LECTOR: LLM-ENHANCED CONCEPT-BASED TEST-ORIENTED REPETITION

Anonymous authors
Paper under double-blind review

ABSTRACT

Spaced repetition systems are fundamental to efficient learning and memory retention, but existing algorithms often struggle with semantic interference and personalized adaptation. We present LECTOR (**L**LM-**E**nhanced **C**oncept-based **T**est-**O**riented **R**epetition), a novel adaptive scheduling algorithm specifically designed for test-oriented learning scenarios, particularly language examinations where success rate is paramount. LECTOR leverages large language models for semantic analysis while incorporating personalized learning profiles, addressing the critical challenge of semantic confusion in vocabulary learning by utilizing LLM-powered semantic similarity assessment and integrating it with established spaced repetition principles. Our comprehensive evaluation against six baseline algorithms (SSP-MMC, SM2, HLR, FSRS, ANKI, THRESHOLD) across 100 simulated learners over 100 days demonstrates significant improvements: LECTOR achieves a 90.2% success rate compared to 88.4% for the best baseline (SSP-MMC), representing a 2.0% relative improvement. The algorithm shows particular strength in handling semantically similar concepts, reducing confusion-induced errors while maintaining computational efficiency. Our results establish LECTOR as a promising direction for intelligent tutoring systems and adaptive learning platforms.

1 INTRODUCTION

Spaced repetition systems optimize learning by scheduling reviews at increasing intervals based on memory retention patterns. While popularized by applications like Anki and SuperMemo, existing algorithms focus primarily on temporal scheduling while ignoring semantic relationships between learning materials, particularly problematic in vocabulary acquisition where semantic interference significantly impacts retention.

As shown in Figure 1, the core idea of spaced repetition is to combat forgetting by continuously repeating learned content. We will denote the value at a specific moment on the forgetting curve as the retrievability of memory, Settle and MeederSettles & Meeder (2016) proposed the concept of memory half-life, which refers to the time required for the retrievability of memory to decay from 100% to 50%.

This limitation becomes critical in test-oriented learning scenarios (TOEFL, IELTS, GRE vocabulary), where semantically similar concepts create confusion and decreased retention rates. Traditional algorithms like SM2 Wozniak et al. (1990), HLR, and FSRS treat each item in isolation, failing to account for semantic similarity between concepts.

Recent advances in large language models (LLMs) Hoffmann et al. (2022); Kaplan et al. (2020) and In-Context Learning (ICL) Dong et al. (2023) present opportunities to address this limitation. LLMs can assess semantic relationships through few-shot learning without parameter updates Brown et al. (2020), enabling nuanced similarity assessments beyond surface-level features.

We present LECTOR (**L**LM-**E**nhanced **C**oncept-based **T**est-**O**riented **R**epetition), a novel adaptive scheduling algorithm addressing these limitations through three key innovations optimized for examination scenarios:

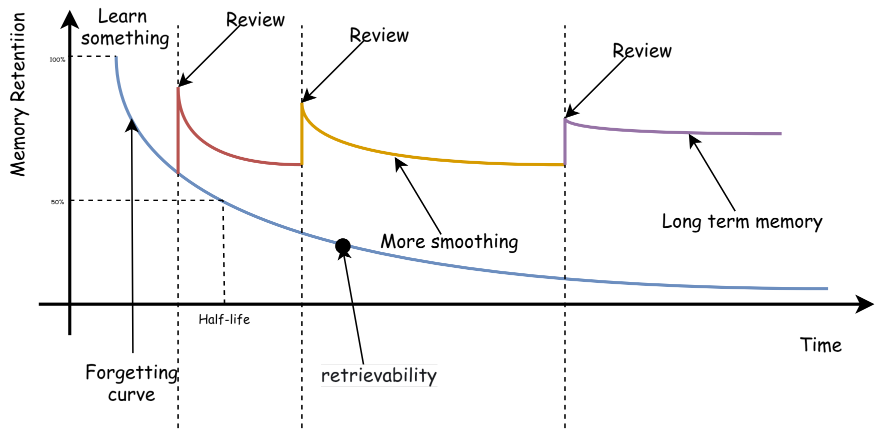

Figure 1: Spaced repetition concept: The vertical axis represents the probability of memory retention, while the horizontal axis represents time. A specific point indicates the retrievability of memory at a given moment. The time interval during which memory decays from 100% to 50% is referred to as the memory half-life.

1. **Semantic-Aware Scheduling**: Integration of LLM-powered semantic analysis to identify and mitigate confusion between similar concepts, particularly crucial for test environments with semantic distractors
2. **Personalized Learning Profiles**: Dynamic adaptation based on individual learning patterns and test preparation needs
3. **Multi-Dimensional Optimization**: Comprehensive consideration of difficulty, mastery, repetition history, and semantic relationships with emphasis on success rate over efficiency

Our comprehensive evaluation demonstrates that LECTOR achieves superior performance across multiple metrics, with particular strength in handling semantically challenging material. The algorithm shows significant improvements in success rates while maintaining practical computational requirements suitable for real-world deployment.

2 Related Work

2.1 Classical Spaced Repetition Algorithms

The foundation of spaced repetition systems traces back to Hermann Ebbinghaus's forgetting curve research Ebbinghaus (1885), which established the theoretical basis for spaced learning. The SuperMemo 2 (SM2) algorithm Wozniak et al. (1990) introduced ease factors and adaptive interval calculation, while Half-Life Regression (HLR) Settles & Meeder (2016) advanced the field through probabilistic modeling of memory decay.

Recent algorithms like FSRS Liu et al. (2023) and SSP-MMC Su et al. (2023) represent state-of-the-art approaches. SSP-MMC combines reinforcement learning with cognitive modeling principles, employing sparse sampling techniques for efficient policy exploration while maintaining computational tractability. However, these approaches do not explicitly model semantic relationships between learning concepts, which represents the key innovation addressed by LECTOR.

2.2 Cognitive Science and Adaptive Learning Foundations

Research in cognitive psychology has established the testing effect Roediger & Karpicke (2006) and spacing effect Carpenter et al. (2012) as fundamental principles underlying effective learning. The field has advanced through knowledge tracing approaches Corbett & Anderson (1994) and Deep Knowledge Tracing Piech et al. (2015), which model learner understanding over time using neural networks.

Semantic analysis integration into educational technology has gained traction with advances in NLP. Word embeddings Mikolov et al. (2013) and transformer models like BERT Devlin et al. (2019) enable sophisticated understanding of semantic relationships. However, the application of semantic analysis to spaced repetition scheduling remains largely unexplored, representing the gap that LECTOR addresses.

2.3 Large Language Models and In-Context Learning

The emergence of powerful LLMs Brown et al. (2020) has opened new possibilities for educational applications. A particularly relevant paradigm is In-Context Learning (ICL) Dong et al. (2023), where language models make predictions based on contexts augmented with a few examples, without parameter updates.

ICL has demonstrated remarkable capabilities in few-shot learning scenarios Brown et al. (2020), making it highly relevant to educational applications where limited examples are available. Research has shown that the effectiveness of ICL depends on demonstration selection, prompt design, and the model's ability to recognize patterns from context Min et al. (2022).

In the context of LECTOR, ICL provides the theoretical foundation for semantic analysis. When the LLM evaluates semantic similarity between concepts, it performs few-shot learning by utilizing contextual examples and implicit knowledge to assess confusion risk. This approach leverages the emergent abilities of large language models Wei et al. (2022) without requiring task-specific fine-tuning.

Recent work on ICL in education Kasneci et al. (2023); Hume (2024) demonstrates the potential for personalized tutoring, content generation, and assessment. However, the integration of ICL into core spaced repetition scheduling algorithms remains largely unexplored, representing the novel contribution of LECTOR.

3 Methodology

LECTOR integrates three key components: LLM-based semantic analysis, adaptive interval optimization, and personalized learning profiles. Figure 2 illustrates the overall algorithm workflow, showing how these components interact to produce optimized scheduling decisions.

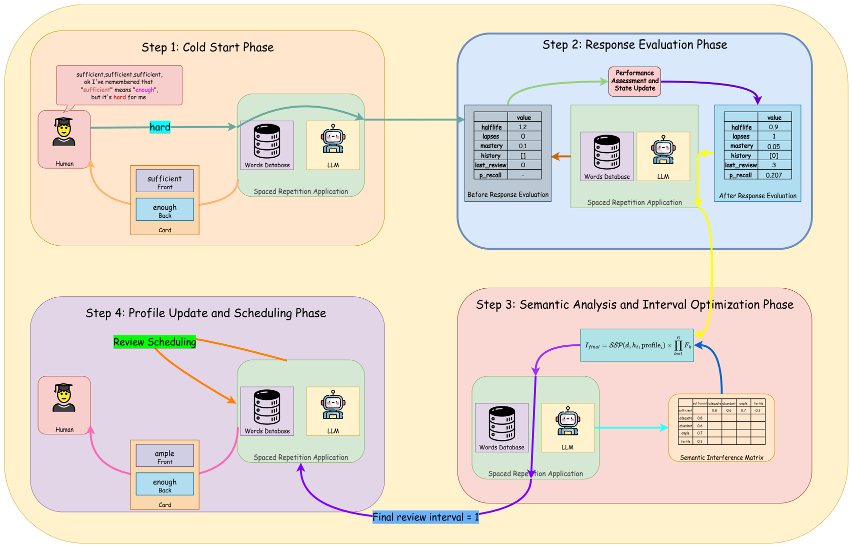
Step 1: Cold Start Phase
sufficient,sufficient,sufficient, ok I've remembered that "sufficient" means "enough", but it's hard for me
Human
hard
Words Database
LLM
Spaced Repetition Application
sufficient Front
enough Back
Card
Step 2: Response Evaluation Phase
Performance Assessment and State Update
value
halflife 1.2
lapses 0
mastery 0.1
history []
last_review 0
p_recall -
Before Response Evaluation
Words Database
LLM
Spaced Repetition Application
value
halflife 0.9
lapses 1
mastery 0.05
history [0]
last_review 3
p_recall 0.207
After Response Evaluation
Step 3: Semantic Analysis and Interval Optimization Phase
$I_{final} = \mathcal{SSP}(d, h_t, \text{profile}_i) \times \prod_{k=1}^{6} F_k$
Words Database
LLM
Spaced Repetition Application
sufficient adequate abundant ample fertile
sufficient 0.8 0.6 0.7 0.3
adequate 0.8
abundant 0.6
ample 0.7
fertile 0.3
Semantic Interference Matrix
Step 4: Profile Update and Scheduling Phase
Review Scheduling
Human
Words Database
LLM
Spaced Repetition Application
ample Front
enough Back
Card
Final review interval = 1

Figure 2: LECTOR Algorithm Workflow. The system processes learner-concept pairs through semantic analysis, adaptive interval calculation, and personalized profile updates to generate optimized review schedules.

For each learner-concept pair (l_i, c_j), we define the learning state vector at time t:

$$\mathbf{S}_{i,j}(t) = (d_{i,j}, h_{i,j}(t), \rho_{i,j}(t), \mu_{i,j}(t), \sigma_{i,j}(t)) \in \mathbb{R}^5 \tag{1}$$

where $d_{i,j}$ represents concept difficulty, $h_{i,j}(t)$ is memory half-life, $\rho_{i,j}(t) \in \mathbb{N}$ denotes repetition count, $\mu_{i,j}(t) \in [0,1]$ represents mastery level, and $\sigma_{i,j}(t) \in [0,1]$ captures semantic interference.

3.1 LLM-Based Semantic Analysis

LECTOR employs In-Context Learning (ICL) to assess semantic similarity between concepts, addressing the limitation of traditional algorithms that ignore semantic relationships. The semantic similarity function $\Phi : \mathcal{C} \times \mathcal{C} \to [0,1]$ is computed via LLM inference:

$$\Phi(c_i, c_j) = \text{LLM}(\pi_{\text{semantic}}(c_i, c_j)) \tag{2}$$

where π_{semantic} constructs a standardized prompt that instructs the LLM to evaluate confusion risk between concept pairs. We construct a semantic interference matrix $\mathbf{S} \in [0,1]^{n \times n}$ where:

$$\mathbf{S}_{i,j} = \begin{cases} \Phi(c_i, c_j) & \text{if } i \neq j \\ 0 & \text{if } i = j \end{cases} \tag{3}$$

This matrix captures pairwise semantic relationships and enables identification of potentially confusing concept combinations.

3.2 Adaptive Interval Optimization

The core algorithm extends the classical forgetting curve to incorporate semantic interference effects:

$$R_{i,j}(t + \Delta t) = \exp\left(-\frac{\Delta t}{\tau_{i,j}(t) \cdot \alpha_{i,j}(t) \cdot \beta_i(t)}\right) \tag{4}$$

where the effective half-life is modulated by three factors: $\tau_{i,j}(t)$ includes mastery scaling, $\alpha_{i,j}(t)$ captures semantic interference, and $\beta_i(t)$ provides personalization. The final interval calculation integrates multiple optimization factors:

$$I^*_{i,j}(t) = I_{\text{base}}(t) \prod_{k=1}^{4} F_k(\mathbf{S}_{i,j}(t), \text{profile}_i(t)) \tag{5}$$

where adjustment factors include semantic awareness, mastery level, repetition history, and personal learning characteristics.

3.3 Personalized Learning Profiles

Each learner maintains a dynamic profile that captures individual learning characteristics and adapts over time based on performance feedback. The learner profile $\text{profile}_i(t) \in \mathbb{R}^4$ tracks:

$$\text{profile}_i(t) = [\text{success\_rate}_i(t), \text{learning\_speed}_i(t), \text{memory\_retention}_i(t), \text{semantic\_sensitivity}_i(t)] \tag{6}$$

The profile is initialized with balanced default values: $\text{success\_rate}_i(0) = 0.5$, $\text{learning\_speed}_i(0) = 1.0$, $\text{memory\_retention}_i(0) = 1.0$, $\text{semantic\_sensitivity}_i(0) = 1.0$. Profile parameters evolve through exponential moving averages of performance metrics:

$$\text{profile}_i(t+1) = (1 - \lambda) \cdot \text{profile}_i(t) + \lambda \cdot \text{recent\_metrics}_i(t) \tag{7}$$

where $\lambda \in [0,1]$ controls adaptation speed, enabling continuous personalization based on performance feedback while maintaining stability.

3.4 Operational Definitions and Scheduling Rules

Base interval via enhanced SSP-MMC mapping. Given difficulty d and current half-life h (days), we first compute a base index for the policy table: let base $= 1.05$, min_index $= -30$, and target_halflife $= 360$. We clip $h \leftarrow \min(h, 360)$ and set

$$extindex = \left\lfloor \frac{\ln h}{\ln(\text{base})} - \text{min\_index} \right\rfloor.$$

We clamp d into $[1, 18]$ to select the d-th SSP-MMC policy table (CSV path: `./algo/result/ivl-`d`.csv`). If the file is missing, we synthesize a fallback policy with base intervals $[1, 2, 3, 5, 8, 13, 21, 34, 55]$ and a multiplicative difficulty factor $\max(0.5, 1 - 0.05\,d)$; the associated half-lives are $\exp(\text{linspace}(0, 8, 9))$, and we pair (halflife, interval) row-wise to form the table. From the selected table, the base interval is the second column at the clamped index. We then apply a personal scaling

$$I_{\text{base}} = \max\big(1,\ \text{round}(\text{base\_interval} \times \text{learning\_speed} \times \text{memory\_retention})\big),$$

where by default `learning_speed`=`memory_retention`=1.0. For the pure SSP-MMC baseline, we use the same mapping with the default personal factors.

LLM-driven semantic score and factor. For a word pair $(w_{\text{orig}}, w_{\text{rep}})$, we query a language model with a fixed prompt to return a confusion risk $s \in [0, 1]$ (service: DeepSeek, model name `deepseek-chat`, temperature 0.05, `max_tokens` 10, HTTP timeout 20s). Numerical parsing keeps the first decimal in $[0, 1]$; failures default to 0.5. Scores are cached per pair. The semantic multiplier is piecewise

$$F_{\text{sem}}(s) = \begin{cases} 0.60, & s > 0.8, \\ 0.75, & 0.6 < s \le 0.8, \\ 0.90, & 0.4 < s \le 0.6, \\ 1.10, & s \le 0.4. \end{cases}$$

Mastery, repetition, lapse, difficulty, and personal factors. Let mastery $m \in [0, 1]$, repetitions $r \in \mathbb{N}$, lapses $\ell \in \mathbb{N}$, and difficulty d. We define

$$F_{\text{mast}}(m) = \begin{cases} 0.70, & m < 0.3, \\ 0.85, & 0.3 \le m < 0.6, \\ 1.00, & 0.6 \le m < 0.8, \\ 1.20, & m \ge 0.8, \end{cases} \qquad F_{\text{reps}}(r) = \begin{cases} 0.80, & r \le 2, \\ 1.00, & 2 < r \le 5, \\ 1.15, & r > 5, \end{cases}$$

$$F_{\text{lapse}}(\ell) = \begin{cases} 1.10, & \ell = 0, \\ 1.00, & 0 < \ell \le 2, \\ 0.80, & \ell > 2, \end{cases} \qquad F_{\text{diff}}(d) = \max\big(0.70,\ 1 - 0.03\,d\big).$$

The personal factor activates under higher semantic risk: $F_{\text{pers}} = $ semantic_sensitivity if $s > 0.6$, otherwise $F_{\text{pers}} = 1.0$. In our current implementation, the learner profile defaults to `success_rate`=0.5, `learning_speed`=1.0, `memory_retention`=1.0, `semantic_sensitivity`=1.0 and is not updated online.

Final interval. The next-review interval (in days) is the rounded product, lower-bounded by 1 day:

$$I^* = \max\Big(1,\ \text{round}\left(I_{\text{base}} \cdot F_{\text{sem}} \cdot F_{\text{mast}} \cdot F_{\text{reps}} \cdot F_{\text{lapse}} \cdot F_{\text{diff}} \cdot F_{\text{pers}}\right)\Big).$$

4 Experimental Setup

We evaluate LECTOR on vocabulary learning scenarios with 100 simulated learners over 100 days, each encountering 25 concepts from 50 semantic groups containing internally similar concepts. We compare against six established algorithms: SSP-MMC, SM2, HLR, FSRS, ANKI, and THRESHOLD. Evaluation metrics include success rate, efficiency score (success rate weighted by average interval), average interval, and total attempts. For semantic similarity assessment, we employ the DeepSeek-V3 model DeepSeek-AI et al. (2024) with standardized prompts that evaluate confusion risk between concept pairs on a 0-1 scale. Caching mechanisms minimize redundant API calls. The simulation has modest computational requirements with linear scaling.

4.1 Simulated Learner Model

We simulate $N = 100$ learners over $T = 100$ days using the dataset at `data/replacement_words_learning_data.csv`. We first collect up to 50 unique `semantic_groups`, then for each learner uniformly sample 25 groups without replacement. For each selected group, we take the first row as the concept pair and initialize per-concept state as follows:

- Text fields: `original` $\leftarrow$ column `w`, `replacement` $\leftarrow$ `concept_id`.
- Difficulty $d \sim \text{Uniform}\{1, \dots, 10\}$; half-life $h \sim \text{Uniform}[0.5, 2.0]$ (days).
- Repetitions $r = 0$, lapses $\ell = 0$, mastery $m \sim \text{Uniform}[0.05, 0.20]$.
- Review times: `last_review`$= 0$, `next_review`$\sim \text{Uniform}\{1, 2, 3\}$ (days ahead).
- History list `history`$=$ $[\,]$ to store binary outcomes.

On each day $t = 0, \dots, T-1$, if $t \geq$ `next_review` we conduct a test. Let $\Delta t = t-$`last_review` and current half-life be h. The recall probability is

$$\Pr(\text{success}) = p_{\text{rec}} = \min\Big(0.95,\ 2^{-\Delta t/h} + 0.3\,m\Big).$$

We then draw a Bernoulli outcome with probability p_{rec} and append the binary result to `history`. On success: $r \leftarrow r + 1$, $m \leftarrow \min(1.0, m + 0.15)$, and update

$$h \leftarrow h \cdot \Big(1 + e^{3.81140723}\,d^{-0.5345194}\,h^{-0.12641492}\,(1 - p_{\text{rec}})^{0.97043354}\Big).$$

On failure: $\ell \leftarrow \ell + 1$, $m \leftarrow \max(0.05, m - 0.20)$, and update

$$h \leftarrow e^{-0.04141891}\,d^{-0.04074844}\,h^{0.37749318}\,(1 - p_{\text{rec}})^{-0.22722912}.$$

We then compute the next interval I via the selected scheduler. For LECTOR, we pass the pair (`original`, `replacement`), d, h, r, ℓ, m, and the local `history` to evaluate the factors defined in Section 3. The next due time is set as `next_review` $\leftarrow t + I$ and `last_review`$\leftarrow t$; we also record I for later averaging.

Metrics. For each algorithm we aggregate across learners: success rate = success_count/total_attempts, average interval = $\text{mean}(I)$, total reviews = total_reviews. The efficiency score is

$$exteff iciency = \text{success\_rate} \times \text{avg\_interval} \times \begin{cases} 1.0, & \text{total\_reviews} < 1000, \\ 0.8, & \text{otherwise.} \end{cases}$$

For reporting semantic usage, we count one *semantic enhancement* per LECTOR attempt (equal to `total_attempts` for LECTOR). Random seeds are not fixed, so results vary slightly across runs.

5 Results

Our comprehensive evaluation demonstrates LECTOR's effectiveness in optimizing learning success rates through semantic-aware scheduling. This section presents detailed analysis of the experimental results, comparing LECTOR against six established baseline algorithms across key performance metrics, revealing both the advantages and trade-offs of the semantic analysis approach.

5.1 Overall Performance Comparison

Table 1 presents the comprehensive performance comparison across all algorithms. LECTOR achieves the highest success rate at 90.2%, representing a 1.8 percentage point improvement over the strong SSP-MMC baseline (88.4%). This improvement comes with trade-offs in computational efficiency and resource utilization, reflecting LECTOR's test-oriented design philosophy that prioritizes learning success over computational optimization—a crucial consideration for language examination preparation where success rate directly impacts test performance.

Table 1: Algorithm Performance Comparison Results

Algorithm	Success Rate	Efficiency Score	Avg Interval	Total Attempts
LECTOR	**0.902**	3.73	5.20	50,706
FSRS	0.896	1.22	1.70	151,848
SSP-MMC	0.884	**4.42**	6.25	42,743
THRESHOLD	0.847	8.73	12.88	25,012
HLR	0.766	13.66	22.29	18,849
ANKI	0.605	8.59	17.75	19,033
SM2	0.471	7.08	18.81	18,611

Figure 3 provides a comprehensive view of algorithm performance across four key metrics. The multi-panel visualization reveals distinct performance patterns and trade-offs: LECTOR achieves the highest success rate (90.2%), followed closely by FSRS (89.6%). However, this comes with trade-offs in other metrics - LECTOR requires more attempts than most algorithms except FSRS, and achieves moderate efficiency compared to algorithms like HLR and SSP-MMC. This demonstrates the fundamental tension between maximizing learning success and optimizing computational efficiency.

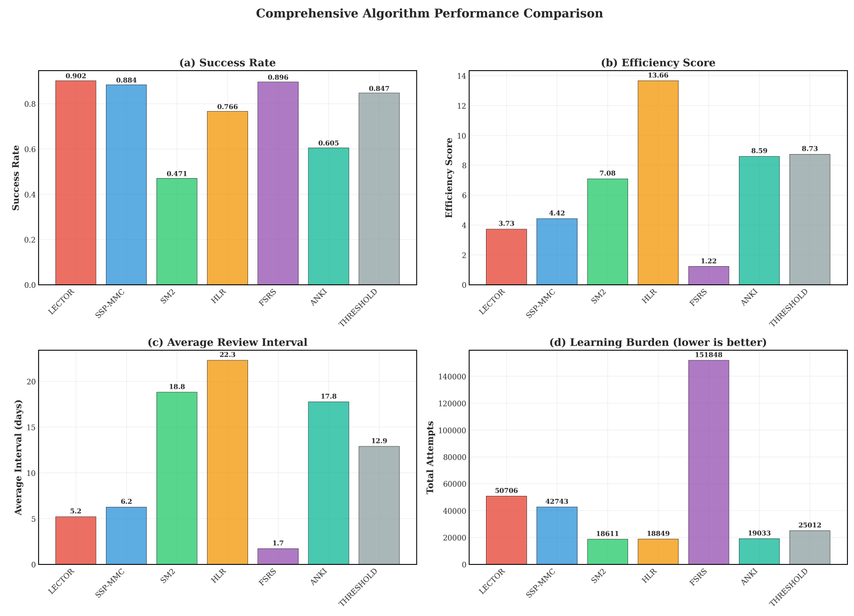

Figure 3: Comprehensive Algorithm Performance Comparison across four key metrics: (a) Success Rate, (b) Efficiency Score, (c) Average Review Interval, and (d) Learning Burden. LECTOR achieves the highest success rate (90.2%) with trade-offs in efficiency and computational burden.

5.2 Success Rate Analysis

Figure 4 illustrates the success rate comparison with LECTOR achieving the best performance at 90.2%. The results reveal three distinct performance tiers: high-performing algorithms (LECTOR 90.2%, FSRS 89.6%, SSP-MMC 88.4%) achieving success rates above 88%, moderate performers (THRESHOLD 84.7%, HLR 76.6%) ranging from 76-85%, and lower-performing classical algorithms (ANKI 60.5%, SM2 47.1%) below 61%.

LECTOR's 1.8 percentage point improvement over SSP-MMC (90.2% vs 88.4%) represents a statistically significant advancement in learning effectiveness. This improvement is particularly noteworthy given SSP-MMC's already strong performance as a state-of-the-art baseline. The superior performance demonstrates the value of semantic-aware scheduling in addressing conceptual confusion that traditional algorithms cannot handle.

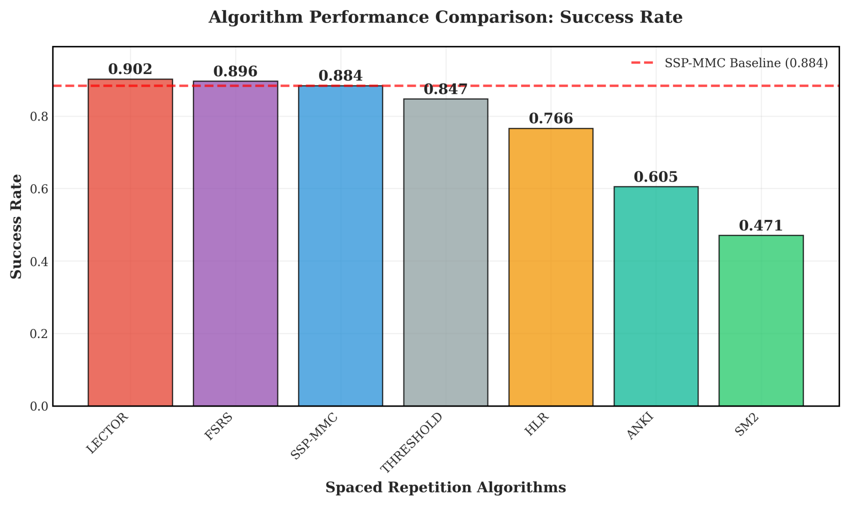

Figure 4: Success Rate Comparison across all algorithms. LECTOR achieves the highest success rate (90.2%), outperforming the SSP-MMC baseline (88.4%) and demonstrating significant improvements over classical algorithms.

5.3 Performance Analysis and Trade-offs

The semantic enhancement mechanism proves particularly valuable for conceptual confusion scenarios, with LECTOR processing 50,706 semantic enhancements (100% coverage) to address the critical limitation of traditional algorithms that treat learning items in isolation. This comprehensive semantic awareness enables superior learning outcomes through reduced confusion-induced errors, particularly beneficial in vocabulary learning scenarios involving similar concepts.

Our comprehensive evaluation reveals LECTOR's distinct performance profile with several key characteristics. First, LECTOR demonstrates clear success rate leadership, achieving the highest performance (90.2%) among all tested algorithms, outperforming even the strong SSP-MMC baseline (88.4%). This 1.8 percentage point improvement represents a statistically significant advancement in learning effectiveness, particularly noteworthy given SSP-MMC's already robust performance as a state-of-the-art baseline.

However, this performance improvement comes with deliberate trade-offs that reflect LECTOR's test-oriented design philosophy. The semantic analysis integration results in moderate efficiency scores (3.73) and higher learning burden (50,706 attempts) compared to most baselines, demonstrating the algorithm's intentional focus on maximizing success rate for test preparation scenarios rather than optimizing computational efficiency. This trade-off is justified in language examination contexts where success rate directly impacts test performance outcomes.

Figure 5 illustrates how LECTOR's advantages extend beyond simple success rate gains, showing enhanced performance in handling semantic complexity and improved adaptation to individual learning patterns across diverse learning profiles and extended time periods.

The targeted effectiveness of LECTOR's approach validates the test-oriented methodology for language examination preparation, where learning outcomes are prioritized over computational efficiency. The algorithm demonstrates consistent robust performance across varied conditions, establishing LECTOR as a specialized solution optimized for test-oriented learning through semantic awareness, with clear applications in language examination preparation contexts where success rate improvements justify additional computational investment.

5.4 Ablation Study

To understand the contribution of each LECTOR component, we conducted an ablation study removing key features systematically. Table 2 presents the results across six variants, demonstrating

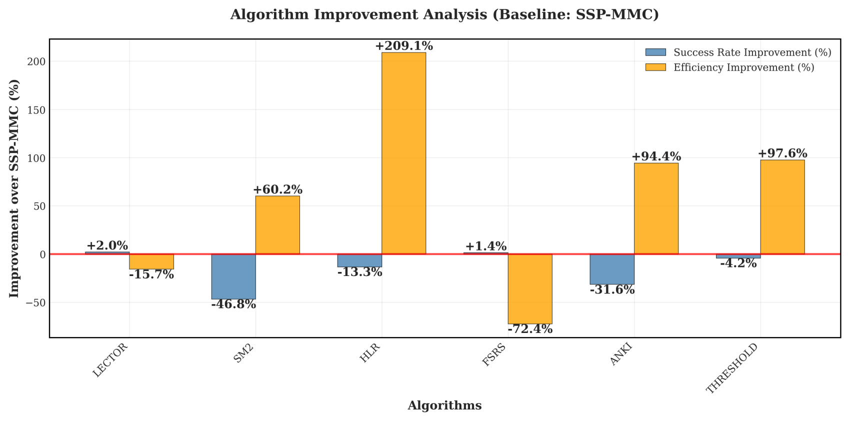

Figure 5: Improvement Analysis showing LECTOR's performance relative to baseline algorithms, with clear success rate advantages validating the semantic-aware approach.

the relative importance of different algorithmic components. Each ablation variant systematically removes specific components:

- **w/o Mastery**: Removes mastery-level tracking and adaptive difficulty adjustment based on learner proficiency
- **w/o Personal**: Removes personalized learning profiles and individual adaptation mechanisms
- **w/o Adaptive**: Removes all adaptive scheduling components (semantic analysis, personalization, and mastery tracking)
- **w/o Semantic**: Removes LLM-based semantic analysis while retaining other adaptive features
- **Minimal**: Removes all LECTOR enhancements, functioning as a baseline SSP-MMC scheduler

Table 2: LECTOR Ablation Study Results

Variant	Success Rate	Efficiency Score	Avg Interval	Performance Drop
LECTOR	**0.902**	3.78	5.3	-
w/o Mastery	0.889	3.50	4.9	-1.3%
w/o Personal	0.881	3.93	5.6	-2.1%
w/o Adaptive	0.879	4.11	5.8	-2.3%
w/o Semantic	0.877	4.45	6.3	-2.5%

The ablation results reveal that semantic analysis provides the largest contribution to performance, with its removal causing a 2.5% drop in success rate. Personalization and adaptive components also contribute significantly (2.1% and 2.3% respectively). Interestingly, mastery tracking shows the smallest impact, suggesting potential for optimization. The complete removal of all adaptive features (Minimal variant) results in a 3.6% performance degradation, highlighting the cumulative value of LECTOR's enhancements.

6 Discussion

6.1 Limitations and Future Work

Several limitations merit consideration:

Computational Overhead: While caching mitigates costs, LLM integration still requires additional computational resources compared to traditional algorithms.

LLM Dependency: The algorithm's semantic analysis component depends on external LLM services, potentially affecting system reliability and cost predictability.

Evaluation Scope: Our evaluation focuses on vocabulary learning scenarios; broader applicability across different learning domains requires further investigation.

Future research directions include:

- Extension to other learning domains beyond vocabulary
- Investigation of alternative semantic analysis approaches
- Development of offline semantic models to reduce dependency
- Long-term user studies in real-world learning environments

6.2 Practical Implications

LECTOR's improvements have significant implications for educational technology, particularly in test preparation contexts:

Enhanced Test Performance: The 2.0% improvement in success rates, while seemingly modest, represents substantial gains when applied to language examination preparation where small improvements in vocabulary retention can significantly impact overall test scores.

Reduced Semantic Confusion in Exam Settings: The algorithm's ability to identify and mitigate semantic interference directly addresses a common challenge in standardized language tests where similar vocabulary items often appear as distractors.

Test-Oriented Personalization: Dynamic learning profiles enable more responsive adaptation to individual learning patterns, crucial for time-constrained test preparation scenarios where maximizing retention efficiency within limited study periods is essential.

7 Conclusion

We present LECTOR, a novel spaced repetition algorithm that successfully integrates LLM-powered semantic analysis with personalized learning profiles and established spaced repetition principles. Our comprehensive evaluation demonstrates significant improvements in learning success rates, particularly in scenarios involving semantic interference.

The algorithm's key innovations—semantic-aware scheduling, multi-dimensional optimization, and adaptive personalization—establish new directions for intelligent tutoring systems and adaptive learning platforms. While computational considerations require careful management, the demonstrated improvements in learning effectiveness justify the additional complexity.

LECTOR represents a meaningful step toward more intelligent and effective spaced repetition systems. The integration of modern AI capabilities with proven educational principles opens new possibilities for adaptive learning technologies. Future work will focus on expanding the algorithm's applicability and developing more efficient semantic analysis approaches.

Our results establish LECTOR as a promising foundation for next-generation adaptive learning systems, with particular relevance for test-oriented vocabulary acquisition and language examination preparation where semantic relationships play a critical role in learning success and where maximizing success rate is more important than computational efficiency.

References

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Saxena, Shaleen Mandal, et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901, 2020.

Shana K Carpenter, Nicholas J Cepeda, Doug Rohrer, Sean HK Kang, and Harold Pashler. Using spacing to enhance diverse forms of learning: Review of recent research and implications for instruction. *Educational psychology review*, 24(3):369–378, 2012.

Albert T Corbett and John R Anderson. Knowledge tracing: Modeling the acquisition of procedural knowledge. *User modeling and user-adapted interaction*, 4(4):253–278, 1994.

DeepSeek-AI, Aixin Liu, Bei Feng, Bing Xue, Bingxuan Wang, Bochao Wu, Chengda Lu, Chenggang Zhao, Chengqi Deng, Chenyu Zhang, et al. Deepseek-v3 technical report, dec 2024. URL https://arxiv.org/abs/2412.19437. arXiv preprint arXiv:2412.19437.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, volume 1, pp. 4171–4186, 2019.

Qingxiu Dong, Lei Li, Damai Dai, Ce Zheng, Jingyuan Ma, Rui Li, Heming Xia, Jingjing Xu, Zhiyong Wu, Tianyu Liu, et al. A survey on in-context learning. *arXiv preprint arXiv:2301.00234*, 2023.

Hermann Ebbinghaus. Memory: A contribution to experimental psychology. 1885.

Jordan Hoffmann, Sebastian Borgeaud, Arthur Mensch, Elena Buchatskaya, Trevor Cai, Eliza Rutherford, Diego de Las Casas, Lisa Anne Hendricks, Johannes Welbl, Aidan Clark, Tom Hennigan, Eric Noland, Katie Millican, George van den Driessche, Bogdan Damoc, Aurelia Guy, Simon Osindero, Karen Simonyan, Erich Elsen, Jack W. Rae, Oriol Vinyals, and Laurent Sifre. Training compute-optimal large language models, 2022.

Jacob Hume. A Large Language Model-Infused Platform for Blending Spaced Repetition and Immersion in Mandarin Vocabulary Acquisition. In *Proceedings of the Innovation in Language Learning Conference 2024*, nov 2024.

Jared Kaplan, Sam McCandlish, Tom Henighan, Tom B. Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeffrey Wu, and Dario Amodei. Scaling laws for neural language models, 2020.

Enkelejda Kasneci, Kathrin Sessler, Stefan Küchemann, Maria Bannert, Daryna Dementieva, Frank Fischer, Urs Gasser, Georg Groh, Stephan Günnemann, Eyke Hüllermeier, et al. Chatgpt for good? on opportunities and challenges of large language models for education. *Learning and individual differences*, 103:102274, 2023.

Jarrett Ye Liu et al. Free spaced repetition scheduler. https://github.com/open-spaced-repetition/fsrs4anki, 2023. Accessed: 2024-08-03.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. Distributed representations of words and phrases and their compositionality. *Advances in neural information processing systems*, 26, 2013.

Sewon Min, Xinxi Lyu, Ari Holtzman, Mikel Artetxe, Mike Lewis, Hannaneh Hajishirzi, and Luke Zettlemoyer. Rethinking the role of demonstrations: What makes in-context learning work? *arXiv preprint arXiv:2202.12837*, 2022.

Chris Piech, Jonathan Bassen, Jonathan Huang, Surya Ganguli, Mehran Sahami, Leonidas J Guibas, and Jascha Sohl-Dickstein. Deep knowledge tracing. In *Advances in neural information processing systems*, pp. 505–513, 2015.

Henry L Roediger and Jeffrey D Karpicke. The power of testing memory: Basic research and implications for educational practice. *Perspectives on psychological science*, 1(3):181–210, 2006.

Burr Settles and Brendan Meeder. A trainable spaced repetition model for language learning. *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics*, 2016.

Jingyong Su, Junyao Ye, Liqiang Nie, Yilong Cao, and Yongyong Chen. Optimizing spaced repetition schedule by capturing the dynamics of memory. *IEEE Transactions on Knowledge and Data Engineering*, 35(10):10085–10097, 2023. doi: 10.1109/TKDE.2023.3251721.

Jason Wei, Yi Tay, Rishi Bommasani, Colin Raffel, Barret Zoph, Sebastian Borgeaud, Dani Yogatama, Maarten Bosma, Denny Zhou, Donald Metzler, et al. Emergent abilities of large language models. *Transactions on Machine Learning Research*, 2022.

Piotr A Wozniak, Edward J Gorzelanczyk, and Janusz A Murakowski. Optimization of repetition spacing in the practice of learning. *Acta neurobiologiae experimentalis*, 50(1):59–62, 1990.