BIRD: Bi-Level Operator Scheduling for Black-Box Attacks on Large Language Models

Anonymous ACL submission

Abstract

Black-box attacks on LLM-based classifiers seek adversarial inputs that induce incorrect predictions without access to model parameters, gradients, or logits. Existing confidence-guided attacks typically rely on a fixed perturbation operator or a manually specified operator set, overlooking that operator utility varies across inputs, target models, and attack trajectories. We formulate black-box LLM attacks as an online operator scheduling problem and propose BIRD (**BI**-level Bandit-d**R**iven Operator Sche**D**uling), a bi-level framework for automatic black-box attacks. BIRD maintains a heterogeneous pool of perturbation operators and adaptively schedules them using elicited confidence feedback. At the upper level, a sliding-window UCB scheduler selects operators according to cost-aware rewards; at the lower level, confidence-guided candidate selection accepts edits that reduce confidence in the original prediction or flip the label. Experiments on three benchmarks and three instruction-tuned LLMs show that BIRD improves average attack success from 27.71 to 36.60 over a strong confidence-guided baseline, while preserving comparable semantic similarity and practical query costs. Code is available at https://anonymous.4open.science/r/BIRD.

1 Introduction

Black-box attacks on large language models (LLMs) aim to generate adversarial inputs that induce incorrect predictions without accessing model parameters, gradients, or internal logits (Maheshwary et al., 2021; Liu et al., 2023; Formento et al., 2025). This setting is increasingly important because many deployed LLM systems are only exposed through APIs, where attackers can observe outputs but cannot inspect the underlying decision process (Ye et al., 2022; Liu et al., 2023; Formento et al., 2025). Unlike white-box attacks, black-box textual attacks must search over a discrete space of input modifications under limited query budgets (Ebrahimi et al., 2018; Formento et al., 2025; Yu et al., 2024). The central difficulty lies not only in generating textual perturbations, but also in deciding which perturbation mechanism should be applied at each attack step with sparse feedback.

A large body of black-box textual attacks has focused on designing effective perturbation operators, ranging from lexical and contextual transformations to character-level or paraphrase-style edits (Alzantot et al., 2018; Ren et al., 2019; Jin et al., 2020; Gao et al., 2018; Li et al., 2018; Garg and Ramakrishnan, 2020; Li et al., 2021a; Formento et al., 2025; Iyyer et al., 2018). More recent confidence-guided attacks show that elicited confidence provides a more informative signal than hard labels for ranking candidate perturbations by their impact on the target model's prediction confidence (Formento et al., 2025). Despite this progress, existing methods remain *operator-centric*: they bind the attack process to a fixed perturbation operator, such as word substitution, or rely on a manually specified combination of operators (Morris et al., 2020; Zeng et al., 2021). In other words, the perturbation operator is treated as a static recipe rather than a decision variable to be adapted during the attack.

This static view overlooks a key property of black-box LLM attacks: operator utility is input-dependent, model-dependent, and trajectory-dependent. Different operators expose complementary vulnerabilities of the target model. Lexical or contextual transformations better preserve semantics and fluency but may be insufficient or query-expensive, whereas surface-form transformations such as word-internal swaps or homoglyph replacements can be effective for certain samples while introducing form-level distortions (Jin et al., 2020; Garg and Ramakrishnan, 2020; Li et al., 2021a; Gao et al., 2018; Li et al., 2018; Boucher et al., 2022; Eger et al., 2019). Moreover, an operator that is useful early in the attack may become ineffective after several edits, while another operator may be-

come useful when the current input moves closer to the decision boundary. Thus, the bottleneck is not merely the lack of perturbation operators, but the lack of an adaptive mechanism to schedule heterogeneous operators under limited feedback.

This introduces three challenges. **(i)** Confidence feedback is informative but limited. Although elicited confidence provides finer-grained guidance than hard labels, it remains noisy, query-expensive, and fundamentally different from gradients or logits (Li et al., 2021b; Yu et al., 2024; Tian et al., 2023a), requiring lightweight online decisions under practical query budgets. **(ii)** Operator utility is heterogeneous and non-stationary. No single perturbation operator consistently dominates across datasets, target models, input instances, and attack stages. **(iii)** Simply composing multiple operators is insufficient. Static combinations may enlarge the search space and sometimes improve attack success, but they also introduce unstable cost-quality trade-offs, often increasing query cost, running time, or language-model perplexity, while depending heavily on manual operator-set selection (Morris et al., 2020; Zeng et al., 2021; Li et al., 2021a). These challenges naturally motivate viewing black-box LLM attacks as an online operator scheduling problem under limited confidence feedback.

To address this problem, we propose BIRD (**BI**-level Bandit-d**R**iven Operator Sche**D**uling), a confidence-feedback-driven framework for black-box LLM attacks. BIRD decouples the attack process into two levels of decision-making. The upper level decides *how* to perturb the current input, while the lower level decides *which* perturbation should be accepted. At the upper level, BIRD maintains a heterogeneous pool of perturbation operators and treats each operator as an arm in a multi-armed bandit (Auer et al., 2002). It estimates an upper confidence bound score for each operator using a cost-aware reward and a sliding window of recent attempts, balancing the exploitation of operators and the exploration of under-tested ones (Auer et al., 2002). At the lower level, the selected operator generates candidate perturbations, and BIRD uses elicited confidence to accept the candidate that reduces the confidence of the original prediction or directly flips the predicted label. In this way, BIRD uses confidence feedback only for lightweight online scheduling and candidate acceptance, continuously updates operator preferences along the attack trajectory, and avoids the unstable cost-quality trade-offs caused by brute-force operator composition. This allows the attack to allocate queries according to the evolving utility of different operators instead of committing to a fixed perturbation strategy throughout the search process.

We conduct experiments on benchmark datasets and multiple instruction-tuned LLMs to evaluate BIRD. Our experiments examine whether adaptive operator scheduling improves attack success over strong single-operator confidence-guided attacks, whether different perturbation operators exhibit heterogeneous utility across datasets and target models, whether static operator combinations suffer from unstable cost-quality trade-offs, and whether BIRD learns interpretable operator-switching behavior along attack trajectories. Results show that BIRD achieves stronger attack success rates than competing approaches while maintaining comparable semantic similarity and practical query costs. Further analyses show that no single operator dominates, static combinations require manual selection and may increase cost or fluency degradation, and BIRD dynamically switches among lexical, contextual, and surface-form perturbations according to the evolving attack state. This paper makes three contributions:

❶ ***Bi-Level Online Operator Scheduling***. We formulate black-box LLM attacks as a bi-level online operator scheduling problem, shifting the focus from designing a fixed perturbation operator to adaptively selecting heterogeneous operators under confidence-based feedback.

❷ ***Bandit-Driven Attack Framework***. We propose BIRD, a confidence-feedback-driven framework that combines an upper-level UCB scheduler over heterogeneous perturbation operators with a lower-level confidence-guided search for effective adversarial edits.

❸ ***Stronger and Interpretable Attacks***. Experiments and analyses show that BIRD improves over strong single-operator confidence-guided attacks while preserving comparable semantic similarity. Further ablations and case studies reveal the heterogeneous and trajectory-dependent behavior of different perturbation operators.

2 Related Work

2.1 Black-Box Attacks on LLMs

Textual adversarial attacks aim to modify input texts while preserving their original semantics and inducing incorrect model predictions. Early black-box attacks mainly focus on designing per-

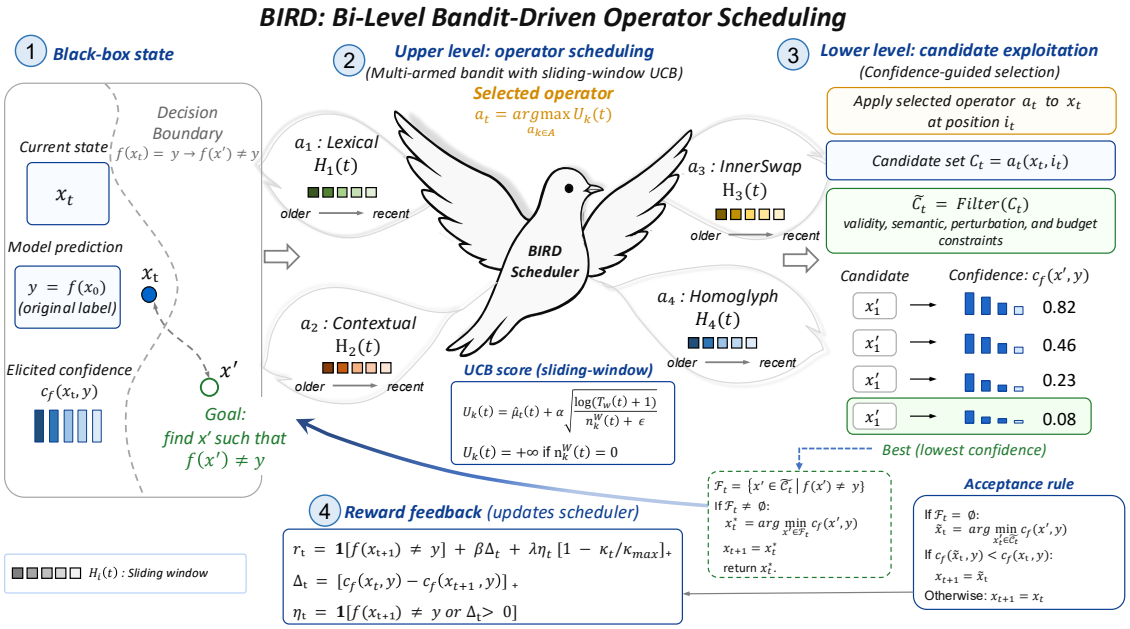

Figure 1: Overview of BIRD. Given the current black-box attack state x_t, BIRD formulates operator selection as an online bandit problem. The upper-level scheduler maintains sliding-window reward histories for heterogeneous perturbation operators and selects the operator with the largest UCB score. The lower level applies the selected operator to generate valid candidates, queries the target model, and accepts the candidate that either flips the original prediction or maximally reduces the elicited confidence $c_f(\cdot, y)$. The resulting prediction flip, confidence reduction, and query cost are converted into an operator-level reward, which is fed back to update future scheduling decisions.

turbation operators, such as character-level noise, synonym or embedding-based substitution, insertion and deletion, contextual replacement, and paraphrase-style rewriting, together with search heuristics such as word-importance ranking or genetic search (Alzantot et al., 2018; Ren et al., 2019; Gao et al., 2018; Li et al., 2018; Jin et al., 2020; Garg and Ramakrishnan, 2020; Li et al., 2021a; Feng et al., 2021; Bayer et al., 2022). Representative methods such as TextFooler (Jin et al., 2020), BAE (Garg and Ramakrishnan, 2020), and CLARE (Li et al., 2021a) generate adversarial examples through fixed transformation rules combined with semantic or grammatical constraints. Recent hard-label attacks further study more restricted settings where only predicted labels are available, emphasizing query efficiency and semantic preservation under limited feedback (Ye et al., 2022; Liu et al., 2023; Yu et al., 2024). With the deployment of LLMs through closed APIs, recent work explores alternative feedback signals for black-box attack optimization (Maheshwary et al., 2021; Formento et al., 2025). CEAttack shows that elicited confidence can provide a more informative signal than hard labels for guiding textual attacks on LLMs (Formento et al., 2025), while recent work further studies prompt-induced confidence variation for black-box attacks (Chen et al., 2025). Overall, existing methods provide strong perturbation recipes or useful feedback signals, but the perturbation mechanism is usually predefined rather than adaptively scheduled during the attack process.

2.2 Adaptive Operator Selection

The problem of dynamically selecting among a pool of search operators based on recent performance has been extensively studied as Adaptive Operator Selection (AOS) in evolutionary computation (Fialho et al., 2010; Maturana et al., 2010). AOS decomposes into two components: credit assignment, which quantifies each operator's recent contribution, and operator selection, which chooses the next operator given accumulated credits. Bandit-based formulations, where each operator corresponds to an arm and UCB-style policies balance exploration and exploitation, have proven particularly effective for handling non-stationary operator utility (Auer et al., 2002; Fialho et al., 2008; Li et al., 2013). A sliding window of recent rewards allows the scheduler to adapt as operator effectiveness evolves over the search trajectory (Fialho et al., 2010). While AOS has been primarily studied in numerical and combinatorial optimization, concurrent work explores bandit-guided strategy selection for LLM jailbreaking (Doumbouya et al., 2025; Ramesh et al., 2025), targeting safety alignment circumvention with compositional

prompt rewriting rather than textual perturbation scheduling for classification attacks. BIRD adapts the bandit-based AOS paradigm to black-box adversarial attacks on LLM classifiers: each perturbation operator is an arm, confidence reduction serves as the reward signal, and a sliding-window UCB scheduler handles the trajectory-dependent non-stationarity of operator utility. To our knowledge, this is the first application of AOS-style adaptive scheduling to select among heterogeneous textual perturbation operators under confidence-based feedback.

3 The Proposed BIRD

3.1 Problem Setup and Operator Pool

We consider a black-box LLM-based classifier f. Given an input text x, the attacker can only query f and observe its predicted label $\hat{y} = f(x)$ together with an elicited confidence score. The attacker has no access to model parameters, gradients, hidden states, logits, or token probabilities. For a label $\ell \in \mathcal{Y}$, we denote the elicited confidence assigned to ℓ by $c_f(x, \ell) \in [0, 1]$.

Given an input x_0, we denote its original prediction as $y = f(x_0)$. Following standard attack evaluation, we only attack originally correctly classified examples, so y is identical to the ground-truth label in evaluation. The attack aims to find an adversarial example x' such that:

$$\begin{aligned} f(x') \neq y, \qquad & \mathrm{Sim}(x_0, x') \geq \tau, \\ d(x_0, x') \leq \epsilon, \qquad & Q(x_0 \to x') \leq B. \end{aligned} \tag{1}$$

where $\mathrm{Sim}(\cdot,\cdot)$ measures semantic similarity, $d(\cdot,\cdot)$ denotes perturbation magnitude, $Q(\cdot)$ is the number of target-model queries, and B is the query budget.

We view the attack as a constrained online decision process. At step t, the attacker observes the current text state x_t, selects a perturbation operator a_t, generates valid candidates, and decides whether to update x_t. This can be abstracted as learning an operator scheduling policy π:

$$\begin{aligned} \max_{\pi} \quad & \mathbb{E}_{a_t \sim \pi(\cdot|x_t)} \left[\sum_{t=0}^{T-1} R(x_t, a_t) \right] \\ \text{s.t.} \quad & x_{t+1} \in a_t(x_t), \quad \mathrm{Sim}(x_0, x_{t+1}) \geq \tau, \\ & Q \leq B, \quad t = 0, \ldots, T-1. \end{aligned} \tag{2}$$

Eq. (2) is not solved by white-box optimization; instead, it motivates a lightweight online scheduler driven by black-box confidence feedback.

BIRD maintains a heterogeneous perturbation operator pool:

$$\mathcal{A} = \{a_1, a_2, \ldots, a_K\}, \tag{3}$$

where each operator a_k maps the current text state x_t and a candidate position i_t to a candidate set:

$$\mathcal{C}_t^{(k)} = a_k(x_t, i_t). \tag{4}$$

In our implementation, we instantiate $\mathcal{A}$ with four operators chosen to cover three complementary perturbation granularities, each targeting a different stage of the model's text-processing pipeline (Jin et al., 2020; Garg and Ramakrishnan, 2020; Gao et al., 2018; Boucher et al., 2022): (1) *CEAttack-style lexical substitution* operates at the **lexical level**, replacing a word with an embedding-similar alternative that perturbs the token embedding lookup (Jin et al., 2020; Formento et al., 2025); (2) *masked language model replacement* operates at the **contextual level**, filling a masked position with a context-conditioned alternative that may alter subtle semantic cues (Garg and Ramakrishnan, 2020; Li et al., 2020); (3) *word-internal character swap* and (4) *homoglyph replacement* operate at the **surface-form level**, disrupting subword tokenization and Unicode normalization respectively (Gao et al., 2018; Boucher et al., 2022). This design ensures the pool is heterogeneous by construction: each operator exposes a structurally different class of model vulnerability, so adaptive scheduling can discover attack paths unreachable by any single granularity.

3.2 Bi-Level Scheduling Loop

As illustrated in Figure 1, BIRD follows a four-stage feedback loop. First, the black-box state contains the current text x_t, the original prediction y, and the confidence $c_f(x_t, y)$. Second, the upper level selects an operator a_t from $\mathcal{A}$ using a bandit-based scheduling score. Third, the lower level applies a_t at position i_t, filters the generated candidates, and accepts the candidate that flips the prediction or most reduces $c_f(\cdot, y)$. Finally, the prediction flip, confidence reduction, and query cost are converted into an operator-level reward, which updates the scheduler for future steps. Algorithm 1 summarizes the full online loop in Appendix A.

This loop differs from static operator composition. Static composition fixes an operator set or order before the attack begins, whereas BIRD updates operator preferences online according to recent attack feedback. Therefore, query budget is allocated

according to the evolving utility of operators rather than a manually specified perturbation recipe.

3.3 Upper-Level Operator Scheduling

The upper level treats each perturbation operator as an arm in a multi-armed bandit. Since operator utility can change across inputs, target models, and attack stages, BIRD uses a sliding-window UCB score rather than a fixed operator prior.

After applying operator a_t, BIRD computes its reward from prediction flipping, confidence reduction, and query cost. Let x_{t+1} be the state after lower-level exploitation. We define

$$\Delta_t = [c_f(x_t, y) - c_f(x_{t+1}, y)]_+ . \tag{5}$$

$$r_t = \mathbb{I}[f(x_{t+1}) \neq y] + \beta \Delta_t + \lambda \eta_t \left[1 - \frac{\kappa_t}{\kappa_{\max}}\right]_+ \tag{6}$$

where $[z]_+ = \max(z, 0)$, β and λ weight confidence reduction and query efficiency, and κ_t is the number of target-model queries consumed by the current invocation, and $\kappa_{\max}$ is the maximum query cost allowed for one invocation. The indicator $\eta_t = \mathbb{I}[f(x_{t+1}) \neq y \text{ or } \Delta_t > 0]$ ensures that the efficiency bonus is assigned only when the operator makes useful attack progress. Thus, an operator receives high reward when it directly flips the prediction or reduces confidence in the original label with low query cost.

For each operator a_k, BIRD maintains a sliding-window reward history $\mathcal{H}_k(t)$, which stores the most recent W rewards obtained by a_k before step t. Its recent empirical utility is

$$\begin{aligned} \hat{\mu}_k(t) &= \frac{1}{|\mathcal{H}_k(t)|} \sum_{r \in \mathcal{H}_k(t)} r, \\ n_k^W(t) &= |\mathcal{H}_k(t)|, \quad T_W(t) = \sum_{m=1}^{K} n_m^W(t) \end{aligned} \tag{7}$$

The upper-level UCB score is then computed as

$$U_k(t) = \begin{cases} +\infty, & n_k^W(t) = 0, \\ \hat{\mu}_k(t) + \alpha \sqrt{\dfrac{\log(T_W(t)+1)}{n_k^W(t) + \varepsilon}}, & n_k^W(t) > 0. \end{cases} \tag{8}$$

where α controls exploration strength and ε is a small numerical constant. The scheduler selects:

$$a_t = \arg\max_{a_k \in \mathcal{A}} U_k(t). \tag{9}$$

The empirical term exploits recently effective operators, while the exploration bonus encourages under-tested ones. This operator-level scheduler avoids brute-force composition while still adapting to trajectory-dependent utility.

3.4 Lower-Level Candidate Exploitation

Given the selected operator a_t, the lower level instantiates it into concrete perturbations. BIRD first constructs a modifiable position set $\mathcal{I}(x_t)$ by excluding invalid positions such as punctuation, stopwords, and previously modified tokens, and then selects $i_t \in \mathcal{I}(x_t)$ following the same position-selection protocol as the confidence-guided baseline. The selected operator produces

$$\begin{aligned} \mathcal{C}_t &= a_t(x_t, i_t), \\ \widetilde{\mathcal{C}}_t &= \left\{ x' \in \mathcal{C}_t \;\middle|\; \begin{array}{l} \mathrm{Sim}(x_0, x') \geq \tau, \quad d(x_0, x') \leq \epsilon, \\ \Omega(x', x_t) = 1 \end{array} \right\}. \end{aligned} \tag{10}$$

where $\Omega(\cdot)$ denotes textual validity constraints, including syntactic consistency, repeated-modification prevention, and task-specific constraints. To respect the remaining query budget, $\widetilde{\mathcal{C}}_t$ is truncated when $|\widetilde{\mathcal{C}}_t| > B - Q$. If no valid candidate remains, the invocation receives zero reward.

For each valid candidate, BIRD queries the target model and first checks whether the prediction is flipped:

$$\mathcal{F}_t = \{x' \in \widetilde{\mathcal{C}}_t \mid f(x') \neq y\}. \tag{11}$$

If $\mathcal{F}_t \neq \emptyset$, the attack succeeds and BIRD returns the flipped candidate with the lowest confidence on the original label:

$$x_t^\star = \arg\min_{x' \in \mathcal{F}_t} c_f(x', y). \tag{12}$$

Otherwise, BIRD selects the candidate that most reduces the confidence of the original label and accepts it only if it improves over the current state:

$$\tilde{x}_t = \arg\min_{x' \in \widetilde{\mathcal{C}}_t} c_f(x', y), \tag{13}$$

$$x_{t+1} = \begin{cases} \tilde{x}_t, & c_f(\tilde{x}_t, y) < c_f(x_t, y), \\ x_t, & \text{otherwise.} \end{cases} \tag{14}$$

The accepted state x_{t+1} determines the confidence reduction Δ_t, query cost κ_t, and reward r_t, which are fed back to the upper-level scheduler.

4 Experiments

4.1 Experimental Setup

Datasets and target models. We evaluate BIRD on three classification benchmarks: SST-2 (Socher et al., 2013), AG-News (Zhang et al., 2015), and StrategyQA (Geva et al., 2021), covering sentiment classification, topic classification, and binary

Model	Dataset	AUA ↓		ASR ↑			SemSim ↑		PPL ↓		Avg.Q ↓	
		CE	BIRD	CE	BIRD	Δ	CE	BIRD	CE	BIRD	CE	BIRD
LLaMA-3	SST-2	74.30	**63.25**	18.14	**30.16**	**+12.02**	0.88	0.88	**106.41**	189.75	21.98	**19.30**
	AG-News	43.06	**40.04**	31.41	**35.39**	**+3.98**	**0.93**	0.92	**96.56**	132.21	**42.95**	57.63
	StrategyQA	32.67	**21.24**	45.67	**64.55**	**+18.88**	0.89	0.89	**206.23**	308.52	**8.50**	9.09
Mistral-7B	SST-2	70.98	**63.60**	19.81	**27.79**	**+7.98**	0.88	0.88	**97.31**	145.08	**23.14**	25.21
	AG-News	36.73	**34.11**	44.49	**48.46**	**+3.97**	0.92	0.92	**97.76**	138.36	**43.41**	57.03
	StrategyQA	36.21	**24.14**	39.26	**59.17**	**+19.91**	**0.90**	0.89	**177.94**	286.86	**8.71**	9.41
DeepSeek-V4	SST-2	79.20	**78.60**	15.74	**16.74**	**+1.00**	0.89	0.89	**124.47**	171.96	38.84	**8.65**
	AG-News	73.20	**72.40**	9.41	**11.06**	**+1.65**	0.93	0.93	**94.83**	128.44	**43.79**	58.00
	StrategyQA	49.30	**42.29**	25.44	**36.04**	**+10.60**	0.89	**0.90**	**149.64**	285.13	**8.91**	9.76
Average	–	55.07	**48.85**	27.71	**36.60**	**+8.89**	0.90	0.90	**127.91**	198.48	**26.25**	31.56

Table 1: Main attack performance against $\text{CEAttack}_{\text{rep}}$. CE denotes $\text{CEAttack}_{\text{rep}}$, and ΔASR denotes the absolute ASR improvement of BIRD. AUA, PPL, and Avg.Q are lower better; ASR and SemSim are higher better.

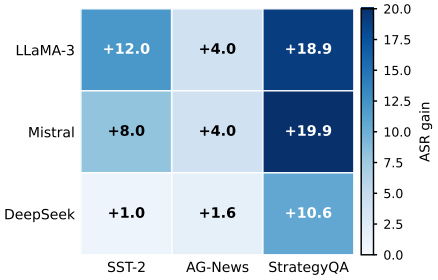

Figure 2: Absolute ASR improvement of BIRD over $\text{CEAttack}_{\text{rep}}$. BIRD improves ASR in all evaluated settings, with the largest gains on StrategyQA.

reasoning-based question answering. Following standard adversarial attack evaluation, we attack only examples that are originally classified correctly by the target model. We use three instruction-tuned LLMs as black-box classifiers: LLaMA-3-8B-Instruct (Touvron et al., 2023), Mistral-7B-Instruct-v0.3 (Jiang et al., 2023), and DeepSeek-V4-Flash (DeepSeek-AI, 2026).

Baselines and metrics. Our main baseline is $\text{CEAttack}_{\text{rep}}$ (Formento et al., 2025), a reproduced confidence-guided lexical substitution attack. We also evaluate four single-operator variants and several static operator compositions to diagnose the effect of operator heterogeneity and adaptive scheduling. All methods share the same confidence elicitation protocol (verbalized confidence with Dirichlet aggregation, k=20 for SST-2/AG-News, k=6 for StrategyQA), semantic filters, perturbation constraints, and query budget. We report Accuracy under attack(AUA), Attack success rate(ASR), Semantic similarity(SemSim), PPL, and Avg.Q; Succ.Q and wall-clock time are reported as additional cost diagnostics.

4.2 Main Attack Performance

BIRD consistently strengthens confidence-guided attacks. Table 1 compares BIRD with $\text{CEAttack}_{\text{rep}}$ under the same evaluation protocol. Across all nine model-dataset pairs, BIRD improves ASR over the reproduced confidence-guided baseline. On average, ASR increases from 27.71 to 36.60 and AUA decreases from 55.07 to 48.85. As shown in Figure 2, the gains are especially pronounced on StrategyQA, where BIRD improves ASR by 18.88 points on LLaMA-3-8B-Instruct, 19.91 points on Mistral-7B-Instruct-v0.3, and 10.60 points on DeepSeek-V4-Flash. Meanwhile, the average SemSim remains 0.90, suggesting that the higher attack success is not obtained by relaxing semantic preservation. The main quality cost is higher PPL in several settings, which reflects that the scheduler sometimes selects stronger surface-form perturbations when they are useful. Overall, adaptive operator scheduling improves attack success over a strong single-operator confidence-guided baseline while maintaining comparable sentence-level semantics.

4.3 Component Ablation

Table 3 isolates each scheduling component on LLaMA-3-8B-Instruct with SST-2 by removing one at a time. All three components contribute: replacing UCB with random selection causes the largest ASR drop (−14.31), confirming that informed operator allocation is essential even when the same operator pool is available. Removing the sliding window (using full-history averaging) reduces ASR by 6.42 points, showing that recent feedback better predicts near-future operator utility than lifetime statistics. Removing the cost-aware reward (λ=0) increases Avg.Q from 19.30 to 24.44 while ASR drops by 9.15, indicating that the efficiency term guides the scheduler toward operators that reduce confidence with fewer queries.

4.4 Cost and Quality Diagnostics

BIRD improves ASR with moderate additional cost. Table 2 summarizes the cost-quality profile of

Model	Method	ASR ↑	Avg.Q ↓	Succ.Q ↓	Avg. Time ↓	SemSim ↑	PPL ↓
LLaMA-3	$\text{CEAttack}_{\text{rep}}$	31.74	**24.48**	**26.58**	**7:05:20**	**0.900**	**136.40**
	BIRD	**43.37**	28.67	29.20	7:52:59	0.897	210.16
Mistral-7B	$\text{CEAttack}_{\text{rep}}$	34.52	**25.09**	**25.85**	**6:15:54**	**0.900**	**124.34**
	BIRD	**45.14**	30.55	30.95	7:18:32	0.897	190.10
DeepSeek-V4	$\text{CEAttack}_{\text{rep}}$	16.86	30.51	32.06	8:17:02	0.903	**122.98**
	BIRD	**21.28**	**25.47**	**25.79**	**7:04:16**	**0.907**	195.18
Average	$\text{CEAttack}_{\text{rep}}$	27.71	**26.25**	**26.21**	**7:12:45**	0.900	**127.91**
	BIRD	**36.60**	31.56	30.08	7:25:16	0.900	198.48

Table 2: Averaged cost and quality diagnostics by target model. ASR, Avg.Q, Avg. Time, SemSim, and PPL are averaged over the three datasets for each target model. Succ.Q is reported for LLaMA-3 and Mistral, where successful-attack query logs are complete. Avg.Q, Succ.Q, Avg. Time, and PPL are lower better; ASR and SemSim are higher better.

Variant	AUA ↓	ASR ↑	Avg.Q ↓
BIRD (full)	**63.25**	**30.16**	**19.30**
w/o UCB (Random)	75.73	15.85	26.97
w/o Sliding Window	69.52	23.74	23.54
w/o Cost-Aware Reward	70.87	21.01	24.44

Table 3: Component ablation of BIRD on LLaMA-3-8B-Instruct with SST-2. Each row removes one scheduling component from the full system.

Operator	ASR ↑	SemSim ↑	PPL ↓	Avg.Q ↓	Profile
$\text{CEAttack}_{\text{rep}}$	33.13	0.900	130.37	24.78	stable lexical
MaskedLM	20.52	**0.908**	**95.41**	15.59	fluent, weaker
InnerSwap	**38.05**	0.907	252.83	5.39	strong, disruptive
Homoglyph	11.95	**0.908**	156.82	**3.89**	cheap, weak

Table 4: Average single-operator profiles over six settings with LLaMA-3-8B-Instruct and Mistral-7B-Instruct-v0.3. No single operator dominates the attack-quality-cost trade-off.

BIRD. Compared with $\text{CEAttack}_{\text{rep}}$, BIRD improves average ASR by 8.89 points, while Avg.Q increases from 26.25 to 31.56. On the settings with complete successful-query logs, Succ.Q increases from 26.21 to 30.08. The average wall-clock time remains close, with BIRD requiring 7:25:16 compared with 7:12:45 for $\text{CEAttack}_{\text{rep}}$. We treat wall-clock time as a diagnostic rather than the primary efficiency metric because it can vary with hardware, batching, and model-serving conditions. The cost pattern is also model-dependent: on DeepSeek-V4, BIRD improves ASR while reducing both Avg.Q and average running time. These results suggest that the improvement is not obtained by unbounded query expansion; instead, BIRD reallocates the query budget toward operators that are more useful for the current attack state.

Quality trade-off. The main quality cost of BIRD is increased after-attack PPL. This is expected because adaptive scheduling may select stronger surface-form operators when lexical or contextual substitutions are insufficient. However, the average SemSim remains unchanged at 0.90. We therefore interpret PPL as a complementary quality signal: BIRD improves attack effectiveness while preserving sentence-level semantics, but may introduce less fluent surface forms in difficult cases.

4.5 Operator Heterogeneity Analysis

Perturbation operators expose complementary vulnerabilities. Table 4 shows that different operators optimize different aspects of the attack-quality-cost trade-off. InnerSwap achieves the highest average ASR, but its PPL is substantially higher, indicating stronger surface-form distortion. MaskedLM produces the lowest PPL and high SemSim, but its ASR is much lower. Homoglyph requires the fewest queries, but rarely succeeds as a standalone attack. $\text{CEAttack}_{\text{rep}}$ provides a stable lexical baseline, but it does not dominate across metrics. These results support the central motivation of BIRD: committing the whole trajectory to a single perturbation operator is suboptimal because operator utility is heterogeneous across effectiveness, fluency, semantic preservation, and query cost.

4.6 Static vs. Adaptive Scheduling

Static composition is strong but not a principled replacement for scheduling. Table 5 compares BIRD with manually specified operator compositions. Several static combinations obtain high ASR, confirming that heterogeneous operators are useful. Some combinations even achieve higher raw ASR than BIRD, especially those involving InnerSwap. However, their favorable metric varies substantially with the manually chosen operator set. For example, CE+InnerSwap achieves the highest average ASR but increases PPL; CE+MaskedLM obtains the lowest PPL but requires many more target-model queries; InnerSwap+Homoglyph is query-efficient but introduces the largest surface-form distortion. Thus, static composition shifts the burden to manual operator-set selection and exposes unstable cost-quality trade-offs. It also cannot update operator preferences along the at-

Strategy	#Ops	ASR↑	SemSim↑	PPL↓	Avg.Q↓	Observed trade-off
$\text{CEAttack}_{\text{rep}}$	1	33.13	**0.900**	130.37	24.78	fixed lexical recipe
BIRD	adap.	44.25	0.897	200.13	29.61	adaptive scheduling
CE + MaskedLM	2	42.98	0.892	**119.98**	55.35	lowest PPL, highest Avg.Q
CE + InnerSwap	2	**53.40**	0.893	232.29	35.71	highest ASR, high PPL
MaskedLM + InnerSwap	2	52.38	0.890	227.68	27.17	high ASR, fixed pair
InnerSwap + Homoglyph	2	45.26	0.897	271.43	**11.90**	lowest Avg.Q, highest PPL
CE + MaskedLM + InnerSwap	3	49.37	0.895	200.77	36.64	larger static search space
CE + InnerSwap + Homoglyph	3	45.99	0.893	219.94	26.86	no clear quality dominance
MaskedLM + InnerSwap + Homoglyph	3	45.92	0.897	217.61	20.75	manual triplet selection

Table 5: Static-composition diagnostic averaged over six settings with LLaMA-3-8B-Instruct and Mistral-7B-Instruct-v0.3. Static combinations confirm that heterogeneous operators are useful, but their favorable metric varies across manually specified operator sets.

Step	Operator	Edit	Score	Pred.	Decision	State fragment after decision
0	–	original input	0.307	pos.	–	*this film ... vast collective memory ... combatants*
1	InnerSwap	reorder phrase	0.495	pos.	ACCEPT	*combatants film ... collective memory ... the this*
2	Homoglyph	force → [hg]force	0.491	pos.	REJECT	unchanged from Step 1
3	CEAttack	vast → gargantuan	0.499	pos.	ACCEPT	*gargantuan collective memory ... the this*
4	MaskedLM	seems → seem	0.503	pos.	ACCEPT	*combatants film seem to bubble up ...*
5	InnerSwap	reorder phrase	0.532	neg.	**SUCCESS**	*collective up ... gargantuan bubble memory ...*

Table 6: A successful BIRD trajectory on SST-2 with LLaMA-3-8B-Instruct. Score denotes the logged attack-side progress score; higher values indicate stronger progress toward flipping the original prediction. The table highlights accepted edits, rejected trials, and operator revisiting.

tack trajectory. In contrast, BIRD provides an online scheduling mechanism that adapts operator preferences using recent confidence-based feedback, rather than committing to a fixed operator set throughout the attack.

4.7 Operator Switching Behavior

BIRD adapts along the attack trajectory. Table 6 illustrates how BIRD switches among operators within a single successful attack. The scheduler first selects InnerSwap, which substantially increases the attack-side score and is therefore accepted. It then explores Homoglyph, but the candidate does not improve the score and is rejected by the lower-level acceptance rule. The attack subsequently switches to lexical substitution and contextual inflection before returning to InnerSwap, which finally flips the prediction. This trajectory shows that operator utility is not fixed: an operator can be ineffective at one state but useful later after the input has moved to a different region of the attack trajectory. Additional qualitative trajectories in the appendix show that some examples are solved by a single dominant operator, whereas others require switching among lexical, contextual, and surface-form perturbations. Overall, BIRD does not merely combine stronger perturbation operators; it uses confidence feedback to allocate them adaptively across different attack states.

5 Conclusion

This paper identifies a key limitation of existing confidence-guided black-box attacks: they bind the attack process to a fixed perturbation operator, overlooking that operator utility is input-dependent, model-dependent, and trajectory-dependent. We formulate black-box LLM attacks as an online operator scheduling problem and propose BIRD, a bi-level bandit-driven framework that decouples operator selection from candidate exploitation. At the upper level, a sliding-window UCB scheduler adaptively allocates query budget across a heterogeneous operator pool covering lexical, contextual, and surface-form perturbation granularities. At the lower level, confidence-guided candidate selection accepts edits that reduce the target model's belief in its original prediction. Experiments on three benchmarks and three instruction-tuned LLMs show that BIRD improves average attack success rate from 27.71 to 36.60 over a strong confidence-guided baseline, while preserving comparable semantic similarity. Further analyses reveal that no single perturbation operator dominates across settings, that static operator compositions shift the burden to manual selection with unstable cost-quality tradeoffs, and that BIRD learns interpretable operator-switching behavior along attack trajectories. These findings suggest that adaptive scheduling is a more principled approach to leveraging heterogeneous perturbation strategies than either committing to one operator or brute-force composition.

6 Limitations

While BIRD demonstrates consistent improvements in attack success rate across multiple datasets and target models, we acknowledge several limitations that point to future research directions.

Language scope. Our experiments are conducted exclusively on English-language benchmarks. Although the bi-level scheduling mechanism is language-agnostic in principle, some perturbation operators in the current pool (e.g., synonym substitution, masked language model replacement) rely on English-specific resources or models. Extending BIRD to morphologically richer or typologically diverse languages may require additional language-specific operators or multilingual embedding models, which we leave for future exploration.

Task coverage. We evaluate BIRD on classification tasks (sentiment analysis, topic classification, and binary QA). These tasks provide well-defined label spaces and confidence signals suitable for bandit-based scheduling. Applying BIRD to generation tasks (e.g., summarization, dialogue) or tasks with open-ended outputs would require rethinking the reward formulation and confidence elicitation protocol, as prediction flipping and label confidence are less naturally defined in those settings.

Operator pool design. The current operator pool consists of four representative perturbation operators covering lexical, contextual, and surface-form transformations. While our ablations confirm that these operators exhibit complementary utility, we did not exhaustively explore all possible operator types (e.g., sentence-level paraphrasing, syntactic restructuring). The modular design of BIRD allows straightforward integration of additional operators, but the optimal pool composition for different attack scenarios remains an open question.

Perplexity increase. As shown in our experimental results, BIRD tends to produce adversarial examples with higher perplexity compared to the single-operator baseline, because surface-form operators (character swap, homoglyph) can introduce less fluent text fragments when selected by the scheduler. Although semantic similarity remains comparable, future work could incorporate a fluency-aware component into the reward function to better balance attack effectiveness and linguistic naturalness.

Computational overhead. The UCB scheduling mechanism itself introduces negligible computational cost. However, maintaining a heterogeneous operator pool means that each scheduling step may invoke different operator backends (e.g., a masked language model), which can vary in latency. In practice, the overall wall-clock time of BIRD is comparable to the baseline (within $1.2\times$), but deployments with strict latency constraints may benefit from operator-specific parallelization strategies.

7 Ethical Considerations

This work studies black-box adversarial attacks for evaluating the robustness of LLM-based classifiers. As with other adversarial robustness research, the proposed method has a dual-use aspect: techniques that reveal model vulnerabilities could be misapplied if used against systems without authorization. Our goal is to provide a controlled evaluation framework that helps researchers and practitioners better understand when confidence-guided classifiers are vulnerable and how different perturbation operators affect robustness. The experiments are limited to standard benchmark classification tasks, and the method is evaluated under explicit query budgets and semantic-preservation constraints rather than in open-ended deployment settings.

We do not collect private user data or involve human subjects. The evaluation uses public text classification benchmarks and reports aggregate attack metrics, such as attack success rate, semantic similarity, perplexity, and query cost. The qualitative examples are included only to illustrate operator-switching behavior, not to target any individual or real-world user. Researchers who use this method should run it only on models and datasets that they own or are authorized to evaluate, and should interpret generated adversarial examples as diagnostic artifacts for robustness assessment rather than as deployable content.

The main intended benefit of this work is to support more reliable evaluation of LLM-based classifiers under realistic black-box feedback. By identifying how adaptive operator scheduling changes the attack-quality-cost trade-off, the results can inform stronger evaluation protocols and motivate defenses that account for heterogeneous perturbations. If code or artifacts are released, they should be accompanied by documentation of the intended research use, evaluation assumptions, and recommended constraints such as query budgets and se-

mantic filters.

References

Moustafa Alzantot, Yash Sharma, Ahmed Elgohary, Bo-Jhang Ho, Mani Srivastava, and Kai-Wei Chang. 2018. Generating natural language adversarial examples. In *Proceedings of the 2018 conference on empirical methods in natural language processing*, pages 2890–2896.

Peter Auer, Nicolo Cesa-Bianchi, and Paul Fischer. 2002. Finite-time analysis of the multiarmed bandit problem. *Machine learning*, 47(2):235–256.

Markus Bayer, Marc-André Kaufhold, and Christian Reuter. 2022. A survey on data augmentation for text classification. *ACM Computing Surveys*, 55(7):1–39.

Nicholas Boucher, Ilia Shumailov, Ross Anderson, and Nicolas Papernot. 2022. Bad characters: Imperceptible nlp attacks. In *2022 IEEE symposium on security and privacy (SP)*, pages 1987–2004. IEEE.

Daniel Cer, Yinfei Yang, Sheng-yi Kong, Nan Hua, Nicole Limtiaco, Rhomni St John, Noah Constant, Mario Guajardo-Cespedes, Steve Yuan, Chris Tar, and 1 others. 2018. Universal sentence encoder. *arXiv preprint arXiv:1803.11175*.

Meina Chen, Yihong Tang, and Kehai Chen. 2025. Exploiting prompt-induced confidence for black-box attacks on llms. In *Findings of the Association for Computational Linguistics: EMNLP 2025*, pages 12897–12903.

DeepSeek-AI. 2026. Deepseek-v4: Towards highly efficient million-token context intelligence. Technical report, DeepSeek-AI. Accessed: 2026-05-25.

Moussa Koulako Bala Doumbouya, Ananjan Nandi, Gabriel Poesia, Davide Ghilardi, Anna Goldie, Federico Bianchi, Dan Jurafsky, and Christopher Manning. 2025. h4rm3l: A language for composable jailbreak attack synthesis. In *International Conference on Learning Representations*, volume 2025, pages 57253–57286.

Javid Ebrahimi, Anyi Rao, Daniel Lowd, and Dejing Dou. 2018. Hotflip: White-box adversarial examples for text classification. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 31–36.

Steffen Eger, Gözde Gül Şahin, Andreas Rücklé, Ji-Ung Lee, Claudia Schulz, Mohsen Mesgar, Krishnkant Swarnkar, Edwin Simpson, and Iryna Gurevych. 2019. Text processing like humans do: Visually attacking and shielding nlp systems. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 1634–1647.

Steven Y Feng, Varun Gangal, Jason Wei, Sarath Chandar, Soroush Vosoughi, Teruko Mitamura, and Eduard Hovy. 2021. A survey of data augmentation approaches for nlp. In *Findings of the association for computational linguistics: ACL-IJCNLP 2021*, pages 968–988.

Álvaro Fialho, Luís Da Costa, Marc Schoenauer, and Michele Sebag. 2008. Extreme value based adaptive operator selection. In *International conference on parallel problem solving from nature*, pages 175–184. Springer.

Álvaro Fialho, Luis Da Costa, Marc Schoenauer, and Michele Sebag. 2010. Analyzing bandit-based adaptive operator selection mechanisms. *Annals of Mathematics and Artificial Intelligence*, 60(1):25–64.

Brian Formento, Chuan Sheng Foo, and See-Kiong Ng. 2025. Confidence elicitation: A new attack vector for large language models. *arXiv preprint arXiv:2502.04643*.

Ji Gao, Jack Lanchantin, Mary Lou Soffa, and Yanjun Qi. 2018. Black-box generation of adversarial text sequences to evade deep learning classifiers. In *2018 IEEE security and privacy workshops (SPW)*, pages 50–56. IEEE.

Siddhant Garg and Goutham Ramakrishnan. 2020. Bae: Bert-based adversarial examples for text classification. In *Proceedings of the 2020 conference on empirical methods in natural language processing (EMNLP)*, pages 6174–6181.

Mor Geva, Daniel Khashabi, Elad Segal, Tushar Khot, Dan Roth, and Jonathan Berant. 2021. Did aristotle use a laptop? a question answering benchmark with implicit reasoning strategies. *Transactions of the Association for Computational Linguistics*, 9:346–361.

Mohit Iyyer, John Wieting, Kevin Gimpel, and Luke Zettlemoyer. 2018. Adversarial example generation with syntactically controlled paraphrase networks. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1875–1885.

Albert Q. Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, Lélio Renard Lavaud, Marie-Anne Lachaux, Pierre Stock, Teven Le Scao, Thibaut Lavril, Thomas Wang, Timothée Lacroix, and William El Sayed. 2023. Mistral 7b. *Preprint*, arXiv:2310.06825.

Di Jin, Zhijing Jin, Joey Tianyi Zhou, and Peter Szolovits. 2020. Is bert really robust? a strong baseline for natural language attack on text classification and entailment. In *Proceedings of the AAAI conference on artificial intelligence*, volume 34, pages 8018–8025.

Dianqi Li, Yizhe Zhang, Hao Peng, Liqun Chen, Chris Brockett, Ming-Ting Sun, and William B Dolan. 2021a. Contextualized perturbation for textual adversarial attack. In *Proceedings of the 2021 conference of the North American chapter of the association for computational linguistics: Human language technologies*, pages 5053–5069.

Dianqi Li, Yizhe Zhang, Hao Peng, Liqun Chen, Chris Brockett, Ming-Ting Sun, and William B Dolan. 2021b. Contextualized perturbation for textual adversarial attack. In *Proceedings of the 2021 conference of the North American chapter of the association for computational linguistics: Human language technologies*, pages 5053–5069.

Jinfeng Li, Shouling Ji, Tianyu Du, Bo Li, and Ting Wang. 2018. Textbugger: Generating adversarial text against real-world applications. *arXiv preprint arXiv:1812.05271*.

Ke Li, Alvaro Fialho, Sam Kwong, and Qingfu Zhang. 2013. Adaptive operator selection with bandits for a multiobjective evolutionary algorithm based on decomposition. *IEEE Transactions on Evolutionary Computation*, 18(1):114–130.

Linyang Li, Ruotian Ma, Qipeng Guo, Xiangyang Xue, and Xipeng Qiu. 2020. Bert-attack: Adversarial attack against bert using bert. In *Proceedings of the 2020 conference on empirical methods in natural language processing (EMNLP)*, pages 6193–6202.

Stephanie Lin, Jacob Hilton, and Owain Evans. Teaching models to express their uncertainty in words, 2022. *URL https://arxiv. org/abs/2205.14334*.

Han Liu, Zhi Xu, Xiaotong Zhang, Xiaoming Xu, Feng Zhang, Fenglong Ma, Hongyang Chen, Hong Yu, and Xianchao Zhang. 2023. Sspattack: a simple and sweet paradigm for black-box hard-label textual adversarial attack. In *Proceedings of the AAAI conference on artificial intelligence*, volume 37, pages 13228–13235.

Rishabh Maheshwary, Saket Maheshwary, and Vikram Pudi. 2021. Generating natural language attacks in a hard label black box setting. In *Proceedings of the AAAI conference on artificial intelligence*, volume 35, pages 13525–13533.

Jorge Maturana, Frédéric Lardeux, and Frédéric Saubion. 2010. Autonomous operator management for evolutionary algorithms. *Journal of Heuristics*, 16(6):881–909.

John Morris, Eli Lifland, Jin Yong Yoo, Jake Grigsby, Di Jin, and Yanjun Qi. 2020. Textattack: A framework for adversarial attacks, data augmentation, and adversarial training in nlp. In *Proceedings of the 2020 conference on empirical methods in natural language processing: System demonstrations*, pages 119–126.

Aditya Ramesh, Shivam Bhardwaj, Aditya Saibewar, and Manohar Kaul. 2025. Efficient jailbreak attack sequences on large language models via multi-armed bandit-based context switching. In *The Thirteenth International Conference on Learning Representations*.

Shuhuai Ren, Yihe Deng, Kun He, and Wanxiang Che. 2019. Generating natural language adversarial examples through probability weighted word saliency. In *Proceedings of the 57th annual meeting of the association for computational linguistics*, pages 1085–1097.

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D Manning, Andrew Y Ng, and Christopher Potts. 2013. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 conference on empirical methods in natural language processing*, pages 1631–1642.

Katherine Tian, Eric Mitchell, Allan Zhou, Archit Sharma, Rafael Rafailov, Huaxiu Yao, Chelsea Finn, and Christopher D Manning. 2023a. Just ask for calibration: Strategies for eliciting calibrated confidence scores from language models fine-tuned with human feedback. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 5433–5442.

Katherine Tian, Eric Mitchell, Allan Zhou, Archit Sharma, Rafael Rafailov, Huaxiu Yao, Chelsea Finn, and Christopher D Manning. 2023b. Just ask for calibration: Strategies for eliciting calibrated confidence scores from language models fine-tuned with human feedback. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 5433–5442.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, Aurelien Rodriguez, Armand Joulin, Edouard Grave, and Guillaume Lample. 2023. Llama: Open and efficient foundation language models. *Preprint*, arXiv:2302.13971.

Muchao Ye, Chenglin Miao, Ting Wang, and Fenglong Ma. 2022. Texthoaxer: Budgeted hard-label adversarial attacks on text. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pages 3877–3884.

Zhen Yu, Zhenhua Chen, and Kun He. 2024. Query-efficient textual adversarial example generation for black-box attacks. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 556–569.

Guoyang Zeng, Fanchao Qi, Qianrui Zhou, Tingji Zhang, Zixian Ma, Bairu Hou, Yuan Zang, Zhiyuan Liu, and Maosong Sun. 2021. Openattack: An open-source textual adversarial attack toolkit. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International*

Joint Conference on Natural Language Processing: System Demonstrations, pages 363–371.

Xiang Zhang, Junbo Zhao, and Yann LeCun. 2015. Character-level convolutional networks for text classification. volume 28.

Appendix

Artifact use. We use publicly available datasets and model checkpoints following their intended research use and cite their original creators. The released code is intended for research and robustness evaluation only. Users should comply with the licenses and terms of use of the underlying datasets, models, and evaluation tools.

We used AI assistants only for language polishing, grammar checking, and formatting suggestions. All technical content, experimental design, analysis, and final decisions were made and verified by the authors.

A Algorithm of the BIRD

Algorithm 1 summarizes the full online loop.

B Additional Experimental Details

We provide additional results that are omitted from the main paper due to space limits. These include full per-setting single-operator results, per-setting efficiency diagnostics, per-setting static-composition results, and additional qualitative attack trajectories. These supplementary results further support the main findings that operator utility is heterogeneous, that static composition exposes unstable cost-quality trade-offs, and that BIRD can dynamically switch among lexical, contextual, and surface-form perturbation operators.

Full single-operator results. Table 7 reports the complete per-setting results for the four single-operator variants on LLaMA-3-8B-Instruct and Mistral-7B-Instruct-v0.3. These results complement the averaged single-operator analysis in the main paper and show that no operator consistently dominates across datasets, target models, and metrics.

Efficiency diagnostics. Table 8 reports per-setting successful-attack query cost and wall-clock execution time. Since wall-clock time is affected by hardware, batching, and model-serving conditions, we use target-model query cost as the primary efficiency metric in the main paper. The additional wall-clock results are provided only as a practical cost reference.

Static-composition diagnostics. Table 9 reports per-setting ASR for all two-operator and three-operator static compositions evaluated in our diagnostic study. These results show that static compositions can be strong, but the best manually fixed operator set varies across datasets and target models.

Additional qualitative trajectories. Table 10 provides additional attack trajectories. These examples illustrate that some attacks are solved by a single dominant operator, while others require switching among lexical, contextual, and surface-form perturbations.

C Additional Experimental Results

C.1 Full Single-Operator Results

Discussion. The full single-operator results show that operator behavior varies substantially across datasets and target models. InnerSwap is often the most attack-effective operator, especially on StrategyQA, but it also produces much higher PPL. MaskedLM tends to preserve fluency better, but its ASR is usually lower than $\text{CEAttack}_{\text{rep}}$ and InnerSwap. Homoglyph is query-efficient, but it is rarely sufficient as a standalone attack. These per-setting results support the main-paper conclusion that operator utility is heterogeneous and that committing the whole attack to one fixed operator is suboptimal.

C.2 Per-Setting Efficiency Diagnostics

Discussion. Table 8 reports the per-setting cost changes relative to $\text{CEAttack}_{\text{rep}}$. The cost pattern is model- and dataset-dependent. On several settings, such as LLaMA-3/SST-2, LLaMA-3/StrategyQA, Mistral-7B/StrategyQA, and DeepSeek-V4/StrategyQA, BIRD improves ASR while reducing successful-attack query cost. On AG-News, BIRD requires more successful-attack queries, suggesting that the larger label space can make adaptive exploration more expensive. Overall, these results support the main-paper conclusion that BIRD improves attack success with moderate additional cost rather than unbounded query expansion.

C.3 Static-Composition Diagnostics

Discussion. Table 9 provides the per-setting ASR behind the static-composition diagnostic. The results confirm that heterogeneous operators are useful, since many static compositions outperform the fixed $\text{CEAttack}_{\text{rep}}$ recipe. However, the strongest static composition varies across settings. For example, MaskedLM+InnerSwap gives the highest ASR on LLaMA-3/SST-2, LLaMA-3/StrategyQA, and Mistral-7B/SST-2, while CE+InnerSwap is strongest on LLaMA-3/AG-News and Mistral-7B/StrategyQA, and CE+MaskedLM is strongest

Algorithm 1 `BIRD`**: Bi-Level Bandit-Driven Operator Scheduling**

Input: target classifier f, input x_0, operators $\mathcal{A} = \{a_k\}_{k=1}^{K}$, budget B, window size W
Output: adversarial example $x^\star$ or failure

$t \leftarrow 0$; $x_t \leftarrow x_0$; $y \leftarrow f(x_0)$; $Q \leftarrow 0$; $\mathcal{H}_k \leftarrow \emptyset$ for all $a_k \in \mathcal{A}$
while $Q < B$ **do**
  $\mathcal{I}_t \leftarrow \mathrm{Index}(x_t)$
  if $\mathcal{I}_t = \emptyset$ **then**
    return failure
  end if
  $i_t \leftarrow \mathrm{SelectIndex}(\mathcal{I}_t)$; $a_t \leftarrow \arg\max_{a_k \in \mathcal{A}} U_k(t)$ ▷ upper-level scheduling
  $\mathcal{C}_t \leftarrow a_t(x_t, i_t)$; $\widetilde{\mathcal{C}}_t \leftarrow \mathrm{Filter}(\mathcal{C}_t)$ ▷ lower-level candidates
  $\widetilde{\mathcal{C}}_t \leftarrow \mathrm{Truncate}(\widetilde{\mathcal{C}}_t, B - Q)$
  if $\widetilde{\mathcal{C}}_t = \emptyset$ **then**
    $\mathcal{H}_{a_t} \leftarrow \mathrm{AppendWindow}(\mathcal{H}_{a_t}, 0, W)$; $t \leftarrow t + 1$; **continue**
  end if
  Query f on $\widetilde{\mathcal{C}}_t$; $\kappa_t \leftarrow |\widetilde{\mathcal{C}}_t|$; $Q \leftarrow Q + \kappa_t$
  $\mathcal{F}_t \leftarrow \{z \in \widetilde{\mathcal{C}}_t \mid f(z) \neq y\}$
  if $\mathcal{F}_t \neq \emptyset$ **then**
    $x_t^\star \leftarrow \arg\min_{z \in \mathcal{F}_t} c_f(z, y)$
    $x_{t+1} \leftarrow x_t^\star$
    $r_t \leftarrow R(x_t, x_{t+1}, a_t)$; $\mathcal{H}_{a_t} \leftarrow \mathrm{AppendWindow}(\mathcal{H}_{a_t}, r_t, W)$
    return $x_t^\star$
  end if
  $\tilde{x}_t \leftarrow \arg\min_{z \in \widetilde{\mathcal{C}}_t} c_f(z, y)$
  $x_{t+1} \leftarrow \begin{cases} \tilde{x}_t, & c_f(\tilde{x}_t, y) < c_f(x_t, y), \\ x_t, & \text{otherwise} \end{cases}$
  $r_t \leftarrow R(x_t, x_{t+1}, a_t)$; $\mathcal{H}_{a_t} \leftarrow \mathrm{AppendWindow}(\mathcal{H}_{a_t}, r_t, W)$; $x_t \leftarrow x_{t+1}$; $t \leftarrow t + 1$
end while
return failure

Model	Dataset	CEAttack$_{\text{rep}}$ ASR↑/SemSim↑/PPL↓/Avg.Q↑	MaskedLM ASR↑/SemSim↑/PPL↓/Avg.Q↑	InnerSwap ASR↑/SemSim↑/PPL↓/Avg.Q↑	Homoglyph ASR↑/SemSim↑/PPL↓/Avg.Q↑
LLaMA-3	SST-2	18.14 / 0.88 / 106.41 / 21.98	16.81 / 0.88 / 82.63 / 15.99	26.16 / 0.89 / 240.15 / 5.88	6.21 / 0.89 / 128.45 / 3.99
	AG-News	31.41 / 0.93 / 96.56 / 42.95	17.59 / 0.94 / 90.27 / 25.42	29.35 / 0.94 / 130.37 / 6.95	17.80 / 0.94 / 114.47 / 5.69
	StrategyQA	45.67 / 0.89 / 206.23 / 8.50	26.33 / 0.90 / 123.62 / 4.03	66.33 / 0.89 / 393.40 / 5.77	13.20 / 0.89 / 220.00 / 1.88
Mistral-7B	SST-2	19.81 / 0.88 / 97.31 / 23.14	12.62 / 0.89 / 81.47 / 15.87	22.91 / 0.89 / 218.42 / 5.92	4.30 / 0.89 / 133.78 / 4.17
	AG-News	44.49 / 0.92 / 97.76 / 43.41	30.26 / 0.94 / 78.05 / 28.03	28.07 / 0.94 / 148.49 / 6.92	19.38 / 0.94 / 109.26 / 5.68
	StrategyQA	39.26 / 0.90 / 177.94 / 8.71	19.50 / 0.90 / 116.43 / 4.18	55.46 / 0.89 / 386.16 / 0.89	10.79 / 0.90 / 234.97 / 1.93

Table 7: Full per-setting single-operator results on LLaMA-3-8B-Instruct and Mistral-7B-Instruct-v0.3. Each cell reports ASR / SemSim / PPL / Avg.Q. ASR and SemSim are higher better; PPL and Avg.Q are lower better.

Model	Dataset	ΔASR	ΔSucc.Q	ΔTime
LLaMA-3	SST-2	+12.02	**-4.46**	**-0:45:29**
	AG-News	+3.98	+13.82	+3:05:47
	StrategyQA	+18.88	**-1.50**	+0:02:40
Mistral-7B	SST-2	+7.98	+2.54	+0:22:27
	AG-News	+3.97	+13.93	+2:43:35
	StrategyQA	+19.91	**-1.16**	+0:01:53
DeepSeek-V4	SST-2	+1.00	**-30.73**	**-6:58:05**
	AG-News	+1.65	+13.42	+3:16:18
	StrategyQA	+10.60	**-1.51**	+0:03:30
Average	LLaMA/Mistral	+11.12	+3.87	+0:55:12
	All settings	+8.89	-6.27	+0:12:31

Table 8: Per-setting efficiency changes of `BIRD` over CEAttack$_{\text{rep}}$. ΔASR is the absolute ASR gain. ΔSucc.Q and ΔTime are computed as `BIRD` minus CEAttack$_{\text{rep}}$, so negative values indicate lower cost. Succ.Q is omitted when successful-attack query logs are incomplete.

on Mistral-7B/AG-News. This setting-dependent behavior illustrates why static composition is not a principled replacement for adaptive scheduling: it requires selecting an operator set before the attack, and this choice may not transfer across datasets and target models.

C.4 Additional Qualitative Trajectories

Discussion. The additional trajectories illustrate several recurring behaviors. First, some successful attacks are mostly lexical, indicating that CEAttack-style substitution remains useful in many cases. Second, several examples require a sequence of heterogeneous edits, such as surface-form perturbation followed by lexical substitution or contextual replacement. Third, some operators are useful only after previous edits have moved the input to a different attack state. These examples qualitatively support the trajectory-dependent operator utility assumed by `BIRD`.

Strategy	#Ops	LLaMA SST	LLaMA AG	LLaMA SQA	Mistral SST	Mistral AG	Mistral SQA	Avg.
$\text{CEAttack}_{\text{rep}}$	1	18.14	31.41	45.67	19.81	44.49	39.26	33.13
BIRD	adaptive	30.16	35.39	64.55	27.79	48.46	59.17	44.25
CE + MaskedLM	2	27.37	41.04	51.33	30.17	**57.14**	50.82	42.98
CE + InnerSwap	2	36.59	**43.51**	76.95	30.46	55.95	**76.95**	**53.40**
CE + Homoglyph	2	20.49	36.72	52.49	19.95	53.33	41.25	37.54
MaskedLM + InnerSwap	2	**40.35**	42.90	**79.28**	**32.86**	50.44	68.46	52.38
MaskedLM + Homoglyph	2	22.84	32.25	36.84	18.81	44.44	34.84	31.67
InnerSwap + Homoglyph	2	30.22	39.22	71.48	25.06	40.97	64.63	45.26
CE + MaskedLM + InnerSwap	3	31.70	44.01	72.43	28.64	51.11	68.33	49.37
CE + MaskedLM + Homoglyph	3	23.95	37.50	49.17	21.56	53.74	42.86	38.13
CE + InnerSwap + Homoglyph	3	27.78	40.00	71.52	24.11	47.79	64.71	45.99
MaskedLM + InnerSwap + Homoglyph	3	29.05	38.76	68.98	26.90	45.58	66.26	45.92

Table 9: Per-setting ASR of adaptive scheduling and static operator compositions over six settings with LLaMA-3-8B-Instruct and Mistral-7B-Instruct-v0.3. Static combinations can achieve high ASR, but the best manually specified operator set varies across settings.

Target / Dataset	Original fragment	Operator trajectory	Key edits	Outcome
LLaMA-3 / SST-2	*sade is an engaging look ... fiercely atheistic hero*	Homoglyph → CEAttack → CEAttack	look → [hg]look; engaging → participate; fiercely → vehemently	prediction flips to negative
LLaMA-3 / SST-2	*the primitive force of this film ... collective memory ...*	InnerSwap → CEAttack → MaskedLM → InnerSwap	reorder phrase; vast → gargantuan; seems → seem; reorder phrase	prediction flips to negative
LLaMA-3 / SST-2	*about a manga-like heroine ... energetic and satisfying ...*	CEAttack → InnerSwap → MaskedLM → CEAttack	energetic → forceful; reorder psychological/satisfying; manga-like → tormented; deep → profound	prediction flips to negative
LLaMA-3 / SST-2	*something akin to a Japanese Alice ... far more seriously*	CEAttack → CEAttack → CEAttack	akin → equivalent; take → pick; seriously → profoundly	prediction flips to negative
LLaMA-3 / SST-2	*nicks, seemingly uncertain ... runs the gamut ...*	MaskedLM → MaskedLM → MaskedLM	runs → run; people → men; seemingly → apparently	prediction flips to positive
DeepSeek-V4 / SST-2	*the acting, costumes, music, cinematography and sound ...*	InnerSwap → Homoglyph → CEAttack → CEAttack	reorder acting/given; astounding → [hg]astounding; austere → stiff; sound → audio	prediction flips to negative

Table 10: Additional qualitative attack trajectories. The table summarizes representative successful cases showing different operator-switching patterns. Only the key edits are shown for readability; full perturbed texts are omitted.

D Limitations of Current Evaluation

The current evaluation focuses on black-box classification attacks with elicited confidence feedback. Although we evaluate multiple datasets and target models, the experiments do not exhaust all possible LLM tasks or downstream safety systems. In addition, after-attack PPL increases in some settings, especially when stronger surface-form perturbations are selected. Future work can further evaluate human-perceived fluency, label preservation with human annotation, query-budget curves under different budgets, and sensitivity to scheduler hyperparameters such as the UCB exploration coefficient and sliding-window size. These analyses would provide a more complete picture of the cost-quality frontier of adaptive operator scheduling.

E Prompts

Figure 3 shows the prompt we use to first perform a prediction on x='text' and then elicit confidence following CEAttack (Formento et al., 2025). This prompt is a combination of methods from (Lin et al.), where verbal confidence is utilized, and (Tian et al., 2023b), where a two-shot approach is used for confidence elicitation.

F Further Implementation Details

We use 8 NVIDIA A100-80GB GPUs for our testing, every test can be conducted on only 1 A100GPU. For our tests we perturb 500 samples on 1 A100 GPU with 80GB of memory.

G Experimental Setup Details

G.1 Datasets, tasks and models

We conducted our confidence elicitation attacks on Meta-Llama-3-8B-Instruct (Touvron et al., 2023) and Mistral-7B-Instruct-v0.2 (Jiang et al., 2023) while performing classification on two common datasets to evaluate adversarial robustness: *SST-2* (Socher et al., 2013), *AG-News* (Zhang et al., 2015) and one modern dataset: *StrategyQA* (Geva et al., 2021).

G.2 Evaluation metrics

We utilize the evaluation framework previously proposed in (Morris et al., 2020), where an evaluation set is perturbed, and we record the following data from the Total Attacked Samples (TAS) set: Number of Successful Attacks ($N_{succ-atk}$), Num-

Prediction Prompt

```
f"""{self.start_prompt_header}
Provide your k best guess for the following text (positive, negative).
Give ONLY the guesses, no other words or explanation.
For example:
Guesses: (most likely guesses, either positive or negative; not a complete sentence, just the
↪  guesses!
Separated by a comma, for example [Negative, Positive, Positive, Negative ... xk])


The text is:{text}
Guesses:
{self.end_prompt_footer}"""
```

Confidence Elicitation Prompt

```
f"""{self.start_prompt_header}
You're a model that needs to give the confidence of answers being correct.
The previous prompt was:
Provide your k best guesses for the following text (positive, negative).
Give ONLY the guesses, no other words or explanation.
For example:


Guesses: (most likely guess, either positive or negative; not a complete sentence, just the
↪  guesses!)
The text is:{text}
the guesses were: {guesses_output},
given these guesses provide the verbal confidences that your guesses are correct.
Give ONLY the verbal confidences, no other words or explanation.
For example:
Confidences: (the confidences, from either (Highest, High, Medium, Low, Lowest) that your
↪  guesses are correct,


without any extra commentary whatsoever, for example [Highest, High, Medium, Low, Lowest ...];
just the confidence! Separated by a comma
Confidences:
{self.end_prompt_footer}"""
```

Figure 3: An example of the Verb. 2S top-k prompting technique. The first prompt generates answers, $\hat{y}$, through the model f_θ. These answers are then passed to the Confidence Elicitation Prompt as the variable `guesses_output`. The second prompt is passed through f_θ to generate the verbal confidence, $\mathbf{p}_\mathcal{C}$. The variable `text` is the sample under analysis, while `start_prompt_header` and `end_prompt_footer` are the model's formatting tokens. Here we strictly adhere to the setup described in CEAttack (Formento et al., 2025).

ber of Failed Attacks ($N_{fail-atk}$), and Number of Skipped Attacks ($N_{skp-atk}$). We utilize these values to record the following metrics. *Clean accuracy/Base accuracy/Original accuracy*, which offers a measure of the model's performance during normal inference. *After attack accuracy/Accuracy under attack* ($A_{aft-atk} = \frac{N_{fail-atk}}{TAS}$) or (AUA), is critical, representing how effectively the attacker deceives the model across the dataset. Similarly, the *After success rate* ($A_{succ-rte} = \frac{N_{succ-atk}}{TAS - N_{skp-atk}}$) or (ASR) excludes previously misclassified samples. The paper also considers the *Semantic similarity/SemSim*, an automatic similarity index, as modeled by d_ϵ (Cer et al., 2018). We compare the original perplexity with the new perturbed sample's perplexity, calculated using a GPT-2 model. A higher perplexity indicates that the example is less natural and fluent to the language model. *Queries* denotes the number of model calls for inference. We subdivide this metric into two categories: *All Att Queries Avg* and *Succ Att Queries Avg*. The latter records the queries for successful attacks only, while the former includes all queries. Additionally, we track the duration of the attack process to perturb all the samples under *Total Attack Time*.