

Incremental Maximal Frequent Itemset Framework for Adaptive E-learning Recommendation Systems

Emad Mahmoud Al-Azazi*, Ahmed Sultan Al-Hegami

Department of Computer Science, Faculty of Computer and Information Technology, Sana'a University, Sana'a, Yemen.
*Corresponding Author's Email: emadmalazazi@su.edu.ye

Abstract

Traditional frequent pattern mining algorithms become inefficient in dynamic e-learning environments because newly arriving learner interactions require repeated full recomputation. This study proposes the Incremental Intelligent Maximal Frequent Itemsets (IIMFIs) framework, which incrementally maintains maximal frequent itemsets while incorporating novelty-based post-generation filtering controlled by a threshold θ . Experiments were conducted on the Open University Learning Analytics Dataset (OULAD), comprising 180,596 learner-session transactions divided chronologically into four batches. At a minimum support of 0.005, the maximal-pattern representation reduced the pattern count from 2,597 frequent itemsets to 342 maximal itemsets, corresponding to an 86.8% reduction and a 7.6-fold compression. Across evaluated support thresholds, IIMFIs-Incremental achieved speedups of $2.31\times$ to $5.86\times$ relative to full recomputation. At $\text{minSup}=0.01$, IIMFIs-Incremental required 659 ms compared with 1,071 ms for FP-Growth and 60,964 ms for Apriori. At $\text{minSup}=0.005$, runtimes were comparable (1,465 ms vs. 1,447 ms), indicating that the incremental advantage is more pronounced at moderate support levels. Recommendation evaluation showed comparable $F1@10$ values between IIMFIs and FP-Growth, while $\text{Precision}@5$ was lower for IIMFIs. Recommendation diversity exceeded 0.95 for the evaluated configurations. Novelty filtering served primarily as a computational optimization. These findings indicate that IIMFIs provides a compact and efficient approach for maintaining behavioral patterns and supporting adaptive e-learning recommendations under the evaluated conditions.

Keywords: Adaptive Recommendation Systems, Educational Data Mining, Incremental Data Mining, Maximal Frequent Itemset Mining, Novelty Filtering.

Introduction

The proliferation of online learning platforms has generated volumes of learner interaction data, including clickstream logs, forum posts and assessment attempts. These data provide behavioral traces that can reveal patterns of learner engagement when mined effectively. Such patterns support personalised recommendation systems, early identification of at-risk students and adaptive curriculum design (1, 2). However, e-learning environments continuously generate new interactions, creating computational challenges for traditional pattern mining approaches. Frequent pattern mining is central to educational data mining. Apriori, as implemented in widely used data mining libraries such as SPMF (3), identifies frequent itemsets through iterative candidate generation and repeated database scans, which can become computationally expensive for large datasets. FP-Growth compresses transactions into an FP-tree and mines patterns without candidate generation, providing efficiency advantages for

static datasets (4). Eclat uses a vertical data representation and set intersections to calculate support (5). Although effective for batch processing, they require recomputation when new transactions are added. Repeated processing of historical data increases computational overhead and latency, limiting their suitability for continuously updated adaptive systems.

Unlike retail or web data, educational interactions are time-sensitive, with learner behaviours evolving across weeks and semesters. Patterns observed early in a course may become less relevant as learners progress, while new behavioural motifs emerge over time. Consequently, maintaining an up-to-date pattern representation without discarding valuable historical knowledge requires more than periodic batch execution; it demands a principled incremental strategy. A maximal frequent itemset is a frequent itemset for which no frequent superset exists. Algorithms such as MAFIA (6),

This is an Open Access article distributed under the terms of the Creative Commons Attribution CC BY license (<http://creativecommons.org/licenses/by/4.0/>), which permits unrestricted reuse, distribution and reproduction in any medium, provided the original work is properly cited.

(Received 15th June 2026; Accepted 19th September 2026; Published 17th October 2026)

FP-Max (7), negFIN (8), DiffNodesets (9), CL-MAX (10) and Mi's method (11) produce compact representations by retaining only maximal frequent patterns. However, maximal mining alone does not address incremental updating because the historical database may still need to be rescanned when new transactions arrive. Incremental methods such as FUP and FUP-tree (12, 13) update frequent itemsets using newly arrived transactions, but they retain all frequent itemsets, potentially increasing memory use and pattern redundancy. The pre-large concept (14) reduces rescanning by buffering new transactions, although it does not provide a compact maximal representation. Other tree-based methods (15), FCFP-tree (16) and fast incremental FP-Growth (17) improve update efficiency, while parallel optimization approaches (18) further address computational performance. Related variants have considered weighted maximal patterns (19), uncertain data (20) and parallel incremental frameworks (21).

Existing approaches present several limitations for continuously evolving e-learning data. First, batch-oriented methods such as Apriori, FP-Growth and Eclat process the complete database during each mining execution (3-5), causing repeated processing of historical interactions. Second, maximal algorithms such as MAFLA (6) reduce reported pattern redundancy but do not by themselves maintain discovered knowledge incrementally. Third, incremental methods such as FUP (12) and FUP-tree (13) reduce complete recomputation but may maintain large collections of frequent itemsets, particularly at lower support thresholds. Fourth, novelty- (22) or interestingness-based (23) approaches provide similarity criteria for pattern processing, but do not combine incremental maintenance with compact maximal-pattern representation. These limitations motivate a unified approach that addresses computational updating and pattern compactness simultaneously in evolving educational environments.

The research gap is the limited empirical evaluation of a workflow combining incremental pattern-state maintenance, maximal frequent-itemset representation and structural novelty-based filtering for adaptive e-learning recommendation under chronologically arriving transactions.

The aim is to develop and evaluate IIMFIs as a framework for maintaining a compact maximal frequent-pattern knowledge base in evolving educational data. IIMFIs integrates three mechanisms: incremental maintenance, which updates stored pattern state using new transactions; maximal frequent itemset mining, which retains maximal patterns; and novelty-based post-generation filtering, which controls redundant pattern retention. The novelty measure is treated only as a computational heuristic and is not intended to determine pedagogical relevance, semantic importance, or educational value. Effectiveness is evaluated across multiple support and novelty thresholds using chronologically partitioned OULAD transactions.

The objectives are:

- (a) design an incremental update procedure avoiding full historical restarts;
- (b) maintain a compact maximal itemset representation;
- (c) quantify novelty filtering effects on runtime and pattern redundancy;
- (d) evaluate performance against Apriori, FP-Growth, Eclat and full IIMFIs recomputation; and
- (e) assess whether the compact representation preserves recommendation quality and diversity.

The novelty of this study lies in integrating these three mechanisms—incremental maintenance, maximal extraction and novelty-based filtering—within a unified workflow, with empirical evaluation on chronologically ordered OULAD batches. The study distinguishes novelty as a computational filtering heuristic rather than a semantic pattern-selection or concept-drift detection mechanism.

The main contributions are:

- (i) a unified framework for dynamic pattern discovery;
- (ii) an incremental update mechanism avoiding full recomputation;
- (iii) a compact maximal-pattern representation evaluated across support thresholds; and
- (iv) an empirical analysis of computational performance and recommendation quality on OULAD.

Table 1 summarises the methodological positioning of IIMFIs against existing approaches. IIMFIs combines incremental updates, maximal itemset

representation, novelty filtering and redundancy control, whereas prior methods such as FUP, FP-Max and FUPP-tree address only subsets of these

requirements, highlighting the distinct contribution of the proposed framework.

Table 1: Methodological Comparison of Incremental and Maximal Pattern Mining Approaches

Study/ Method	Incremental Update	Maximal Item sets	Novelty Measure	Redundancy Control	Knowledge Preservation Structure	Dynamic Dataset Support	Statistical Evaluation	Referen ces
Novelty Framework	No	No	Yes	No	---	No	No	(22)
FUP	Yes	No	No	No	Frequency counts	Yes	Limited	(12)
FP-Growth	No	No	No	No	FP-tree	No	Limited	(4)
FP-Max	No	Yes	No	Partial	MFI-tree	No	Limited	(7)
FUPP-tree	Yes	No	No	No	FP-tree	Yes	Limited	(13)
Pre-large	Yes	No	No	No	---	Yes	Limited	(14)
Proposed IIMFIs	Yes	Yes	Yes	Yes	Incremental FP-tree + PD_IMFIs	Yes	Yes	—

Note: PD_IMFIs = Previously Discovered Interesting Maximal Frequent Itemsets, MFI = Maximal Frequent Itemset, FP = Frequent Pattern.

Methodology

Framework Overview

The overall architecture of the proposed IIMFIs framework is illustrated in Figure 1, which comprises offline and online pathways. The offline pathway processes historical learner sessions from the Learning Management System (LMS) to build an initial knowledge base of maximal frequent itemsets. The online pathway simulates the arrival of new learner interactions by processing chronologically ordered batches as they become available. New interactions are

preprocessed and passed to an incremental model update component. This component updates previously discovered patterns using only newly arrived data. Generated maximal frequent itemsets are evaluated using a novelty measure before being incorporated into the updated knowledge base. This evaluation is performed as a post-generation filtering step and does not alter the FP-Max generation process. The retained patterns are then incorporated into the updated pattern state and used by the recommendation component.

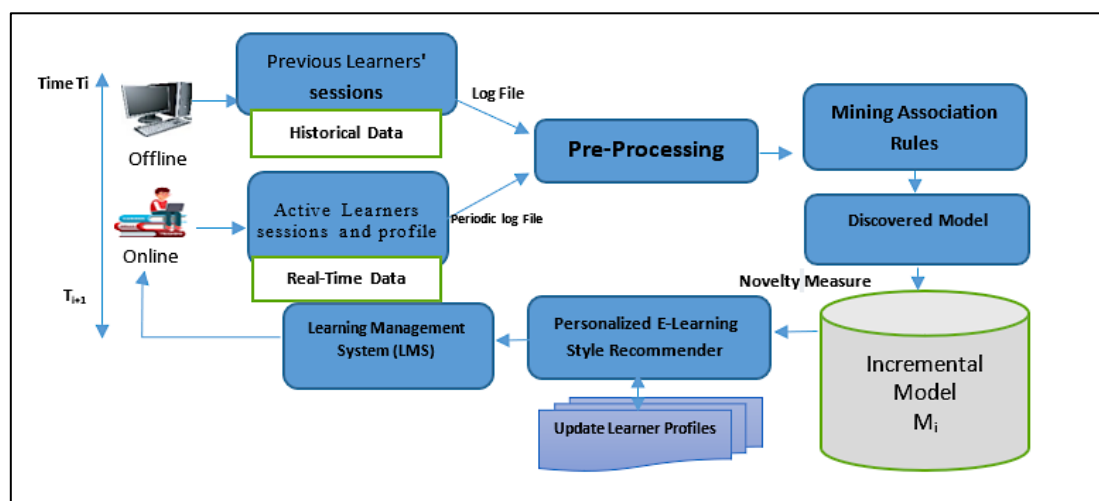


Figure 1: Architecture of the Proposed Framework

Dataset and Data Pre-processing

Raw interaction logs are cleaned and segmented into sessions. Each transaction represents the set

of distinct learning resources (e.g., course materials, assessment activities, forum interactions) accessed by a single learner within a continuous session, where a session is defined as a sequence of

activities with inter-event intervals not exceeding 30 minutes. Duplicate accesses to the same resource within a session are removed to represent binary occurrence and each resource is encoded as an integer. The resulting transactional dataset is divided into chronological batches to simulate dynamic streaming.

The Open University Learning Analytics Dataset (OULAD) contains detailed learner interactions with course materials, assessments and forums. After cleaning, the dataset comprised 180,596 transactions. Data were divided chronologically

into four batches of 45,149 transactions each. Table 2A summarizes the characteristics of the processed OULAD dataset used in the incremental evaluation. The dataset comprises 180,596 learner-session transactions, where each transaction represents a learner session. The data were divided chronologically into four batches of 45,149 transactions each, ensuring an equal distribution of data across the incremental evaluation cycles. This structure enables the simulation of dynamic streaming conditions in which new learner interactions arrive over time.

Table 2A: Summary of the Processed Dataset Characteristics

Property	Value
Dataset	OULAD
Number of transactions	180,596
Data representation	Transactional sessions
Number of batches	4
Batch size	45,149 transactions

Note: OULAD = Open University Learning Analytics Dataset, each transaction corresponds to a learner session.

Incremental Maximal Frequent Itemset Mining

IIMFIs processes the database incrementally over a sequence of data batches $D_1, D_2 \dots D_i$. When a new batch D_{i+1} arrives, the framework reuses previously discovered maximal frequent itemsets maintained in the Previously Discovered Inter-

esting Maximal Frequent Itemsets (PD_IMFIs) representation. The retained patterns and their stored support values are incorporated into an updated search structure together with the newly processed transactions.

Conceptually, for an itemset X , its support in a cumulative database of size N_{t+1} is defined as shown in Equation [1]:

$$Support_{t+1}(X) = \frac{C_{t+1}(X)}{N_{t+1}} \quad [1]$$

Where, $C_{t+1}(X)$ denotes the number of transactions containing X up to time $t + 1$ and N_{t+1} denotes the corresponding cumulative number of transactions.

This definition provides the conventional interpretation of support; however, the implementation does not recompute this

cumulative quantity directly from the complete historical transaction database.

Algorithm 1: Incremental Maintenance of Maximal Frequent Itemsets

Input:

- PD_IMFIs_{prev} – previously retained maximal frequent itemsets with their stored cumulative support counts
- new batch D_{t+1}
- minimum support $minSup$
- novelty threshold θ

Output: Updated PD_IMFIs

Step 1: Update the cumulative transaction count:

$$N_{t+1} = N_t + |D_{t+1}|$$

Step 2: Compute the cumulative minimum support count:

$$MinCount_{t+1} = \lceil minSup \times N_{t+1} \rceil$$

Step 3: For each previously retained maximal itemset $X \in PD_IMFIs_{prev}$:

a) Compute $\Delta C_{t+1}(X)$, the support count of X in D_{t+1}

b) Update cumulative support:

$$C_{t+1}(X) = C_t(X) + \Delta C_{t+1}(X),$$

c) Compute relative support:

$$Support_{t+1}(x) = \frac{C_{t+1}(X)}{N_{t+1}}$$

d) If $C_{t+1}(X) < MinCount_{t+1}$ mark X as inactive and exclude it from the updated pattern set.

Step 4: Process the new batch D_{t+1} to identify item occurrences and previously unseen items for construction of the incremental search representation.

Step 5: Construct a compressed heuristic search representation using the active retained historical maximal patterns and their stored support counts, together with transactions from the new batch D_{t+1} .

Step 6: Apply FP-Max to generate maximal frequent itemsets from the resulting search representation.

Step 7: For each generated maximal itemset Y :

a) Compute maximum Dice similarity against the previously retained historical maximal itemsets

(PD_IMFIs_{prev}) :

$$maxSim = \max_{X \in PD_IMFIs_{prev}} Dice(Y, X)$$

b) Compute novelty:

$$NM(Y) = 1 - maxSim$$

c) **If** novelty filtering is disabled, retain Y unconditionally; otherwise, retain Y only if $NM(Y) \geq \theta$.

Note: For the initial batch, when no previously retained historical patterns exist ($H = \emptyset$), novelty filtering is not applied and the generated maximal frequent itemsets are retained as the initial pattern state.

Step 8: Update PD_IMFIs with the retained maximal patterns and their available support information.

Step 9: Return updated PD_IMFIs .

Support Evolution Under Incremental Updates

When a new batch $|\Delta D_{t+1}|$ arrives, the cumulative occurrence count of a previously retained pattern X is updated as shown in Equation [2]:

$$C_{t+1}(X) = C_t(X) + \Delta C_{t+1}(X) \quad [2]$$

Where, $\Delta C_{t+1}(X)$ denotes the number of occurrences of X in the newly arrived batch.

The updated relative support is given by Equation [3]:

$$Support_{i+1} = \frac{C_t(X) + \Delta C_{t+1}(X)}{N_t + \Delta N_{t+1}} \quad [3]$$

Although the cumulative occurrence count cannot decrease when transactions are appended, the relative support may increase, decrease, or remain unchanged depending on the number of new occurrences relative to the size of the incoming batch. If the updated support falls below minSup, the pattern is excluded from the retained maximal pattern state. Example: If a pattern occurs in 100 of 1,000 transactions, its initial support is 0.10. If 200 new transactions arrive but the pattern occurs only 10 additional times, its updated support becomes $110/1200 = 0.0917$. Thus, the cumulative count increases while the relative support decreases.

Note on Novelty Based Post Generation Filtering

Novelty based filtering is applied as a post generation acceptance stage. After FP Max

generates the maximal frequent itemsets, each generated candidate pattern is evaluated against previously retained historical patterns using the Dice based novelty measure. A generated pattern is retained when its novelty score meets the threshold θ ; otherwise, it is discarded. This operation does not alter the FP Max search or candidate generation process. Therefore, novelty is used as a computational filtering mechanism for controlling the retained pattern representation rather than as a semantic, pedagogical, or formal concept drift detection mechanism. The filtering remains heuristic and does not provide a formal completeness or set equivalence guarantee.

Novelty Based Post Generation Filtering

Novelty is used to quantify the structural difference between a newly generated maximal

frequent itemset and previously retained historical patterns.

For a generated maximal itemset m and a previously retained historical maximal itemset h , the Dice similarity is defined in Equation [4]:

$$\text{Dice}(s_1, s_2) = \frac{2|m \cap h|}{|m| + |h|} \quad [4]$$

The novelty of m relative to the previously retained pattern set H is defined in Equation [5] (24):

$$NM(m) = 1 - \max_{h \in H} \text{Dice}(m, h) \quad [5]$$

Thus, a high $NM(m)$ indicates that the generated pattern is structurally different from the most similar previously retained pattern. When novelty-based filtering is enabled, a generated pattern is retained if $NM(m) \geq \theta$; otherwise, it is discarded. This filtering is applied after FP-Max generates the maximal frequent itemsets and therefore does not modify the FP-Max search process.

The value of NM lies in $(0,1)$, where 0 indicates identical itemsets and 1 indicates disjoint itemsets.

Similar novelty measures have been explored in the context of interestingness and unexpectedness (24).

In the current IIMFIs implementation, novelty is evaluated after a maximal frequent itemset has been generated by the FP-Max procedure. For a newly generated itemset Y , the implementation computes its maximum Dice similarity against the previously retained historical itemsets H , as shown in Equation [6]:

$$NM(Y) = 1 - \max_{X \in H} \text{Dice}(Y, X) \quad [6]$$

If novelty filtering is enabled, Y is retained in PD_IMFIs only when $NM(Y) \geq \theta$. If novelty filtering is disabled, the generated itemset is retained without applying the novelty condition. This behavior is implemented in the itemset-saving stage after FP-Max generation. Importantly, novelty is not used for branch-level filtering during FP-Max enumeration in the current implementation. Instead, it acts as a post-generation pattern-acceptance filter. Thus, the novelty threshold determines whether a

generated maximal pattern is incorporated into the maintained PD_IMFIs representation; it does not terminate the recursive FP-Max search based on the novelty value of an intermediate candidate.

Recommendation Generation

For a learner profile P (the set of resources the learner has accessed) and a candidate unseen resource a , the recommendation score is computed as shown in Equation [7]:

$$\text{score}(a) = \sum_{\substack{I \in IMFIs \\ a \in M}} \frac{|M \cap P|}{|M|} \quad [7]$$

Where,

P is the learner's current profile (set of previously accessed resources),

a is the candidate resource to be scored,

M is a maximal frequent itemset maintained in the IMFIs set,

$|M \cap P|$ denotes the number of resources in M that the learner has already accessed, and

$|M|$ is the size of the maximal itemset.

Top-kk resources with the highest scores are recommended to the learner. The learner profile is updated after interaction. This pattern-based approach complements existing adaptive learning path recommenders (25). Similarity measures underlying collaborative filtering have been surveyed elsewhere (26).

Time and Space Complexity

The computational advantage of IIMFIs-Incremental arises primarily from avoiding repeated processing of the complete historical transaction database rather than from a lower asymptotic worst-case complexity than FP-Growth. Let ΔN denote the number of transactions in the newly arrived batch and L the average transaction

length. The incremental update component requires approximately $O(\Delta N \cdot L)$ processing for the newly arrived transactions, together with the The overall runtime can be conceptually decomposed as Equation [8]:

$$T_{IIMFIs} \approx T_{\Delta D} + T_{candidate} + T_{novelty} + T_{maximality} \quad [8]$$

The novelty-filtering component depends on the number of generated candidates and the number of previously retained patterns. For C generated candidates and H retained historical patterns, a direct comparison implementation may require approximately $O(C \cdot H)$ similarity evaluations. Consequently, the actual computational benefit depends on the size of the retained state and the number of generated candidates.

In contrast, full recomputation repeatedly processes the complete accumulated database, whose transaction volume increases over time. Therefore, although both approaches may have similar worst-case combinatorial behavior for

cost of updating the retained pattern state and evaluating newly generated patterns.

frequent-itemset enumeration, incremental processing can reduce repeated historical data access substantially under suitable streaming workloads as discussed in Table 2B.

The space requirement is dominated by the retained maximal-pattern state, the structures required for candidate generation and the temporary representation of the incoming batch. Multi-scale FP-tree structures (27, 28) can further optimize this trade-off. Thus, the compact maximal representation can reduce the stored pattern space relative to retaining all frequent itemsets, but it does not eliminate the worst-case exponential nature of frequent-itemset mining.

Table 2B: Theoretical Comparison of Computational Characteristics (per update cycle)

Method	Main Processing per Update	Historical Rescanning	Pattern-Search Worst Case
Full recomputation	Processes entire N_t transactions	Yes	Data-dependent, exponential in distinct items
FP-Growth (batch)	Builds FP-tree from N_t + mine	Yes	Data-dependent, exponential in distinct items
IIMFIs-Incremental	Updates retained patterns using ΔN only + FP-Max on combined state	No	Data-dependent, exponential in distinct items; novelty applied as post-generation filtering

Note: The worst-case exponential complexity is common to all frequent itemset mining algorithms. The practical runtime is highly data- and support-dependent. The key distinction of IIMFIs-Incremental is that it avoids rescanning the full historical database by maintaining a compact pattern state, but it does not claim formal exactness or completeness from the compact state alone.

Ethical Considerations

This study used the publicly available and anonymized Open University Learning Analytics Dataset (OULAD) for secondary analysis. No direct interaction with human participants was involved. Therefore, ethical approval and informed consent were not required for this secondary analysis, in accordance with applicable requirements for publicly available de-identified data.

Experimental Setup

Support thresholds used are 0.05, 0.03, 0.01 and 0.005. Metrics include execution time, peak memory and pattern count.

Baseline Algorithms

Baselines are Apriori, FP-Growth and Eclat. All baselines were executed using SPMF version 2.51. IIMFIs was implemented in Java. Experiments ran

on an Intel Core i5-6300U CPU, 8 GB RAM, Windows 10.

Evaluation Metrics and Statistical Analysis

Recommendation evaluation used a user-based collaborative filtering framework with 30 nearest neighbors and five recommendations per user. We sampled 10,000 training users and 1,000 test users. Metrics were *Precision@5*, *Recall@5*, *Normalized Discounted Cumulative Gain at 5 (NDCG@5)*, *diversity* and *novelty*. For recommendation evaluation, *F1@10* is defined as the harmonic mean of *Precision@10* and *Recall@10* and is computed at a recommendation cut-off of $k=10$. Jaccard and Coverage are reported separately as pattern-level evaluation measures and are not part of the recommendation *F1@10*

definition. No separate pattern-level F1 score is computed; pattern quality is assessed directly via Jaccard and Coverage.

All experiments were executed under controlled conditions. Recommendation experiments were repeated over five independent random train-test splits, with the same split used across compared methods. Means and standard deviations for recommendation metrics were calculated across these splits. For runtime measurements, each configuration was executed multiple times and mean, standard deviation and 95% confidence intervals were computed. No formal statistical hypothesis tests were performed for all reported comparisons; therefore, p-values are not included in the main tables and confidence intervals are interpreted descriptively. Deterministic pattern counts are reported without variance.

Important note regarding the recommendation protocol: In this study, the maximal frequent patterns used for recommendation were mined from the full cumulative dataset (all learners and their interactions) for each support threshold, not from the training splits alone. This choice was made to evaluate the recommendation potential of the patterns themselves, independent of the specific training sample. However, the authors acknowledge that this constitutes a limitation: it may introduce optimistic bias because the patterns

incorporate information from users who later appear in the test set. Therefore, the recommendation results reported later in this study should be interpreted as potentially optimistic estimates of recommendation performance rather than as estimates obtained under a strict train-only evaluation protocol. Future work will adopt a stricter evaluation protocol in which patterns are mined exclusively from the training data.

Results and Discussion

Dynamic Characteristics of the Data Stream

Table 3 summarizes the characteristics of the four chronological batches used in the incremental evaluation. Each batch contains 45,149 transactions, ensuring an equal distribution of data across the incremental evaluation cycles. The number of unique activities increased gradually from 42 in the first batch to 45 in the fourth batch, indicating that the fourth batch contains three more activity types than the first. The average session length remained relatively stable, ranging from 6.1 to 6.5 activities per session. These variations suggest a gradual evolution in learner interaction behavior across the chronological sequence, supporting the use of a dynamic, incrementally updated pattern representation.

Table 3: Characteristics of the Four Chronological Batches

Batch	Transactions	Unique Activities	Avg. Session Length
1	45,149	42	6.2
2	45,149	44	6.4
3	45,149	43	6.1
4	45,149	45	6.5

Note: Avg. = Average, Session Length refers to the average number of distinct learning activities accessed by a learner within a continuous session (inter-event intervals not exceeding 30 minutes).

Overall Mining Performance and Pattern Compactness

Table 4 reports the runtime and memory usage of a single execution on the complete dataset, without averaging across multiple runs. At $minSup = 0.005$, classical algorithms yield 2,597 frequent itemsets; IIMFIs retains 342 maximal patterns, an approximately 86.8% reduction in retained

pattern count. At $minSup = 0.005$, Apriori required 182 seconds, Eclat consumed 645 MB and FP-Growth completed in 1,533 ms but output all 2,597 frequent itemsets. IIMFIs completed in 6,037 ms and used 426 MB while retaining only 342 maximal patterns. IIMFIs provides a substantially more compact maximal-pattern representation, although this compactness is accompanied by additional mining cost at some support thresholds.

Table 4: Mining Performance Comparison on the Full Dataset (Single Full Run)

Support	Algorithm	Frequent Patterns	Maximal Patterns	Time (ms)	Memory (MB)
0.05	Apriori	18	—	1,158	39.1
0.05	FP-Growth	18	—	900	39.6
0.05	Eclat	18	—	1,029	121.0

0.05	IIMFIs	—	15	1,039	121.0
0.03	Apriori	46	—	2,368	175.3
0.03	FP-Growth	46	—	756	412.8
0.03	Eclat	46	—	1,085	247.0
0.03	IIMFIs	—	31	507	247.0
0.01	Apriori	575	—	25,451	206.0
0.01	FP-Growth	575	—	758	473.0
0.01	Eclat	575	—	3,489	374.0
0.01	IIMFIs	—	160	2,011	374.0
0.005	Apriori	2,597	—	182,056	381.0
0.005	FP-Growth	2,597	—	1,533	381.5
0.005	Eclat	2,597	—	8,581	645.0
0.005	IIMFIs	—	342	6,037	426.0

Note: minSup = Minimum Support, ms = milliseconds, MB = megabytes, IIMFIs = Incremental Intelligent Maximal Frequent Itemsets, FP-Growth = Frequent Pattern Growth.

The loss of incremental advantage at very low minimum-support thresholds can be attributed to the shift in the dominant cost component of IIMFIs-Incremental. As $minSup$ decreases from 0.01 to 0.005, the number of retained maximal patterns (P) increases from 160 to 342 and the candidate space expands substantially (from 575 to 2,597 frequent itemsets). This expansion directly increases the cost of two implementation-dependent components: the novelty-filtering overhead, which scales with $O(C \cdot P \cdot L_{max})$ and the maximality-update cost, which scales with $O(P \cdot L_{max})$. Consequently, the cumulative maintenance overhead approaches the cost of a single FP-Growth execution on the complete dataset, which benefits from a highly compact FP-tree structure under dense data conditions without incurring novelty or

maximality checks. Therefore, the incremental approach is most beneficial when the candidate space remains moderate ($minSup \geq 0.01$), batch sizes are relatively small and the data stream is continuous. Its advantage diminishes under very low support thresholds, large batch sizes, or static workloads where a single batch execution suffices. Figure 2 illustrates the runtime performance of all mining algorithms across different minimum support thresholds. As shown in the Figure, Apriori exhibits exponential growth in execution time as the support threshold decreases, while FP-Growth and IIMFIs maintain significantly lower and more stable runtimes. IIMFIs achieves competitive runtime performance compared to FP-Growth, particularly at moderate support levels, while providing a substantially more compact pattern representation.

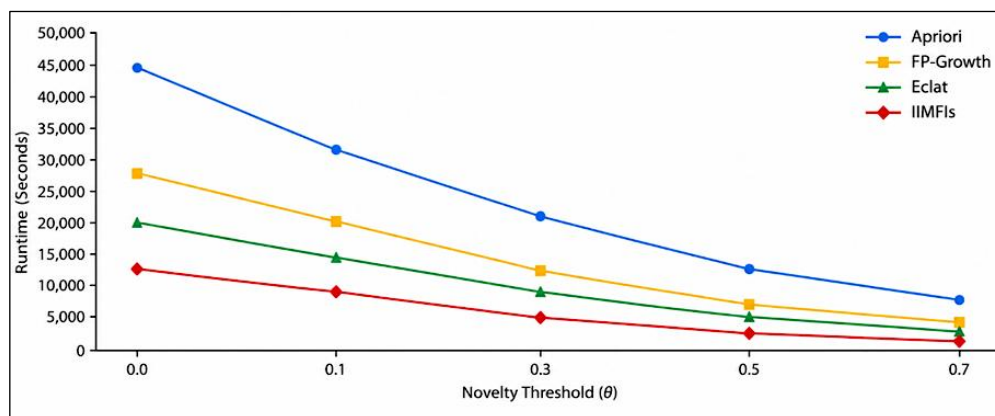


Figure 2: Runtime of Mining Algorithms at Different Support Thresholds

Figure 3 compares the relationship between runtime and the final number of retained patterns for each approach. The Figure highlights that IIMFIs achieves a favorable trade-off by producing far fewer patterns than FP-Growth, with only a

moderate increase in runtime at very low support levels. This indicates that the compact maximal representation is achieved without prohibitive computational cost.

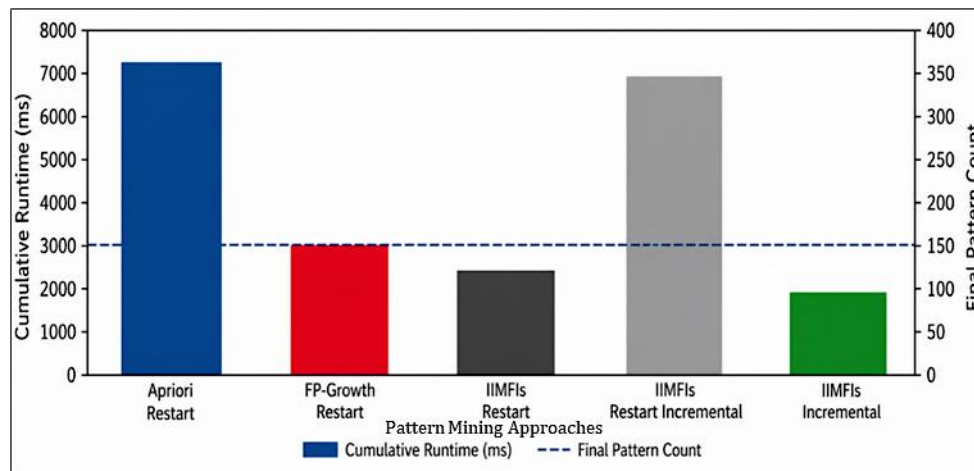


Figure 3: Comparison of Runtime and Final Pattern Count Across Pattern Mining Approaches

Sensitivity to Minimum Support

The behavior of IIMFIs was evaluated under four minimum-support thresholds ($\text{minSup} = 0.05, 0.03, 0.01$ and 0.005) to examine how increasing or decreasing the support requirement affects the discovered pattern space. As minSup decreases, the frequency constraint becomes less restrictive, allowing increasingly less frequent combinations of learner activities to be considered. Accordingly, the number of frequent itemsets increased from 18 at $\text{minSup} = 0.05$ to 2,597 at $\text{minSup} = 0.005$. The corresponding number of maximal itemsets retained by IIMFIs increased from 15 to 342. Conversely, increasing minSup progressively restricts the pattern space. The maximal representation removes a larger proportion of frequent subsets at lower support levels, as reflected by the increase in the frequent-to-maximal reduction from 16.7% at $\text{minSup} = 0.05$ to 86.8% at $\text{minSup} = 0.005$. Therefore, the compact maximal representation is particularly relevant when lower support

thresholds produce a larger number of redundant frequent subsets.

Impact of the Novelty Threshold θ

Table 5 reports the effect of the novelty threshold θ on IIMFIs at $\text{minSup} = 0.05$. The number of maximal patterns remains constant at 15 across all evaluated θ values, indicating that novelty filtering does not alter the final retained pattern set in this configuration. Runtime decreases from 822 ms at $\theta = 0.0$ to 402 ms at $\theta = 0.3$, while peak memory usage increases substantially from 118.5 MB at $\theta = 0.0$ to 579.0 MB at $\theta = 0.7$. At lower support thresholds, novelty filtering also reduced the incremental runtime. These observations suggest that the evaluated implementation requires additional intermediate storage during similarity-based filtering at higher thresholds and therefore the observed memory variation should be interpreted as an implementation-level characteristic rather than as a direct consequence of the final pattern-set size.

Table 5: Effect of the Novelty Threshold θ on IIMFIs ($\text{minSup} = 0.05$)

θ	Maximal Patterns	Time (ms)	Memory (MB)
0.0	15	822	118.5
0.1	15	750	125.0
0.3	15	402	158.8
0.5	15	635	378.8
0.7	15	459	579.0

Note: θ = Novelty threshold, ms = milliseconds, MB = megabytes, IIMFIs = Incremental Intelligent Maximal Frequent Itemsets.

The pattern reduction ratio is calculated using Equation [9]:

$$\text{Reduction}(\%) = \frac{F-M}{F} \times 100 \quad [9]$$

Where, F is the number of frequent itemsets generated by the baseline mining algorithm and M is the number of maximal frequent itemsets retained by IIMFIs. The compression factor is defined as F/M .

Table 6 reports the pattern reduction and compression achieved by IIMFIs across the evaluated minimum-support thresholds. The reduction ratio, calculated using Equation [9], increases from 16.7% at minSup = 0.05 to 86.8% at minSup = 0.005, while the compression factor (F/M) increases from 1.2× to 7.6×. The reduction is

most pronounced at minSup = 0.005, where the number of retained patterns is reduced from 2,597 to 342. These results indicate that the compact maximal representation becomes increasingly advantageous at lower support thresholds, where the number of redundant frequent subsets grows substantially.

Table 6: Pattern Reduction and Compression Across Minimum-Support Thresholds

Minimum Support	Frequent Itemsets (FP-Growth)	Maximal Frequent Itemsets (IIMFIs)	Reduction (%)	Reduction (×)
0.05	18	15	16.7%	1.2×
0.03	46	31	32.6%	1.5×
0.01	575	160	72.2%	3.6×
0.005	2,597	342	86.8%	7.6×

Note: minSup = Minimum Support, IIMFIs = Incremental Intelligent Maximal Frequent Itemsets, FP-Growth = Frequent Pattern Growth.

Incremental Mining Capability

Table 7 reports the per-batch and cumulative runtimes for the incremental version under two novelty thresholds ($\theta = 0.0$ and $\theta = 0.5$). Table 7 provides the per-batch breakdown at minSup = 0.01. IIMFIs-Incremental completes in 659 ms

cumulatively, compared with 2,338 ms for IIMFIs-Full and 1,071 ms for FP-Growth-Full. Pattern count stabilizes at 160 after the second batch. Figure 4 shows the cumulative runtime. The cumulative runtimes of all evaluated algorithms under the selected novelty threshold $\theta = 0.5$ are reported later in the analysis.

Table 7: Cumulative Runtime and Pattern Evolution for Incremental Mining Across Four Batches (minSup = 0.01)

Batch	Cumulative Transactions	IIMFIs-Incremental per-batch (ms)	IIMFIs-Full Restart per- batch (ms)	FP-Growth-Full Restart per- batch (ms)	Patterns (IIMFIs-Incr)
1	45,149	248	452	248	158
2	90,298	201	610	260	160
3	135,447	120	620	278	160
4	180,596	90	656	285	160
Total	—	659 ($\theta = 0.5$) / 1,601 ($\theta = 0.0$)	2,338	1,071	160

Note: θ = Novelty threshold, ms = milliseconds, IIMFIs = Incremental Intelligent Maximal Frequent Itemsets, FP-Growth = Frequent Pattern Growth.

Figure 4 presents the cumulative runtime comparison over the four chronological batches. As depicted in the figure, IIMFIs-Incremental maintains a nearly constant low runtime per batch, whereas full recomputation (IIMFIs-Full and FP-

Growth-Full) shows increasing cumulative time as the dataset grows. This visualisation clearly demonstrates the scalability advantage of the incremental approach for streaming educational data.

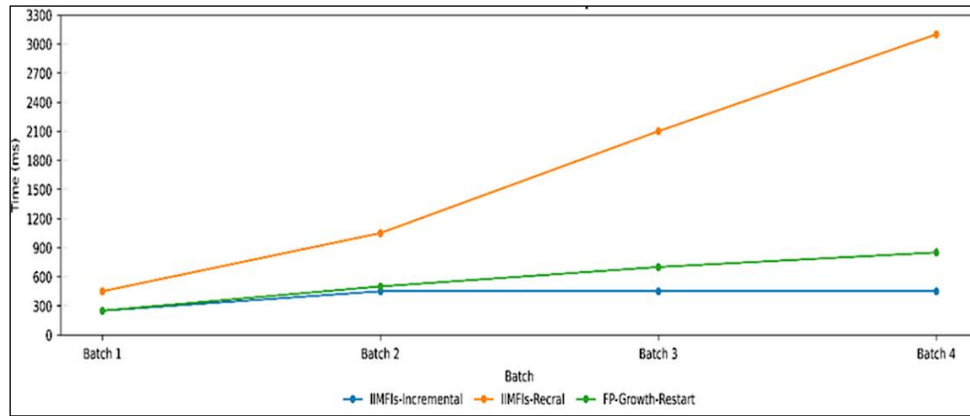


Figure 4: Cumulative Runtime Comparison

Temporal Evolution and Concept Drift

The chronological partitioning of OULAD provides a controlled simulation of evolving learner interactions and allows the incremental behavior of IIMFIs to be evaluated across successive batches. The observed changes in activity types, from 42 to 45 across the four batches, together with the emergence and disappearance of some maximal patterns, indicate that the retained pattern state evolves as new learner interactions arrive.

However, the proposed novelty measure should not be interpreted as a formal concept-drift detector. It measures structural dissimilarity between newly generated patterns and previously retained patterns and is used primarily as a computational filtering heuristic. Formal concept-drift detection generally involves dedicated monitoring and statistical or distributional change-detection mechanisms, such as Adaptive

Windowing (ADWIN) and Drift Detection Method (DDM), which are designed to identify changes in evolving data streams (29, 30).

The present study does not implement an explicit changepoint or concept-drift detection mechanism, nor does it establish that changes in the novelty scores correspond to statistically significant changes in learner behavior. Instead, the temporal experiments evaluate whether IIMFIs can accommodate newly arriving transactions and update its retained pattern representation without repeatedly recomputing the complete historical database. Therefore, the observed temporal changes should be interpreted as evidence of evolving pattern states and incremental adaptation rather than as formal evidence of concept drift.

Formal concept-drift detection, adaptive forgetting mechanisms and drift-triggered parameter adjustment are therefore identified as directions for future work.

Table 8: Recommendation Quality Comparison (minSup = 0.005, k = 5)

Algorithm	Support	θ	Precision@5	Recall@5	NDCG@5	Diversity	Novelty
User CF	—	—	0.1875	0.6353	0.5504	0.8457	0.9978
FP-Growth	0.005	—	0.1513	0.5164	0.4303	0.8747	0.9983
IIMFIs	0.005	0.0	0.0206	0.0699	0.0501	0.9676	0.9964
IIMFIs	0.005	0.7	0.0238	0.0868	0.0575	0.9599	0.9966

Note: k = number of recommendations; θ = Novelty threshold; CF = Collaborative Filtering; NDCG = Normalized Discounted Cumulative Gain; IIMFIs = Incremental Intelligent Maximal Frequent Itemsets; FP-Growth = Frequent Pattern Growth.

Recommendation Quality

Table 8 reports recommendation results at minSup = 0.005 using Precision@5, Recall@5 and NDCG@5, which are computed at k = 5. These metrics use a different cutoff from the F1@10 results reported later in this study and are therefore not directly comparable. At k = 5, FP-Growth achieved substantially higher Precision@5 [0.1513] than IIMFIs [0.0206–0.0238], corres-

ponding to an approximately 84–86% lower Precision@5. This difference is consistent with the smaller candidate pool available to the maximal pattern representation and with the higher recommendation diversity observed for IIMFIs (>0.95 vs. 0.87 for FP-Growth). Recommendation diversity was additionally evaluated to determine whether the compact maximal pattern representation resulted in overly homogeneous

recommendations. The IIMFIs configurations-maintained diversity values above 0.95 under the evaluated conditions. For example, at $\text{minSup} = 0.005$, diversity was 0.9676 at $\theta = 0.0$ and 0.9599 at $\theta = 0.7$. These results indicate that the evaluated configurations produced diverse recommendation lists, although the observed diversity should be interpreted within the specific OULAD-based evaluation protocol. Although the reported $F1@10$ values were comparable between IIMFIs and FP-Growth under the evaluated protocol, the IIMFIs configurations-maintained recommendation diversity above 0.95. The observed relationship between compactness and $\text{Precision}@5$ is correlational; a direct causal attribution to any single factor is not established by the current experimental design.

Extended Statistical Evaluation

Table 9 reports the mean cumulative runtime over four batches, with standard deviation and 95%

confidence intervals, based on 30 independent runs. At $\text{minSup} = 0.01$, IIMFIs-Incremental achieves 659 ms versus 60,964 ms for Apriori, corresponding to an approximately 92 $\times$ speedup. The confidence intervals show substantial differences in cumulative runtime between IIMFIs-Incremental and the baseline algorithms at several support thresholds; however, formal statistical significance was not evaluated for these comparisons and the intervals are presented as descriptive summaries. It is important to distinguish Table 4 from Table 9: Table 4 reports a single full run on the complete dataset at each support level, whereas Table 9 reports the cumulative time over four full restarts in the incremental simulation. Therefore, the values in the two tables are not expected to be identical and the cumulative execution time over four batches are averaged over 30 runs with confidence intervals.

Table 9: Cumulative Execution Time Over Four Batches – Incremental vs. Full Restart (Mean $\pm$ Std, 95% CI)

Algorithm	$\text{minSup} = 0.05$	$\text{minSup} = 0.03$	$\text{minSup} = 0.01$	$\text{minSup} = 0.005$
Apriori Full	1,078 $\pm$ 85 (1,048, 1,108)	2,551 $\pm$ 120 (2,508, 2,594)	60,964 $\pm$ 2,150 (60,194, 61,734)	287,275 $\pm$ 5,500 (281,308, 293,242)
FP-Growth Full	810 $\pm$ 40 (796, 824)	808 $\pm$ 38 (794, 822)	1,071 $\pm$ 50 (1,053, 1,089)	1,447 $\pm$ 70 (1,422, 1,472)
Eclat Full	1,946 $\pm$ 95 (1,912, 1,980)	1,821 $\pm$ 90 (1,789, 1,853)	2,036 $\pm$ 110 (1,997, 2,075)	2,539 $\pm$ 130 (2,493, 2,585)
IIMFIs Full ($\theta = 0.5$)	809 $\pm$ 34 (797, 821)	826 $\pm$ 37 (813, 839)	2,338 $\pm$ 102 (2,301, 2,375)	8,588 $\pm$ 212 (8,417, 8,759)
IIMFIs Incr ($\theta = 0.5$)	350 $\pm$ 19 (343, 357)	347 $\pm$ 18 (340, 354)	659 $\pm$ 27 (649, 669)	1,465 $\pm$ 57 (1,445, 1,485)

Note: CI = Confidence Interval, Std = Standard Deviation, minSup = Minimum Support, ms = milliseconds, IIMFIs = Incremental Intelligent Maximal Frequent Itemsets.

Pattern Count and Compactness

Table 10 presents the final maximal-pattern counts produced by IIMFIs-Full, IIMFIs-Incremental and the baseline algorithms across the evaluated support thresholds. At each support level, IIMFIs-Incremental produces the same final maximal-pattern count as full IIMFIs recomputation; for example, at $\text{minSup} = 0.005$,

both configurations yield 342 maximal patterns, compared with 2,597 frequent itemsets produced by Apriori, FP-Growth and Eclat. This matching of final counts provides evidence of consistency at the count level. However, equality of pattern counts does not establish set-level equivalence and direct comparison of the generated itemset sets would be required to confirm exact equivalence.

Table 10: Final Pattern Count Under the Evaluated Support Thresholds

Algorithm	$\text{minSup} = 0.05$	$\text{minSup} = 0.03$	$\text{minSup} = 0.01$	$\text{minSup} = 0.005$
Apriori / FP-Growth / Eclat	18	46	575	2,597
IIMFIs Full	15	31	160	342
IIMFIs-Incremental	15	31	160	342

Note: minSup = Minimum Support, IIMFIs = Incremental Intelligent Maximal Frequent Itemsets, FP-Growth = Frequent Pattern Growth, Eclat = Equivalence Class Transformation.

Recommendation Quality $F1@10$

Table 11 reports the mean $F1@10$ values across the evaluated support thresholds. IIMFIs (all θ variants) and the Apriori/FP-Growth baseline produce comparable $F1@10$ values at every

support level, whereas Eclat yields slightly lower values. These results indicate that, under the reported experimental protocol, the compact maximal-pattern representation maintained the reported $F1@10$ performance relative to FP-Growth.

Table 11: Mean $F1@10$ ($\pm$ Std)

Algorithm	minSup = 0.05	minSup = 0.03	minSup = 0.01	minSup = 0.005
Apriori / FP-Growth	0.7143 $\pm$ 0.0	0.3571 $\pm$ 0.0	0.0342 $\pm$ 0.001	0.0077 $\pm$ 0.0002
Eclat	0.5714 $\pm$ 0.02	0.2857 $\pm$ 0.015	0.0274 $\pm$ 0.0015	0.0061 $\pm$ 0.0003
IIMFIs (all θ variants)	0.7143 $\pm$ 0.0	0.3571 $\pm$ 0.0	0.0342 $\pm$ 0.001	0.0077 $\pm$ 0.0002

Note: $F1@10$ = Harmonic mean of Precision@10 and Recall@10, Std = Standard Deviation; minSup = Minimum Support.

Interpretation and Comparison with Prior Work

The incremental version of IIMFIs reduced cumulative runtime compared with full IIMFIs recomputation under the evaluated settings, while the maximal representation reduced pattern redundancy and maintained comparable $F1@10$ performance. The pattern representation is reduced from 2,597 to 342 at $minSup = 0.005$. Novelty threshold θ affects runtime but not final pattern count or $F1@10$ in the reported experiments. The incremental update mechanism achieves substantial speedups compared with full recomputation.

FP-Growth remained competitive in runtime at very low support levels, while IIMFIs provided a substantially more compact maximal-pattern representation. At $minSup=0.005$, the frequent – pattern representation decreased from 2,597 itemsets to 342 maximal itemsets, corresponding to an 86.8% reduction. Under the reported recommendation protocol, the $F1@10$ values were comparable between IIMFIs and FP-Growth, although IIMFIs produced lower Precision@5. Existing incremental miners such as FUP (12) and FUIFP-tree (13) lack maximal pattern extraction and novelty-aware filtering (22).

IIMFIs tracks evolving patterns, enabling adaptive recommendations without periodic full recomputation. High recommendation diversity (>0.95) is valuable for broad exposure to learning materials. This aligns with recent perspectives on adaptive e-learning systems (31). Furthermore, emerging techniques in learning style identification (32) and adaptive clustering for continual learning (33) suggest promising directions for extending the current framework to incorporate learner-specific behavioral models.

Conclusion

This study presented the Incremental Intelligent Maximal Frequent Itemsets (IIMFIs) framework for maintaining a compact maximal frequent-pattern representation in evolving e-learning transaction data. The term "Intelligent" refers to the incorporation of novelty-based heuristic filtering; it does not imply a complete AI/ML-based mining process. The framework integrates incremental state maintenance, maximal frequent-itemset extraction and novelty-based post-generation filtering, with novelty explicitly treated as a computational optimization rather than as a semantic, pedagogical, or formal concept-drift detection mechanism. Importantly, IIMFIs is presented as a heuristic incremental method; it does not claim formal completeness or exact set-level equivalence to full recomputation from the compact state alone.

Using 180,596 OULAD learner-session transactions divided chronologically into four batches, the framework reduced the pattern representation from 2,597 frequent itemsets to 342 maximal frequent itemsets at $minSup = 0.005$, corresponding to an 86.8% reduction and a 7.6-fold compression factor, as detailed in Table 6. Across the evaluated support thresholds and configurations, IIMFIs-Incremental achieved speedups ranging from $2.31\times$ to $5.86\times$ relative to full IMFIs recomputation, as shown in Table 9. At $minSup = 0.01$, IIMFIs-Incremental required 659 ms compared with 2,338 ms for full IIMFIs recomputation and 1,071 ms for FP-Growth, as reported in Tables 7 and 9. At $minSup = 0.005$, however, IIMFIs-Incremental required 1,465 ms compared with 1,447 ms for FP-Growth, as shown in Table 9, demonstrating that the incremental approach is not universally faster than FP-Growth.

For recommendation evaluation the reported $F1@10$ values were comparable between IIMFIs and FP-Growth under the evaluated protocol, as reported in Table 11, while $Precision@5$ was lower for IIMFIs, as shown in Table 8. Recommendation diversity remained above 0.95 for the evaluated IIMFIs configurations. These findings indicate that the compact maximal-pattern representation can preserve the reported $F1@10$ metric and provide high recommendation diversity under the evaluated configurations, but they do not establish superior recommendation precision.

Several limitations should be considered. First, the evaluation uses a single publicly available dataset and the chronological batches simulate an incremental environment rather than representing a continuously deployed streaming system. Second, the compact maximal-pattern state does not preserve all historical frequent subsets and may therefore omit some lower-level co-occurrence information. Third, novelty-based filtering is heuristic and no formal completeness guarantee is provided. Fourth, the current framework accommodates temporal updates but does not implement a formal concept-drift detection or adaptive forgetting mechanism. Fifth, the recommendation evaluation mines patterns from the complete dataset under the reported protocol, which may introduce optimistic bias because information from later observations can influence the mined pattern base. Finally, matching pattern counts between incremental and full recomputation do not establish exact set-level equivalence because the actual itemset sets were not directly compared.

Future research should validate IIMFIs on multiple educational datasets and real streaming environments, investigate scalable support -maintenance structures and parallel incremental implementations and perform direct set-level equivalence tests between incremental and full-recomputation outputs. Future studies should also investigate explicit concept-drift detection and forgetting mechanisms, adaptive novelty thresholds, learning style identification, adaptive clustering techniques and recommendation models beyond the current evaluation protocol. Restricting pattern mining to training data and evaluating recommendations on temporally

subsequent test data would provide a less optimistic assessment of generalization.

Abbreviations

CF: Collaborative Filtering, $F1@10$: F1 Score at Top-10 Recommendations, FP-Growth: Frequent Pattern Growth, IIMFIs: Incremental Intelligent Maximal Frequent Itemsets, minSup: Minimum Support, NDCG: Normalized Discounted Cumulative Gain, NM: Novelty Measure, OULAD: Open University Learning Analytics Dataset, PD_IIMFIs: Previously Discovered Interesting Maximal Frequent Itemsets.

Acknowledgement

The authors would like to acknowledge the support provided by Sana'a University during the conduct of this research.

Author Contributions

Emad Mahmoud Al-Azazi: Conceptualization, Methodology, Software, Validation, Formal Analysis, Investigation, Data Curation, Writing – Original Draft, Visualization, Project Administration, Ahmed Sultan Al-Hehami: Supervision, Methodology, Validation, Writing – Review and Editing.

Conflict of Interest

The authors declare that they have no conflict of interest related to this study.

Data Availability

The Open University Learning Analytics Dataset (OULAD) used in this study is publicly available through the Open University Learning Analytics repository: <https://analyse.kmi.open.ac.uk/open-dataset>. The source code for the IIMFIs framework is publicly available on GitHub:

<https://github.com/Emadmalazazi/IIMFIs-Hybrid-Recommender>.

Declaration of Artificial Intelligence (AI) Assistance

The authors declare that generative artificial intelligence (AI) and AI-assisted technologies were used only to enhance the writing style, grammar and language clarity of this manuscript. The authors prepared all figures themselves. AI-based image enhancement tools were used solely to improve the resolution, sharpness and visual quality of the figures (Figures 2, 3 and 4) after they were originally generated from the experimental

results using standard Java charting libraries. The core intellectual content, including the conception of the study, methodology, analysis, interpretation of results and conclusions, was developed entirely by the authors. No part of the scientific content, data analysis, interpretation, or Figure generation was produced by AI systems. All responsibility for the accuracy and integrity of the manuscript remains with the authors.

Ethics Approval

This study used the publicly available and anonymized Open University Learning Analytics Dataset (OULAD) for secondary analysis. No direct interaction with human participants was involved. Therefore, ethical approval and informed consent were not required for this secondary analysis, in accordance with applicable requirements for publicly available de-identified data.

Funding

This research received no external funding.

References

- Romero C, Ventura S. Educational data mining and learning analytics: an updated survey. *WIREs Data Min Knowl Discov.* 2020;10(3):e1355. doi:10.1002/widm.1355
- Murtaza M, Ahmed Y, Shamsi JA, Sherwani F, Usman M. AI-based personalized e-learning systems: issues, challenges and solutions. *IEEE Access.* 2022;10:81323-42. doi:10.1109/ACCESS.2022.3193938
- Fournier-Viger P, Lin JCW, Vo B, Chi TT, Zhang J, Le HB. A survey of itemset mining. *WIREs Data Mining and Knowledge Discovery.* 2017;7(4):e1207. doi:10.1002/widm.1207
- Han J, Pei J, Yin Y, Mao R. Mining frequent patterns without candidate generation: a frequent-pattern tree approach. *Data Min Knowl Discov.* 2004;8(1):53-87. doi:10.1023/B:DAMI.0000005258.31418.83
- Zaki MJ. Scalable algorithms for association mining. *IEEE Trans Knowl Data Eng.* 2000;12(3):372-90. doi:10.1109/69.846291
- Burdick D, Calimlim M, Flannick J, Gehrke J, Yiu T. MAFIA: a maximal frequent itemset algorithm. *IEEE Trans Knowl Data Eng.* 2005;17(11):1490-504. doi:10.1109/TKDE.2005.183
- Grahne G, Zhu J. Fast algorithms for frequent itemset mining using FP-trees. *IEEE Trans Knowl Data Eng.* 2005;17(10):1347-62. doi:10.1109/TKDE.2005.166
- Aryabarzan N, Minaei-Bidgoli B, Teshnehlab M. negFIN: an efficient algorithm for fast mining frequent itemsets. *Expert Syst Appl.* 2018;105:129-43. doi:10.1016/j.eswa.2018.03.041
- Deng ZH. DiffNodesets: an efficient structure for fast mining frequent itemsets. *Appl Soft Comput.* 2016;41:214-23. <https://doi.org/10.1016/j.asoc.2016.01.010>
- Fatemi SM, Hosseini SM, Kamandi A, Shabankhah M. CL-MAX: a clustering-based approximation algorithm for mining maximal frequent itemsets. *Int J Mach Learn Cybern.* 2021;12(2):365-83. doi:10.1007/s13042-020-01177-5
- Mi X. The mining algorithm of maximum frequent itemsets based on frequent pattern tree. *Comput Intell Neurosci.* 2022;7022168. doi:10.1155/2022/7022168
- Nath B, Bhattacharyya DK, Ghosh A. Incremental association rule mining: a survey. *WIREs Data Min Knowl Discov.* 2013;3(3):157-69. doi:10.1002/widm.1086
- Hong TP, Lin CW, Wu YL. Incrementally fast updated frequent pattern trees. *Expert Syst Appl.* 2008;34(4):2424-35. doi:10.1016/j.eswa.2007.04.009
- Hong TP, Wang CY, Tao YH. A new incremental data mining algorithm using pre-large itemsets. *Intell Data Anal.* 2001;5(2):111-29. doi:10.3233/IDA-2001-5203
- Lin CW, Hong TP, Lu WH. The Pre-FUFP algorithm for incremental mining. *Expert Syst Appl.* 2009;36(5):9498-505. doi:10.1016/j.eswa.2008.03.014
- Sun J, Xun Y, Zhang J, Li J. Incremental frequent itemsets mining with FCFP tree. *IEEE Access.* 2019;7:136511-24. doi:10.1109/ACCESS.2019.2943015
- Thurachon W, Kreesuradej W. Incremental association rule mining with a fast incremental updating frequent pattern growth algorithm. *IEEE Access.* 2021;9:55726-41. doi:10.1109/ACCESS.2021.3071777
- Essam A, Abdel-Fattah MA, Abdelhamid LM. Towards enhancing the performance of parallel FP-Growth on Spark. *IEEE Access.* 2022;10:286-96. doi:10.1109/ACCESS.2021.3137789
- Yun U, Lee G. Incremental mining of weighted maximal frequent itemsets from dynamic databases. *Expert Syst Appl.* 2016;54(C):304-27. doi:10.1016/j.eswa.2016.01.049
- Fasihy H, Shahraki MHN. Incremental mining maximal frequent patterns from univariate uncertain data. *Knowl Based Syst.* 2018;152:40-50. doi:10.1016/j.knosys.2018.04.001
- Song Y, Yang L, Wang Y, Xiao X, You S, Tang Z. Parallel incremental association rule mining framework for public opinion analysis. *Inf Sci.* 2023;630:523-45. doi:10.1016/j.ins.2023.02.034
- Wu T, Chen Y, Han J. Re-examination of interestingness measures in pattern mining: a unified framework. *Data Min Knowl Discov.* 2010;21(3):371-97. doi:10.1007/s10618-009-0161-2
- Geng L, Hamilton HJ. Interestingness measures for data mining: a survey. *ACM Comput Surv.* 2006;38(3):9. doi:10.1145/1132960.1132963
- Padmanabhan B, Tuzhilin A. Unexpectedness as a measure of interestingness in knowledge discovery. *Decis Support Syst.* 1999;27(3):303-18. doi:10.1016/S0167-9236(99)00053-6

25. Nabizadeh AH, Gonçalves D, Gama S, Jorge J, Rafsanjani HN. Adaptive learning path recommender approach using auxiliary learning objects. *Comput Educ.* 2020;147:103777. doi:10.1016/j.compedu.2019.103777
26. Fkih F. Similarity measures for collaborative filtering-based recommender systems: review and experimental comparison. *J King Saud Univ Comput Inf Sci.* 2022;34(9):7645-69. doi:10.1016/j.jksuci.2021.09.014
27. Xun Y, Cui X, Zhang J, Yin Q. Incremental frequent itemsets mining based on frequent pattern tree and multi-scale. *Expert Syst Appl.* 2021;163:113805. doi:10.1016/j.eswa.2020.113805
28. Shaikh M, Akram S, Khan J, Khalid S, Lee Y. DIAFM: an improved and novel approach for incremental frequent itemset mining. *Mathematics.* 2024;12(24):3930. doi:10.3390/math12243930
29. Hinder F, Vaquet V, Hammer B. One or two things we know about concept drift—a survey on monitoring in evolving environments. Part A: detecting concept drift. *Front Artif Intell.* 2024;7:1330257. doi:10.3389/frai.2024.1330257
30. Barros RSM, Cabral DRL, Gonçalves PM Jr, Santos SGTC. RDDM: reactive drift detection method. *Expert Syst Appl.* 2017;90(C):344-55. doi:10.1016/j.eswa.2017.08.023
31. Romi IM. An adaptive e-learning systems success model. *Int J Emerg Technol Learn.* 2023;18(18):177-91. doi:10.3991/ijet.v18i18.42929
32. Ayyoub HY, Al-Kadi OS. Learning style identification using semi-supervised self-taught labeling. *IEEE Trans Learn Technol.* 2024;17:1093-106. doi:10.1109/TLT.2024.3358864
33. Masuyama N, Amako N, Yamada Y, Nojima Y, Ishibuchi H. Adaptive resonance theory-based topological clustering with a divisive hierarchical structure capable of continual learning. *IEEE Access.* 2022;10:68042-56. doi:10.1109/ACCESS.2022.3186479

How to Cite: Al-Azazi EM, Al-Hegami AS. Incremental Maximal Frequent Itemset Framework for Adaptive E-learning Recommendation Systems. *Int Res J Multidiscip Scope.* 2026;7(4):1181-1197.

DOI: 10.47857/irjms.2026.v07i04.012921