i) \neq d(x_j)\} \text{ and } \emptyset \text{ otherwise.}$$

Readers should be careful that the discernibility matrix in fuzzy rough set is not symmetric.

2.3 Relevant algorithms:

In this subsection we will give a brief introduction of existing algorithms that are included in our project, which includes K-MEANS clustering for discretization, attribute reduction by discernibility matrix[3], LEM2[2] and fuzzy rough nearest neighbor(FRNN)[15].

2.3.1 K-Means clustering for discretization

The traditional rough set system usually only works on ordinary attributes. If there are numeric attributes in the decision table, we firstly use K-means clustering techniques to discretize them into different bins.

2.3.2 Attribute reduction by discernibility matrix

Discernibility matrix[3] in definition 4 preserves 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[3].

```
--------------------------------------------------------------------------------
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 $Red = Core, \text{ and } Dis(Red) = \bigcup_{a \in Red} Dis(a)$
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 essential classes instead of the whole conditions.

LEM2 (Learning from Examples Module, version 2) [2] is an algorithm for learning minimal rules from examples. In the pseudocode shown below, B is a nonempty lower or upper approximation of an information system, T is a set of attribute-value pairs (a, v) and $\mathbb{T}$ is a 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 – $\bigcup_{T \in \mathbb{T}}$ [T];
end {while};
for each T in 𝕋 do
if $\bigcup_{S \in \mathbb{T} - \{T\}}$ [S] = B then 𝕋 := 𝕋– {T};
end {procedure}.
```

2.3.4 Fuzzy Rough Nearest Neighbor(FRNN)

Fuzzy rough nearest neighbor(FRNN) [15] is a classification method based on the definition of fuzzy rough set's upper and lower approximations. It firstly calculates the K nearest neighbors of the target object, then uses them as rules to make predictions on target objects based on the definition of lower and upper approximations.

```
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
```

2.4 Association rules mining in data streams - 3 models

To mine streaming data, we used 3 models:

- Window Sliding Model[13]: We set a fixed number of data batches for our mining. At anytime, we only keep such number of latest data batches and mine from them. And when new data flows in and new batches are created, old data batches will be abandoned, which is like a window sliding through the data streams.
- 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 from a point of time till the end, but mines them all.
- Time Fading Model[12]: Unlike the 2 models mentioned above which mine all batches equally, this model gives each batch a weight that fades as time passes. Because some data goes bad over time, in this model, we set a factor to reduce the significance of stale data and to weaken its impact on the process of rules mining.

In our project we will combine these three models with rough set theory as well as 2 proposed prediction strategies to build rough set classifiers that work on data streams. These 3 models offer us the flexibility of handling dynamic data by simply adding or dropping specific batches and the ability to diminish the effect of outliers since outliers would only affect its local batch.

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 [4], and the resulting rules provided guidance about financing newly established firms. Seungkoo Lee and George Vachtsevanos applied rough set to identify defects on automotive glass [5] and therefore helped manufacturers minimize financial loss. G. Lambert-Torre assisted power system operators to take a decision about an operation based on mined rules from rough set system [6]. Those are a fraction of the typical applications of rough set theory. One limitation of them is that they can only handle static data. In the environment of dynamic data with concept drifting, these techniques have to be re-applied to the whole dataset every time there is new data flowing in so as to update the mined rules. However, re-applying is inefficient especially when the volume of data is large and the update frequency is high. In our system, by separating data into batches, we needed only to mine the newly created batches instead of the whole dataset.

For those techniques developed for dynamic instead of static data, they also have their own weaknesses. 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 reducts based on the positive and negative regions of the decision attribute; [10] introduced RRIA, a tree based incremental features selection method in a rough set system; in [11], Yang proposed an incremental attribute reduction mechanism for fuzzy rough set based on fuzzy discernibility matrix. Although they can process data streams, their disadvantage is that the way to update the information system is simply incremental, meaning they treated old data and new data equally. However, 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. This phenomenon is called “concept drift”, which we do need a mechanism to reflect. Since stale data is still preserved in the system and playing a role in the mining process, the accuracy of prediction may be influenced. Our system improves this problem by using a time fading model, in which a fading factor is set, so that new batches get bigger weights than older ones.

Another issue in the applications mentioned above is that they didn't handle outliers very well. Outliers could significantly affect the processes of discretization, attribute reduction, and induction of decision rules. Our system deals with this problem by separating data streams into batches and mining each of them, so that the impact of outliers is constrained within that batch.

The most related work to our project is Leung's 3 stream mining models (Window Sliding Model [13], Landmark Model [12], and Time Fading Model [12]), which were used to mine association rules in transaction data. In our project, we are meant to solve the issues mentioned in the last 2 paragraphs by combining these 3 models with rough set theory to build a rough set classifier that works on data streams. The proposed models can handle dynamic data efficiently by incrementally creating a rough set system in each new batch. It can also handle drifting concepts by forgetting the batches that contain outdated data and suppress the effects of outliers as they can only affect the mining and induction processing in the batch they belong to.

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 it is that in rough set system, real value must be categorized or discretized before being processed and the equivariance relation is too rigorous, resulting in loss of similarity and connectivity among real value attributes. Besides, one has to decide the number of clusters for discretization.

4.1.1 Discernibility matrix construction and attribute reduction

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

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

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

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

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

4.1.2 Rule mining in FRS

As numeric attributes are not discretized in FRS, it is impossible to mine specific rules as in rough set system. To make prediction in FRS, one could compare the target objects with the nearest objects existing in the decision table and find its membership degree to each decision class' lower and upper approximations based on the nearest objects, and pick the decision with the maximum sum of membership degrees[15]. However, as the dataset gets larger and larger, such prediction strategy will be time consuming, since it requires comparisons with every instance to find the nearest objects. To improve this, we proposed a method to select the representative objects in the FRS as ruling objects and to keep them static in the system, and find the similarity of the predicted objects with ruling objects instead of with all the objects in the decision table. The proposed method might take some time in training step, however, in the long run, it could save time in finding nearest neighbors.

To find the ruling objects in the decision table, we used 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 made 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 $centers$
                rule_objects[d] = $centers$
        endfor
Step 3: return rule_objects
```

4.1.3 Prediction in FRS

Analogous to FRNN[15], which was introduced in Section 2, we used the same mechanism to make predictions on new target. But instead of comparing the new targets with its nearest neighbors, we compared the targets 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 $m_{max} < m_{low} + m_{up}$:
                        $m_{max} = m_{low} + m_{up}$
                        $best\_decision = d$
                endif
        endfor
Step 3: return $best\_decision$
```

Here using only the reduct to calculate the membership function offers better time complexity and also preserves some vagueness between data. And this prediction procedure will take $O(|red| * |rule|)$ instead of O(|U|*|A|) in FRNN.

4.2 Proposed model for stream mining in RS and FRS

After combining Leung's 3 data streams mining models, we still need some mechanisms to make classifications to target objects. This subsection will introduce 2 strategies for making classifications based on the rules mined in each batch.

4.2.1 Batch-basis strategy

Batch-basis strategy refers to predicting the decision for each record in test data by using rules mined in each batch locally in training data, and combining all the decisions predicted to make the final decision for each record. Within each batch, LEM2 is used to mine decision rules, which are then used to predict local decisions.

Here, we introduce the definitions of support of rule and of support of decision for the convenience of discussion.

Definition 11:(support of rule) The support sup_r of a rule r is the number of objects that match rule r.

Definition 12:(support of decision) When predicting a target object X, the support sup_d of decision d is defined by:

$$sup_d = \sum_{r \in rules\ that\ X\ matches} sup_r$$

```
procedure for combining decisions predicted in all batches
input: target object X that need to be predicted
output: final decision for target object X
Step 1: for each decision $d$:
            $voteCount_d = 0$
            $support_d = 0$
Step 2: for each batch $b$:
            predict(X) by local decision rules and return local decision $d$ as well as its support $sup_d$.
            Let $i$ be the current index of $b$, $N$ be the total number of batches
            $voteCount_d = voteCount_d + \beta^{(N-i)}$
            $support_d = support_d + sup_d * \beta^{(N-i)}$
Step 3:return $d_0\ such\ that\ voteCount_{d_0} = max_{d \in D}\{voteCound_d\}$, if a tie occurs, choose the one with the highest $support_{d_0}$.
```

Here, $\beta = 1$ in the sliding window models and landmark models and $0 < \beta < 1$ in time fading model.

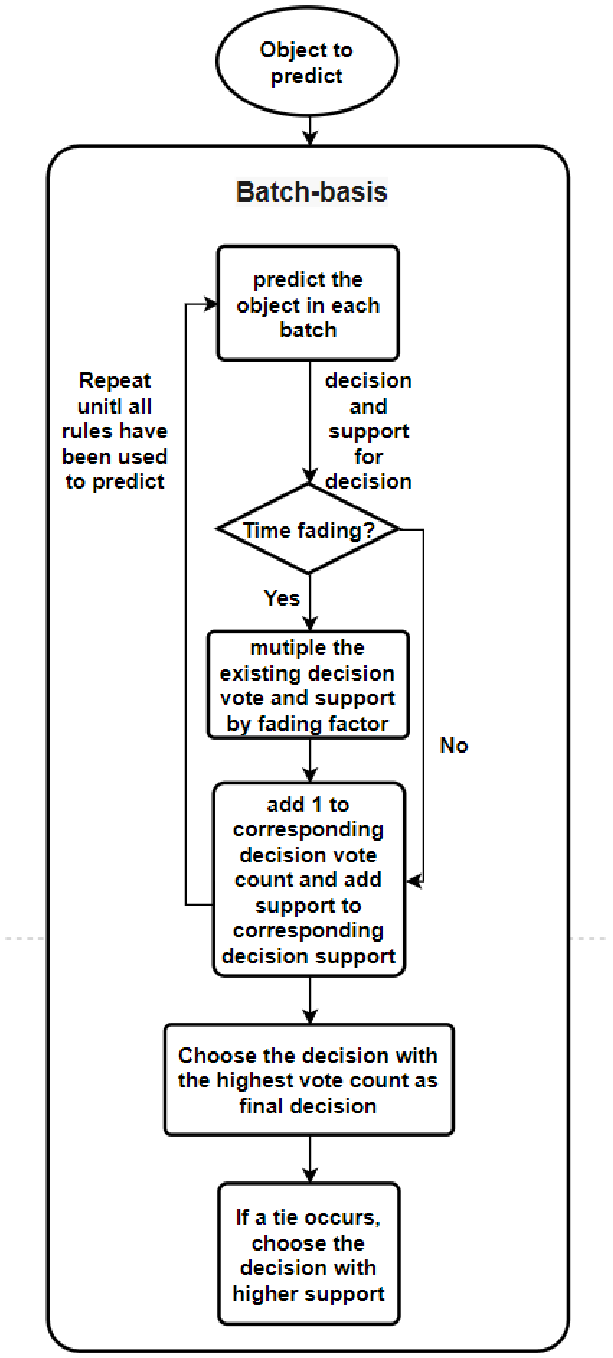

Object to predict
Batch-basis
predict the object in each batch
decision and support for decision
Repeat unitl all rules have been used to predict
Time fading?
Yes
No
mutiple the existing decision vote and support by fading factor
add 1 to corresponding decision vote count and add support to corresponding decision support
Choose the decision with the highest vote count as final decision
If a tie occurs, choose the decision with higher support

Chart 1

This procedure is also demonstrated in Chart 1. We mined each batch, predicted target object based on the rules mined from each batch in existing data, returned the ‘vote’ and support from each batch and selected the decision with the highest ‘vote’ as the final decision of the target object. We did this batch by batch, so we call it “Batch-basis”. The value of this strategy is that it is outlier-bearable and it does well in capturing concepts that drift gradually.

For example, there is a target object X and 5 batches exist in the system. After mining, we found that each of Batch 1 - 4 gave us 2 rules leading to decision “yes” and X meets all of them. Batch 5 gave us 100 rules leading to decision “no” and X meets all of them as well. By this strategy, since Batch 1 - 4 all predict x to be “yes” while only batch 5 predicts x to be “no”, it is 4:1, so the final decision of X 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 X should be “yes” while 100 of them say the opposite, thus, the final decision of X 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 aggregating the rules mined in all batches to form a global set of decision rules and make predictions based on them. Identical to Batch-basis, each batch also uses LEM2 algorithm to induce decision rules.

```
---------------------------------------------------------------------------------------------------
procedure for aggregate decision rule from each batch
Step 1: initialize an empty dictionary $dict$ to hold rules’s support, and confidence
Step 2: for each batch b:
            for each rule $r$ in b:
                dict[$r$].confidence = $\frac{dict[r].support*\beta*dict[r].confidence+ confidence_r*sup_r}{dict[r].support*\beta + sup_r}$
                dict[$r$].support = dict[$r$].support* $\beta$ + $sup_r$
    Here, $\beta$ = 1 in the sliding window models and landmark models and $0 < \beta < 1$ in time fading model.
---------------------------------------------------------------------------------------------------
```

One benefit of aggregated strategy over batch-basis strategy is that identical rules from different batches can be merged together hence more memory space will be saved. However, for the sliding window model in aggregated strategy, the procedure above cannot be used since some batches will be deleted once batch number reaches the window size. As a result, we kept all the rules in a list and aggregated them whenever we need to make predictions. Concerned about efficiency, we used dynamic cache to catch the aggregated rules and dispatched them only if there were new updates in the system.

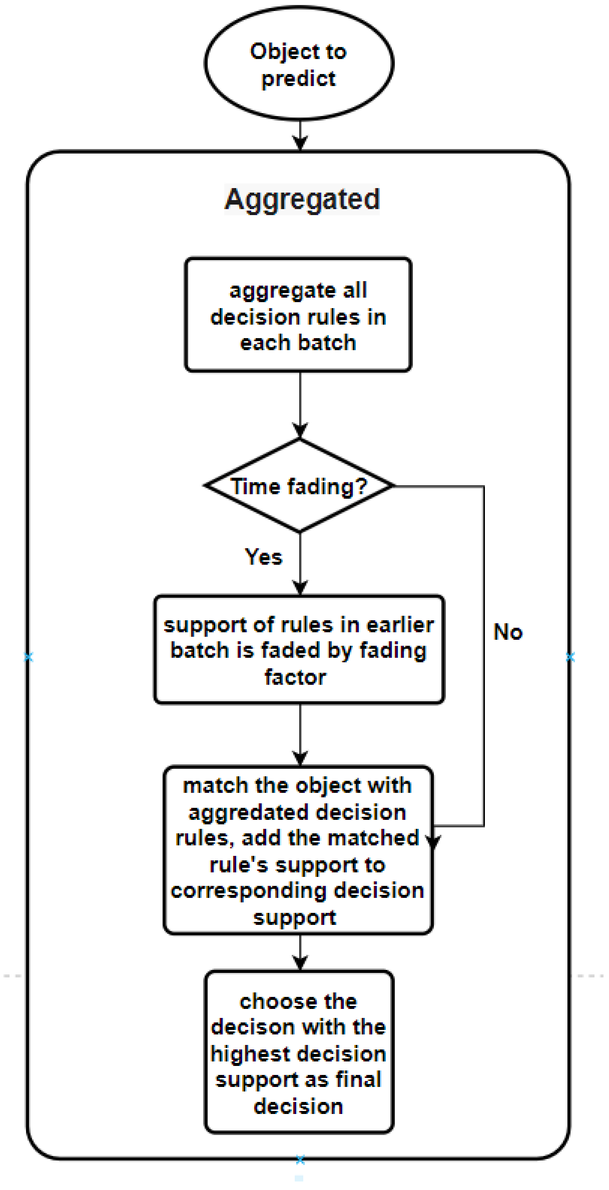

Object to predict
Aggregated
aggregate all decision rules in each batch
Time fading?
Yes
No
support of rules in earlier batch is faded by fading factor
match the object with aggredated decision rules, add the matched rule's support to corresponding decision support
choose the decison with the highest decision support as final decision

Chart 2

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

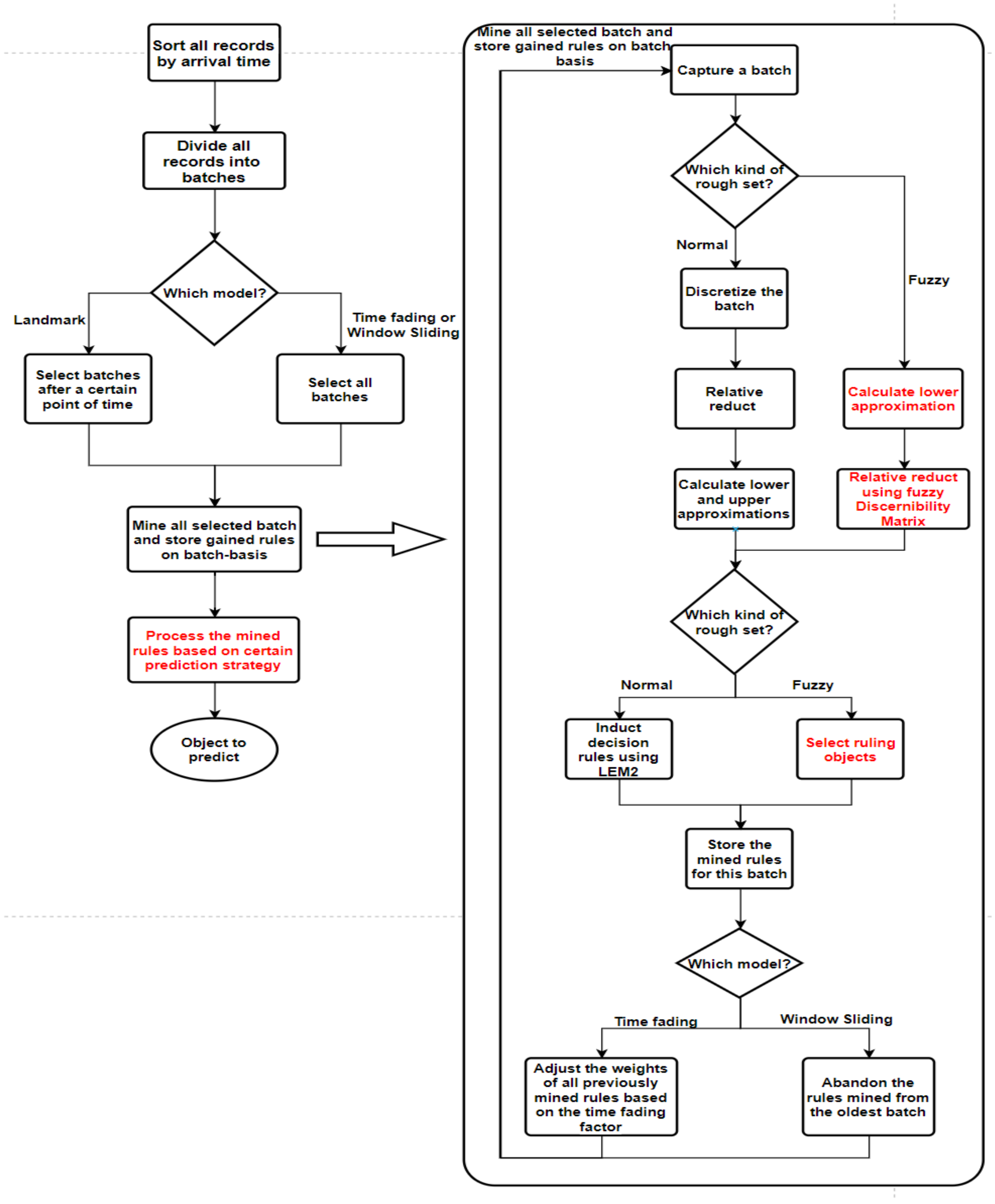
When an object X = (A:0, B:0) needs to be predicted, the rules matched in landmark model are rule 3 and rule 4(rule 1 and 2 in sliding window model), and since rule 4 has higher support than rule 3, we will say X has decision 0 (rule 1 is higher than rule 2 in sliding window, leading to the same decision ‘0’); in the time fading model, the support of rule 3 is higher than rule 4, then we will predict X has decision 1. The effect of different batch sizes, fading factors as well as window sizes will be explored in section 5.2.6.

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

4.3 Summary of proposed models as whole system

The whole process of proposed streams mining framework can be summarized in the following chart 3. The steps in red in the Chart 3, including fuzzy rough set and two strategies for integrating prediction results generated by each batch, are explained in the last two sections. The others are the corresponding techniques, such as discretization, stream mining models[12,13], attribute reduction by discernibility matrix[3] and LEM2[2] mentioned in section 2, interested readers could read the original publications for details.

In words, we used 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; we have 2 strategies 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 separating the data into batches, mining rules using rough set theory and predicting test data using our customized strategies.



Sort all records by arrival time
Divide all records into batches
Which model?
Landmark
Time fading or Window Sliding
Select batches after a certain point of time
Select all batches
Mine all selected batch and store gained rules on batch-basis
Process the mined rules based on certain prediction strategy
Object to predict
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 and upper approximations
Calculate lower approximation
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

Chart 3

5. Experiment

For convenience, we would like to give name for each of the proposed combination. For rough set theory with batch-basis strategy, we will call them incremental vote rough set (IVRS), sliding window vote rough set(SwVRS), and time fading vote rough set(TFVRS). For our proposed fuzzy rough set, they would be named as incremental vote fuzzy rough set (IVFRS), sliding window vote fuzzy rough set(SwVFRS) and time fading vote fuzzy rough set(TFVFRS). For the aggregated strategy with rough set theory, we used incremental cumulative rough set (ICRS), sliding window cumulative rough set(SwCRS) and time fading cumulative rough set(TFCRS). Unfortunately, for the proposed fuzzy rough set model, each rule/ruling object is tightly connected to its local batch, we did not combine it with the aggregate strategy. (All our proposed models shown below are in **bold.**)

5.1 Description of test data

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 4 datasets[14] to make comparisons to KRNN algorithm as well as traditional rough set with LEM2, the validation approach is identical to the authors did in KRNN, using 2×10 fold cross validation:

Table2. 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

5.1.2 Experiment data for streams mining

To show the scalability and performance of streams mining models, we ran experiments on several streaming datasets[14,19,20]. The following table is the detailed description about the datasets we used for experiment.The experiment result is presented in section 5.2.2 and section 5.2.3.

Table3. 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 /batch size	total test size
electricity	6	0	1	2	balance	10000	2000	200	8000
rotated hyperpla	10	0	0	2	balance	10000	2000	200	8000

ne 10									
rotated hyperpla ne 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 focused on making predictions in the context of data streams and updating new information as data arrives, the traditional cross validation model is not appropriate for us. Instead, we used predict-reveal-update approach. For example, we initialized our model with the first 2000 objects/rows, used the fitted models to predict the next 200 instances, recorded the prediction performance, then updated our model by these 200 instances and made another 200 predictions. For comparison of performance, we used traditional rough set system with incremental attribute reduction combined with LEM2 algorithm, which made 1 prediction then incrementally updated the whole decision rules by adding the real decision of the predicted instance into decision table.

5.1.3 Experiment data for sensitivity to different concept drift types

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 ran experiments on some artificial concept drift data[15,16,17]. The experiment result is presented in section 5.2.5.

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 results

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

Table 5. comparison between proposed prediction method and FRNN and Rough Set with LEM2(RS+LEM2)

	proposed method	FRNN	RS+LEM2
wine	90.9%	95.77%	86.05%(3 clusters for discretization)
iris	93.3%	96%	93.3%(3 clusters for discretization)

pima	70.39%	70.76%	60%(3 clusters for discretization)
ionosphere	84.77%	91.31%	82.10%(3 clusters for discretization)

As Table 3 shows, the proposed fuzzy rough set prediction method outperforms the traditional rough set in almost all of the experiment datasets even when the cluster number is manually optimized, while the accuracy tends to go down a bit compared to FRNN. However, as claimed in section 4, the proposed methods are more scalable than FRNN in the environment of streaming data. We believe these small amounts of trade off are acceptable.

5.2.2 The accuracy of our proposed model in the streaming dataset

Table 6. accuracy of the proposed models in data streams environment

	IVRS	**SwVRS**	**TFVRS**	**ICRS**	**SwCRS**	**TFCRS**	**IVFRS**	**SwVFRS**	**TFVFRS**	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 shows, our models outperform their counterparts in all of the test data.The 3 proposed fuzzy rough set models also tend to have better accuracy when attributes are numeric, which proves our concept that using the similarity measurement could preserve more intra-relationship than discretization, which leads to better accuracy. Furthermore, the sliding window and time fading models tend to have better performance in some data than the landmark model when the starting point is the same, which implies that the concept in the data really changed and our models detected them well.

5.2.3 Scalability of proposed models

For proving the proposed models are scalable in the context of data streams mining, Table 6 shows the total running time of the proposed models spent on finishing the experiment above. And Table 5 is the description of the experiment environment. As for fair comparison, all of the proposed rough set models used the same implementation of discernibility matrix method for attribute reduction in each batch and used the same implementation of LEM2 algorithm for decisions rule induction.

Table 7. experiment environment

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

Table 8.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	35.98sec	39.91sec	41.27sec	34.75sec	38.93sec	~12min	~12min	~12min	>40mins
covtype	~5min	~3min	~5min	~4min	~2.5min	~4min	N/A	N/A	N/A	>5 hours
weather	19.71sec	18.42sec	20.37sec	16.71sec	16.69sec	18.42sec	~8min	~8min	~8min	>40mins

According to the result, with the same core algorithms--discernibility matrix and LEM2, the proposed models performed much better than their traditional counterparts. However, the experiment results also imply that our models are not satisfying enough when working on data streams at this point due to the imperfect of attribute reduction process using discernibility matrix. We have decided to leave these improvements as our future work.It is very promising that more efficient reduct algorithms could be applied to our models, thus we believe there is room for improvement.

5.2.4 Robustness to outliers of proposed models

To check the concept that our models are more robust to outliers than traditional rough set systems, even when processing static datasets, we used the hyperplane10 dataset with several levels of manually imposed noise to do the comparison. Among our models, LVRS and LCRS gave equal weight to each batch of data, it could be used in the environment of static data, so we selected them for the comparison. And the comparison result is shown in Table 9.

train size	test size
2000	500

Table 9. Accuracy of static dataset with different noise level imposed

	RS+LEM 2	**LVRS**	**LCRS**
hyperplane10 noise = 0.01	43%	64%	64%
hyperplane10 noise = 0.02	44.8%	64%	64.8%
hyperplane10 noise = 0.05	39.6%	62%	61.2%
hyperplane10 noise = 0.1	37.8%	63.2%	64.6%
hyperplane10 noise = 0.2	31.8%	57.2%	59.6%

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

5.2.5 Experiment for gradual concept drift and abrupt concept drift

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

batch size	window size	fading factor
50	20	0.95

Table 10. comparison of proposed models in different kinds of concept drift

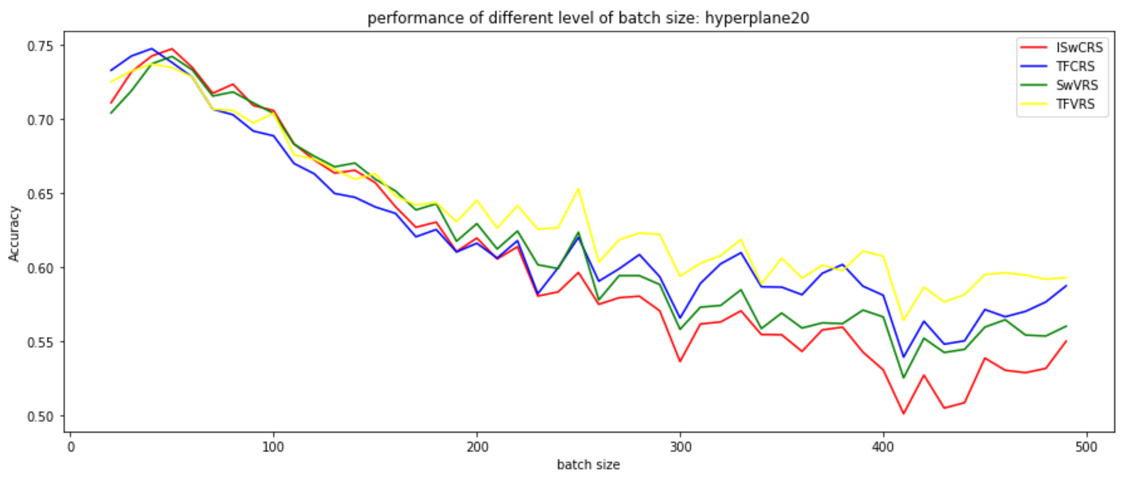
	IVRS	**SwVRS**	**TFVRS**	**ICRS**	**SwCRS**	**TFCRS**
Circles(gradual)	71.44%	85.13%	86.35%	74.17%	83.43%	84.31%
LED(gradual)	85.37%	81.86%	82.95%	85.35%	81.67%	81.67%
Stagger(abrupt)	67.11%	81.34%	80.96%	68.17%	94.49%	94.16%
Sine(abrupt)	54.70%	81.06%	84.14%	55%	83.86%	80.50%

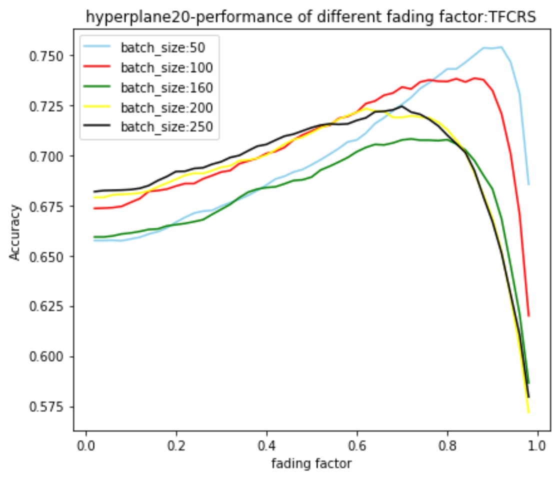
As the experiment result shows, only the “Stagger” dataset obviously supported our argument, although the “Circles” data showed that batch-basis did slightly better in gradual concept drift. Thus, we cannot make any conclusion about our argument at this point. We would like to leave this in future work.

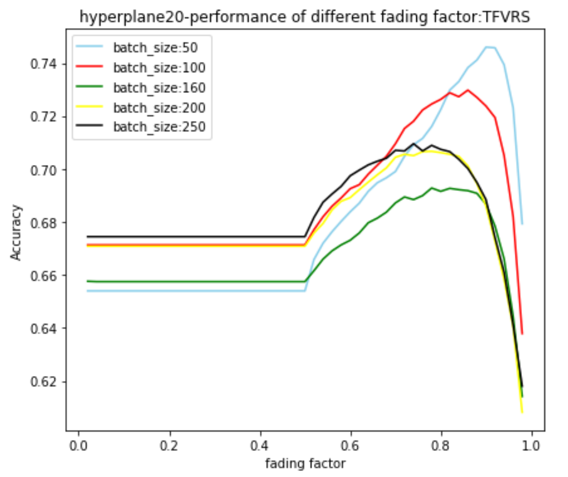
5.2.6 Experiments for effects of different hyperparameters

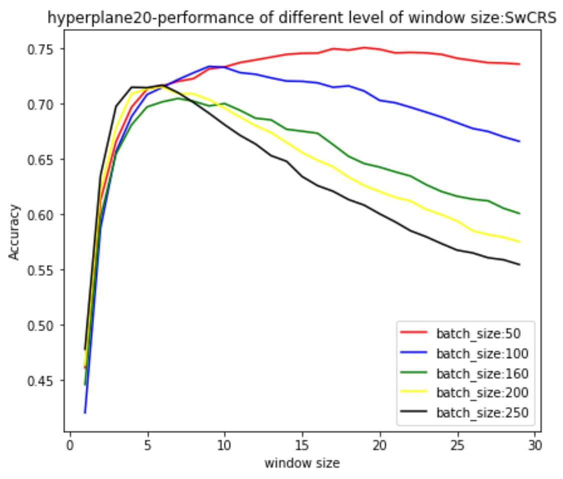
We also tested the effects of different values of hyperparameter, such as batch size, window size and time fading factor, to the prediction performance. Take batch size as an example: intuitively, if the batch is too large, it will become vulnerable to outliers like the situation in traditional rough set; if the batch size is too small, the information in each batch is incomplete and produces meaningless decision rules. To conduct the experiments, we used the ‘covtype’ dataset[20] and the ‘rotated hyperplane20’ dataset to test the performance of different levels of batch size, fading factor and window size.

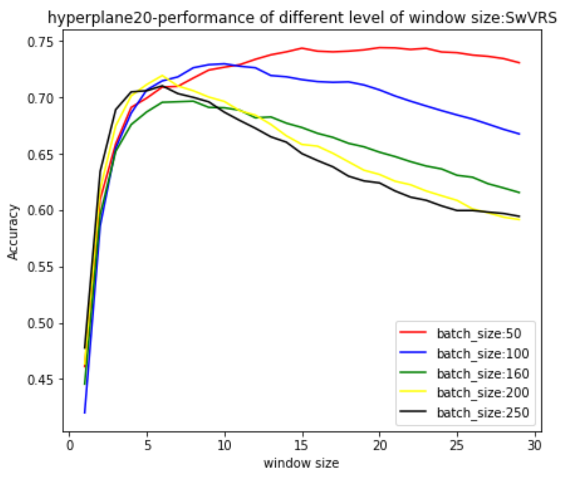
Experiment result of ‘hyperplane20’:



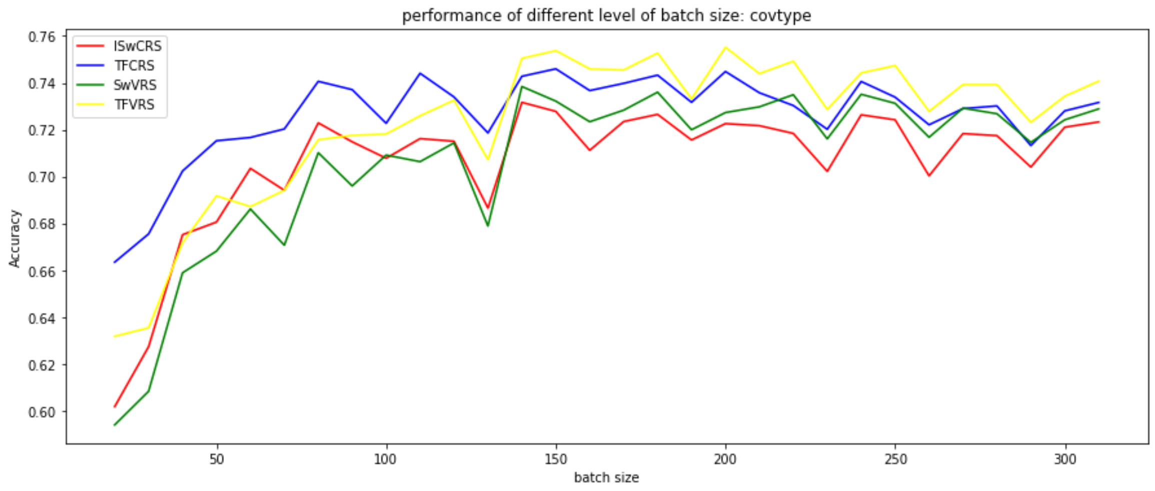


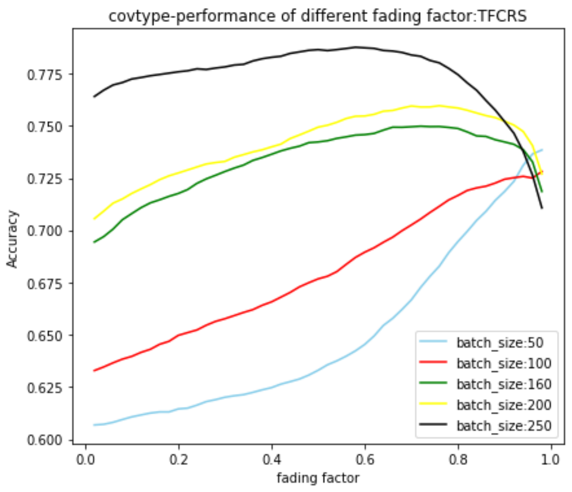


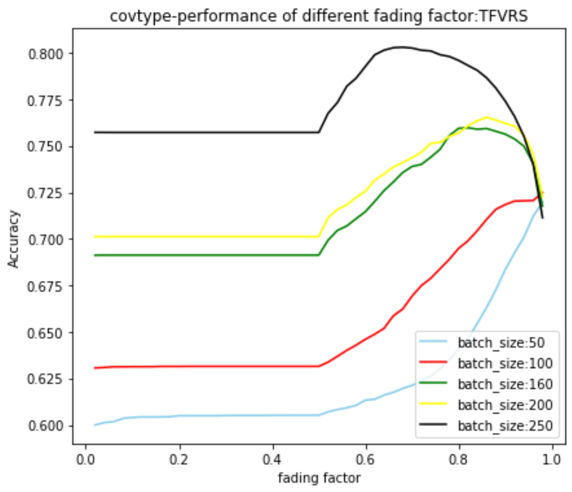


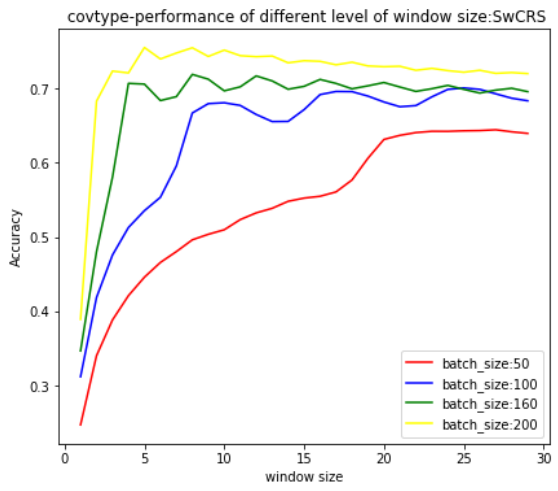


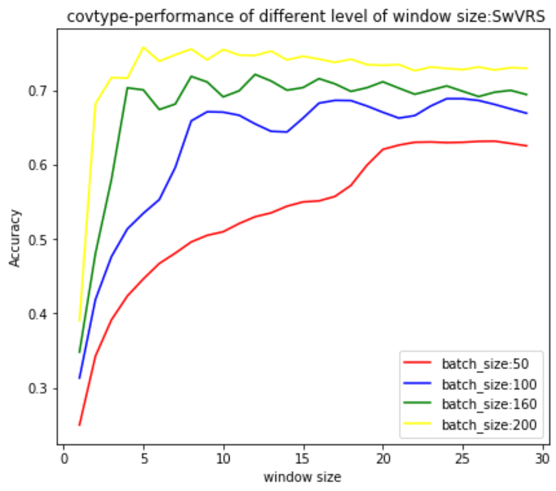
Experiment result of ‘covtype’:











As the result shows, different values of hyperparameter do matters in the performance of the models. One needs to explore the sequential characteristics of data before using the proposed models for optimal performance.

6. Conclusion

In this project, as the volumes of datasets are growing rapidly and rough set system is a powerful data mining tool but lacks the ability to handle data stream properly, we proposed stream classifier models that combine rough set system with landmark, sliding window and time fading streams mining models. Compared with existing models, the proposed models have the following advantages:

- compared with traditional rough set models, the proposed models are able to efficiently and flexibly update new observations and decision rules.
- compared with existing incremental rough set models, the proposed models are able to properly handle the phenomenon of 'concept drift' and to get more accurate decision rules for classification.
- compared with traditional rough set models, the proposed models are more robust to outliers.
- compared with traditional rough set models, the proposed fuzzy rough set models perform better when data is dominated by numeric attributes.

On the other hand, since our models introduced extra hyperparameters: batch size, window size and fading factor, which are closely related to the performance of the models, in order to optimize the performance, one needs to explore the characteristics of the data before hand.

7. Future work

Based on the above results, the promising future work will be:

- improving the performance of the fuzzy rough set models by finding better similarity measurements, and implementing better attribute reduction techniques.
- We used K means clustering approach to find rule objects, but K means clustering only works on numeric attributes, so we will try to develop a better approach that could also work on nominal attributes.
- improving our strategies by taking the confidence and strength of decision rules into consideration in addition to the support of rules and implementing advanced attribute reduction techniques to boost the efficiency then conduct more comprehensive run time experiments.
- exploring our argument that whether batch-basis strategy is good for gradual drift and aggregate strategy is good for abrupt drift.
- developing techniques that could self-adjust batch size, fading factor as well as sliding window size, so as to make our models more flexible.
- develop multi-level batch system and test performance in streaming data.

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.

[11] Y. Yang, D. Chen, H. Wang, E. C. C. Tsang, and D. Zhang, "Fuzzy rough set based incremental attribute reduction from dynamic data with sample arriving," Fuzzy Sets Syst., vol. 312, pp. 66–86, Apr. 2017.

[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

[19] Michael Harries. Splice-2 comparative evaluation: Electricity pricing. Technical report, University of New South Wales, 1999.

[20] Blackard JA, Dean DJ (1999) Comparative accuracies of artificial neural networks and discriminant analysis in predicting forest cover types from cartographic variables. Computers and electronics in agriculture 24(3):131–151

[21] D. Dubois, H. Prade, Rough fuzzy sets and fuzzy rough sets, International Journal of General Systems 17 (1990) 191–208.