Published by :
https://www.ijert.org/
An International Peer-Reviewed Journal

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 09 , September - 2026

Deployment-Oriented Evaluation of Lightweight SMS Spam Filters: False-Alarm Guarantees, Adaptive Evasion and Shortcut Side-Effects

Rounak Panda
Independent Researcher
B.Tech, Computer Science and Engineering, Techno Main Salt Lake
Maulana Abul Kalam Azad University of Technology (MAKAUT), West Bengal
Kolkata, West Bengal, India

Abstract **- SMS spam filters are among the most widely deployed consumer machine-learning systems, yet they are still judged by accuracy on a random split of a public corpus. We evaluate five lightweight linear filters on the de-duplicated SMS Spam Collection under an eight-stage, deployment-oriented protocol: template-grouped splits, prevalence-adjusted metrics, three operating-point rules, random and adaptive character-level attacks, a benign-message stress test, full-precision, int8 and hashed deployment variants, and paired Wilcoxon tests with Holm correction over 35 hypotheses. Benchmark accuracy (98.0–99.0%) hides a tenfold range in false-positive rate (FPR). A naive empirical-quantile threshold exceeds its FPR target in 50–80% of splits, whereas a Neyman–Pearson umbrella threshold holds violations to 0–10% and shows that about 633 validation messages cannot support any FPR target below 0.5%. A greedy black-box attacker with eight character edits evades 98.5% of detected spam for word-level logistic regression but 24.0% for a character n-gram SVM. Perturbation augmentation improves robustness yet teaches a shortcut: it raises FPR on benignly stylised genuine messages up to fourfold for the word-level model (0.60% to 2.45%; adjusted p = 0.003), while appended Devanagari, Bengali or Tamil text and emoji leave every model unchanged. A vocabulary-free hashed character model needs 16 KB in int8 at 90.6% recall and 0.06% FPR, against 300 KB for its vocabulary-based counterpart. We release the evaluation code and propose a six-item reporting checklist for consumer text filters.**

Index Terms **- SMS spam, smishing, Neyman–Pearson classification, adversarial robustness, shortcut learning, feature hashing, quantization, evaluation methodology**

I. INTRODUCTION

Text-message filtering is one of the most frequent points of contact between machine learning and everyday life. In India, fraudulent SMS impersonating banks, KYC services, couriers and utilities are common enough that the Department of Telecommunications runs a dedicated citizen reporting facility, Chakshu, on the Sanchar Saathi portal [1]. A filter that intercepts such a message protects the user at the moment of exposure; a filter that suppresses a genuine one-time password or bank alert causes direct harm. The two errors are asymmetric, and the second is the one users notice.

Published evaluation practice does not reflect this asymmetry. Studies on public corpora [2], [3], [4] usually report accuracy or F-measure on a random split, at the corpus prevalence, against unmodified text, with no statement about model size. Each of these choices hides a property that decides whether a filter is fit for a phone. This paper makes those properties measurable and asks seven research questions:

- **RQ1** Does near-duplicate template leakage inflate recall?
- **RQ2** What precision does one user see at realistic prevalence?
- **RQ3** Can a deployer *guarantee* a false-alarm rate, and with how much data?
- **RQ4** How far does recall fall under random and adaptive character-level attacks?
- **RQ5** Do normalisation and augmentation defences generalise to unseen attacks?
- **RQ6** Do those defences change behaviour on genuine messages?
- **RQ7** How small can a filter be without losing accuracy?

The contributions are: (i) an eight-stage evaluation protocol (Fig. 1) that combines leakage control, prevalence adjustment, finite-sample FPR control, graded and adaptive attacks, a benign-message stress test and multiplicity-corrected paired testing; (ii) an application of Neyman–Pearson umbrella thresholds [5] to SMS filtering, with the sample sizes it implies; (iii) evidence that augmentation-based robustness is partly a shortcut [6] that penalises stylised genuine messages; (iv) a 16 KB vocabulary-free model with near-baseline accuracy; and (v) a reporting checklist. No new classifier is proposed; the aim is to show which conclusions change when evaluation reflects deployment.

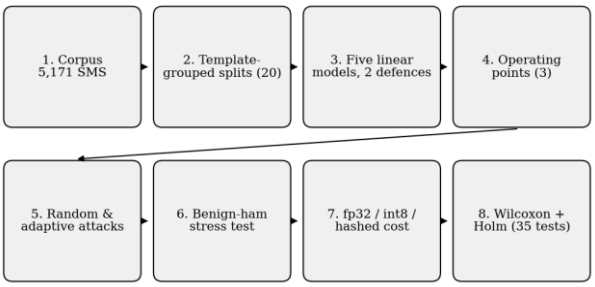

Fig. 1. The evaluation protocol. Stages 4–7 are the deployment-oriented additions studied in this paper.

II. RELATED WORK

SMS filtering. Gómez Hidalgo et al. [4] transferred content-based e-mail filtering to SMS; Almeida et al. [2] released the SMS Spam Collection and found linear SVMs and Naive Bayes

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 09 , September - 2026

strong on it; Delany et al. [3] identified duplicated messages as a source of optimistic estimates. Online linear SVMs remain competitive for spam [7].

Evaluation and error control. Kapoor and Narayanan [8] show that train–test leakage is a widespread cause of irreproducible results. Axelsson [9] showed that rare positives turn small FPRs into mostly false alarms, and Saito and Rehmsmeier [10] recommend precision–recall analysis under imbalance. Neyman–Pearson (NP) classification minimises the type-II error subject to a type-I constraint [11]; the umbrella algorithm of Tong et al. [5] attains that constraint with high probability for any scoring classifier, using an order statistic of held-out negative scores.

Adversarial text. Dalvi et al. [12] framed spam filtering as a game with an adaptive adversary. HotFlip [13] searches character flips with gradient access; DeepWordBug [14] and TextBugger [15] do so with queries only. Pruthi et al. [16] defend with word recognition. Unicode documents confusable characters [17] and normalisation forms [18]. Adversarial and augmentation-based training [19], [20] is a standard defence, and shortcut learning [6] describes models that succeed on a benchmark for the wrong reason.

Efficient inference. Linear n-gram models are compact and strong [21]; feature hashing removes the vocabulary [22]; integer quantisation reduces weight storage [23].

III. PROBLEM SETTING AND THREAT MODEL

A filter on the handset or at the operator scores each incoming message and diverts those above a threshold. Its cost to the user is the number of genuine messages diverted (false alarms) and spam messages delivered (misses) per unit time; the first is typically the more expensive. We consider two adversaries. The *random* adversary applies readability-preserving character edits at rate q with no access to the filter. The *adaptive* adversary may query the filter's score, as a fraudster can by testing messages against a purchased handset, and greedily chooses up to B character edits. Semantic paraphrase, URL-level evasion and white-box gradient attacks are out of scope.

IV. EXPERIMENTAL DESIGN

A. Corpus and template grouping

We use the SMS Spam Collection v.1 [2] (5,574 English messages). Removing exact duplicates leaves 5,171, of which 653 (12.6%) are spam. A template key (lower-casing, URL token, digit runs collapsed to 0, punctuation stripped) maps these to 5,098 templates; the 653 spam messages contain 56 template-level near-duplicates.

B. Classifiers and splits

Five scikit-learn [24] pipelines with fixed, untuned hyper-parameters: **NB-word** (word counts, multinomial Naive Bayes); **LR-word** (sublinear word TF-IDF, logistic regression, C = 10); **SVM-word** (same features, linear SVM, C = 1); **SVM-char** (character 2–5-grams within word boundaries, min_df = 2, linear SVM, C = 1); and **SVM-w+c** (both feature sets concatenated). The *random* protocol draws 20 stratified 70/30 splits. The *grouped* protocol, used for all results unless stated, takes three of ten StratifiedGroupKFold folds over template keys as the test set (1,551–1,552 messages, 195–197 spam, about 1,355 ham) for each of 20 seeds.

C. Operating points

Besides the classifier's *default* boundary, two data-driven thresholds are fitted on a stratified 20% validation split of the training set (632–633 ham messages), with the model refit on the remaining 80%. The *empirical-quantile* rule sets the threshold at the $(1 - \alpha)$ quantile of validation ham scores. The *NP umbrella* rule [5] sets it at the k-th smallest of the n validation ham scores, with k the smallest integer satisfying

$$\sum_{j=k}^{n} C(n, j)\,(1-\alpha)^{j}\,\alpha^{n-j} \le \delta,$$

which guarantees $P(\mathrm{FPR} > \alpha) \le \delta$ for any score function; we use $\delta = 0.05$. A valid k exists only if $n \ge \ln \delta / \ln(1 - \alpha)$.

D. Prevalence-adjusted metrics

For prevalence p, precision is $\mathrm{PPV} = \mathrm{TPR}\cdot p / [\mathrm{TPR}\cdot p + \mathrm{FPR}\cdot(1 - p)]$, computed per split with 95% intervals from the 2.5th and 97.5th percentiles. For an illustrative user receiving $N = 1{,}200$ messages a month, false alarms and misses per month are $N(1 - p)\mathrm{FPR}$ and $Np(1 - \mathrm{TPR})$.

E. Random perturbations

Applied to test spam only at rate q: *leet* (o, i, e, a, s → 0, 1, 3, @, $), *split* (a dot inside words longer than three letters), *homoglyph* (Cyrillic look-alikes for a, e, o, p, c, x) and *combined* (all three; $q \in \{0.1, 0.25, 0.5, 0.75, 1.0\}$). The *held-out* attack, written after the defences were fixed, inserts zero-width spaces, hyphens or underscores and substitutes Greek omicron and capital alpha ($q = 0.5$).

F. Adaptive attack

For each initially detected spam message, the attacker enumerates every single-character edit from a substitution table (12 letters, one to three visual or numeric substitutes each, including Cyrillic, Greek and Ukrainian look-alikes) plus a zero-width space inside any word, queries the score of every candidate and keeps the lowest-scoring one. This repeats until the message is classified as ham or $B = 8$ edits are used. The procedure is a black-box greedy search in the spirit of [14], [15]. It is run on 80 randomly chosen test spam messages from each of five grouped splits (400 messages; median length 149 characters). Evasion rates carry Wilson 95% intervals [25].

G. Defences and benign stress test

Normaliser: NFKC [18], reverse mapping of the six Cyrillic homoglyphs and the leet table inside alphabetic tokens, and removal of dots between letters, applied to training and test text. *Augmentation*: each training spam message is added three more times, under leet, split and homoglyph at $q = 0.5$. To probe side-effects, every test ham message is also rewritten three ways: *script* appends a common Devanagari, Bengali or Tamil phrase; *emoji* appends one to three emoji; *casual* hyphenates or underscores 15% of longer words and applies leet at $q = 0.1$, imitating informal texting.

H. Deployment variants

Size counts the UTF-8 vocabulary plus float32 weights and IDF vector; the int8 variant quantises weights symmetrically. The *hashed* variant replaces the vocabulary with a 2^b-bucket character hash [22] ($b \in \{10, \ldots, 18\}$), omits IDF so that no per-feature table other than the int8 weights is stored, and uses sublinear TF with L2 normalisation. Latency is the mean single-

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 09 , September - 2026

message time over 1,000 messages on one Intel Xeon core (2.8 GHz), Python 3.11, scikit-learn 1.8.0.

I. Statistical analysis

Paired comparisons over the 20 splits use the two-sided Wilcoxon signed-rank test [26], [27]. All 35 hypothesis tests reported in this paper form one family and are corrected with the Holm procedure [28]; we write adj. p for adjusted values. With 20 pairs the smallest attainable raw p is about 9×10^{-5}, so adj. $p \geq 0.003$.

V. RESULTS

A. RQ1: Clean performance and leakage

Table I shows that accuracy spans one point while FPR spans a factor of ten. Under random splits 11.8% of test spam (range 9.7–14.3%) shares a template with training data, yet recall differs from the grouped protocol by at most 0.63 points and no difference is significant (all adj. $p = 1$). After exact de-duplication, template leakage is not a material bias on this corpus.

TABLE I. CLEAN PERFORMANCE, GROUPED SPLITS (%, MEAN ± SD, n = 20)

Model	Accuracy	Recall	FPR	AP	Recall (random)
NB-word	98.31±0.26	89.42±2.14	0.40±0.14	95.90	89.85
LR-word	97.97±0.38	85.69±2.62	0.26±0.12	97.33	86.33
SVM-word	98.34±0.37	88.65±2.75	0.26±0.11	97.61	88.95
SVM-char	98.95±0.27	91.99±2.14	0.04±0.06	98.77	91.68
SVM-w+c	98.96±0.26	92.32±2.07	0.08±0.07	98.73	92.07

AP: average precision. Random-versus-grouped recall differences: all raw p ≥ 0.40.

B. RQ2: Precision at realistic prevalence

Table II translates Table I to a single inbox. At 1% prevalence, one in four messages diverted by a word-level model is genuine (precision 69.9–78.2%), and the illustrative user loses 3.1–4.8 genuine messages a month. SVM-char keeps 96.2% precision and loses 0.48.

TABLE II. SINGLE-USER VIEW, DEFAULT THRESHOLD (N = 1,200 SMS/MONTH)

Model	PPV @1% [95% int.]	PPV @2%	False alarms/mo	Missed/mo @2%
NB-word	69.9 [56.8, 80.7]	82.3	4.78	2.54
LR-word	77.7 [64.1, 85.9]	87.4	3.07	3.43
SVM-word	78.2 [65.1, 86.3]	87.7	3.07	2.72
SVM-char	96.2 [83.6, 100]	98.0	0.48	1.92
SVM-w+c	92.7 [80.9, 100]	96.2	0.92	1.84

False alarms per month at 1% prevalence. One false positive among ~1,355 test ham messages moves FPR by 0.074 points, which explains the wide intervals.

C. RQ3: Guaranteed false-alarm control

Table III and Fig. 2 compare the two data-driven rules. The empirical-quantile rule, the natural choice for a practitioner, exceeds its target in 65–80% of splits at $\alpha = 0.5\%$ and 50–70% at $\alpha = 1\%$, with mean test FPR above target for every model. The NP umbrella rule keeps mean test FPR well below target and violates it in 0–10% of splits. Measured violations slightly above $\delta = 5\%$ are expected, because test FPR is itself estimated from about 1,355 messages and so carries sampling noise.

TABLE III. DATA-DRIVEN THRESHOLDS (δ = 0.05; %, MEAN OVER 20 SPLITS)

Model	α	Quantile FPR / viol.	NP FPR / viol.	NP recall
NB-word	0.5	0.82 / 80	0.10 / 0	84.98
LR-word	0.5	0.61 / 70	0.13 / 5	74.80
SVM-word	0.5	0.59 / 65	0.12 / 10	79.83
SVM-char	0.5	0.72 / 65	0.17 / 5	92.73
SVM-w+c	0.5	0.70 / 65	0.17 / 10	91.31
NB-word	1.0	1.25 / 70	0.42 / 0	88.55
LR-word	1.0	1.11 / 50	0.35 / 0	85.57
SVM-word	1.0	1.08 / 55	0.41 / 0	89.34
SVM-char	1.0	1.22 / 55	0.45 / 10	95.46
SVM-w+c	1.0	1.21 / 55	0.45 / 5	95.23

viol.: percentage of splits whose test FPR exceeds α. Validation sets hold 632–633 ham messages.

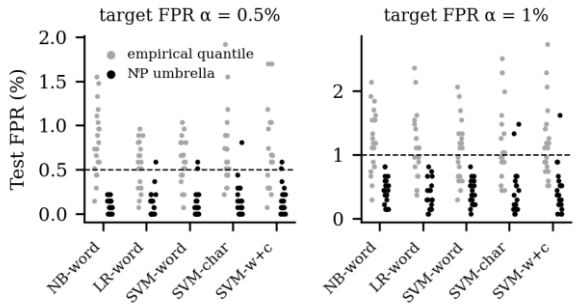

Fig. 2. Per-split test FPR under the empirical-quantile rule (grey) and the NP umbrella rule (black). Dashed line: target α.

The sample-size condition is the central practical result. With $\delta = 0.05$, a valid NP threshold requires at least 299 validation ham messages for $\alpha = 1\%$, 598 for 0.5%, 1,497 for 0.2% and 2,995 for 0.1%. The 633 messages available here, typical of a 20% validation split on public SMS corpora, therefore cannot support any guarantee tighter than 0.5%; at $\alpha = 0.5\%$ the umbrella threshold is the maximum validation score. A deployer who wants one false alarm per 1,000 messages needs 2,995 validation ham messages; with the split proportions used here that implies about 21,000 genuine messages in total, 4.7 times the ham in the whole corpus. Under the guarantee, SVM-char keeps 92.7–95.5% recall while LR-word falls to 74.8% at $\alpha = 0.5\%$.

D. RQ4: Random and adaptive attacks

Random attacks. Fig. 3 plots recall against the rate of the combined attack. At $q = 0.5$, LR-word falls from 85.7% to 30.9% and SVM-word from 88.7% to 39.2%; at $q = 1.0$ they catch 12.9% and 19.0%. NB-word declines more gently (72.0% at $q = 0.5$). SVM-char loses 5.0 points at $q = 0.5$ and 7.4 at $q = 1.0$, while adding word features (SVM-w+c) brings back fragility (70.6% at $q = 1.0$). Against the held-out attack, SVM-char keeps 89.4% recall, 21.2 points above LR-word (adj. $p = 0.003$).

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 09 , September - 2026

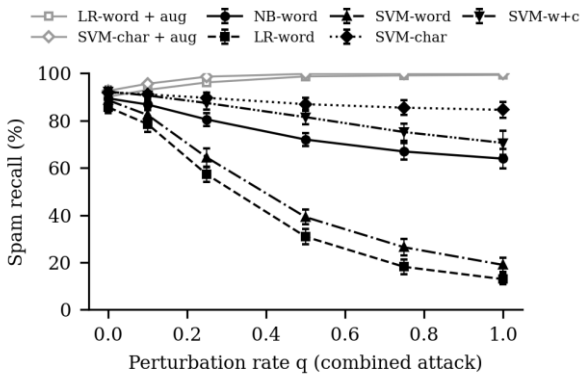

Fig. 3. Spam recall under the combined random attack versus rate q (grouped splits, ±1 SD). Grey: models trained with augmentation.

Adaptive attack. Table IV and Fig. 4 report the greedy black-box attacker. Word-level discriminative models are almost completely evadable: eight edits, about 5% of a typical message, defeat 98.5% of detected spam for LR-word and 96.5% for SVM-word, with a median of three to four edits and 600–700 queries. SVM-char limits evasion to 24.0% [19.9, 28.7], and its curve flattens after four edits: this suggests that the messages it loses lie close to the decision boundary, while the rest resist the budget. Augmentation reduces evasion of LR-word to 68.1% but does not close the gap to an undefended character model, and gives SVM-char no significant further gain (20.6% [16.7, 25.0]; intervals overlap).

TABLE IV. ADAPTIVE BLACK-BOX ATTACK (5 SPLITS, 400 MESSAGES)

Model	Det.	B≤1	B≤2	B≤4	B≤8 [95% CI]	Med. q
NB-word	343	7.0	15.7	30.9	60.6 [55.4, 65.7]	722
LR-word	334	15.3	31.7	71.3	98.5 [96.5, 99.4]	602
SVM-word	339	13.0	25.1	58.7	96.5 [93.9, 98.0]	703
SVM-char	354	7.1	10.5	16.1	24.0 [19.9, 28.7]	373
SVM-w+c	354	5.9	13.6	22.0	39.8 [34.9, 45.0]	694
LR-word+aug	339	8.3	15.9	32.7	68.1 [63.0, 72.9]	816
SVM-char+aug	360	6.4	9.2	13.9	20.6 [16.7, 25.0]	403
SVM-char+norm	352	6.2	10.8	16.2	22.4 [18.4, 27.1]	393

Det.: spam messages detected before the attack. B≤k: evasion rate (%) within k edits. Med. q: median queries for successful evasions.

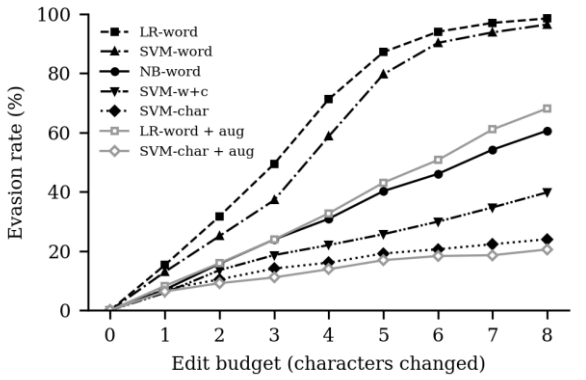

Fig. 4. Cumulative evasion rate of the greedy black-box attacker against edit budget.

E. RQ5: Do defences generalise?

Table V summarises the defences under random attacks. The normaliser restores word-model recall on the combined attack it encodes (LR-word 30.9% → 85.5%, adj. $p = 0.003$) and would appear sufficient if only encoded attacks were tested. On the held-out attack it significantly *lowers* recall for LR-word (−3.3 points), SVM-word (−2.4) and SVM-w+c (−1.0) (adj. $p \leq 0.015$); the −0.4 change for SVM-char is not significant after correction (adj. $p = 0.46$). Augmentation raises held-out recall by 20.3, 13.8, 3.4 and 2.1 points for the four models (all adj. $p = 0.003$). It also raises clean FPR, significantly for LR-word (0.26% → 0.60%) and SVM-word (0.26% → 0.45%) (adj. $p \leq 0.003$); the +0.04-point increases for the character models do not survive correction (adj. $p = 0.10$ and 0.49).

TABLE V. DEFENCES UNDER RANDOM ATTACKS (%, MEAN OVER 20 SPLITS)

Model	Defence	Recall	FPR	Comb. q=0.5	Held-out
LR-word	none	85.7	0.26	30.9	68.2
	norm.	85.4	0.31	85.5	64.9
	aug.	90.2	0.60	98.7	88.5
SVM-word	none	88.7	0.26	39.2	75.3
	norm.	87.9	0.29	88.0	72.9
	aug.	90.1	0.45	98.8	89.1
SVM-char	none	92.0	0.04	87.0	89.4
	norm.	91.8	0.04	92.0	89.0
	aug.	92.7	0.08	99.8	91.5
SVM-w+c	none	92.3	0.08	81.5	88.3
	norm.	92.1	0.09	92.4	87.3
	aug.	92.8	0.11	99.7	91.7

F. RQ6: Side-effects on genuine messages

Recall under the combined attack *exceeds* clean recall for augmented models (up to 99.8%), which suggests that they partly learn obfuscation itself as evidence of spam. Table VI tests this directly on rewritten genuine messages. Undefended and normalised models are essentially unaffected by all three rewrites (changes of at most 0.07 points, none significant after correction). Augmented models are not: casual stylisation raises FPR from 0.60% to 2.45% for LR-word, from 0.08% to 0.23% for SVM-char and from 0.11% to 0.42% for SVM-w+c (adj. $p \leq 0.005$ for all three). For the illustrative user whose genuine messages were all written in this style, augmented LR-word would divert about 29 of them a month. This is a shortcut in the sense of [6]: the robustness gain is bought partly with a style prior that penalises informal writers.

TABLE VI. FPR (%) ON REWRITTEN GENUINE MESSAGES

Model	Defence	Clean	Script	Emoji	Casual
NB-word	none	0.40	0.40	0.40	0.47
LR-word	none	0.26	0.26	0.26	0.27
	aug.	0.60	0.60	0.60	2.45
SVM-char	none	0.04	0.04	0.04	0.05
	aug.	0.08	0.08	0.08	0.23
SVM-w+c	none	0.08	0.08	0.08	0.08
	aug.	0.11	0.11	0.11	0.42

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 09 , September - 2026

Normalised models (not shown) change by at most 0.05 points (no change significant after correction).

The script and emoji rewrites change nothing because vocabulary-based features silently discard symbols unseen in training. This rules out one feared failure, the penalising of code-mixed Indian messages, for these models. It also exposes the converse risk: a scam written mostly in an Indic script would be scored on its Latin fragments alone. Both effects are properties of the feature extractor, not of the classifier, and they need corpora with real code-mixed traffic to quantify.

G. RQ7: Deployment cost

Table VII lists cost. Because the vocabulary dominates model size, int8 quantisation saves only 20–23% and changes none of the 1,552 test predictions. Feature hashing removes the vocabulary altogether. At 2^{14} buckets the hashed model occupies 16 KB in int8, keeps 90.6% recall and 0.06% FPR (1.4 and 0.02 points from SVM-char), and agrees with its float32 version on 99.99% of predictions. Its held-out recall is lower (83.7% against 89.4%). Because the hashed variant also omits IDF, we cannot attribute this loss to hashing alone. Latency is below 1.4 ms per message for every model on a server core.

TABLE VII. DEPLOYMENT COST (GROUPED SPLITS)

Model	Size fp32 / int8 (KB)	Recall	FPR	Held-out	ms/msg
NB-word	105 / —	89.4	0.40	84.7	0.18
LR-word	105 / 84	85.7	0.26	68.2	0.35
SVM-word	105 / 84	88.7	0.26	75.3	0.35
SVM-char	389 / 300	92.0	0.04	89.4	0.56
SVM-w+c	495 / 384	92.3	0.08	88.3	1.35
Hash 2^{10}	— / 1	88.2	0.20	85.1	—
Hash 2^{12}	— / 4	90.3	0.09	85.4	—
Hash 2^{14}	— / 16	90.6	0.06	83.7	—
Hash 2^{16}	— / 64	90.7	0.05	81.8	—
Hash 2^{18}	— / 256	90.5	0.05	81.5	—

Hashed rows report int8 predictions; int8/float32 agreement is $\geq$ 99.98% for every bucket size. Latency was measured for the vocabulary models only.

VI. DISCUSSION

Accuracy is a weak proxy, but rankings are stable. A one-point accuracy range conceals a tenfold FPR range, a fourfold difference in adaptive evasion and a sixfold difference in recall under full-rate random attack. Across all of these, SVM-char is first or statistically tied for first; the order among models rarely changes, but their absolute fitness for deployment does.

Guarantees are a data problem. The NP analysis turns an intuition into a number: the false-alarm rates users tolerate need thousands of labelled genuine messages for validation, more than public SMS corpora contain. Operators hold such data; academic studies should state the n behind any operating point and prefer finite-sample rules to empirical quantiles.

Robustness should come from representation before training tricks. Character n-grams deliver most of the robustness that augmentation buys for word models, without its shortcut. Augmentation helps, but it must be audited on stylised genuine messages, and rule-based normalisers must be tested on attacks they do not encode.

Small can be good enough. A 16 KB hashed model is competitive on clean data and could fit on feature phones or in a SIM-toolkit applet. Its lower robustness shows that compression must be evaluated on the same robustness axes as accuracy.

Reporting checklist. We recommend that consumer text-filter studies report: (1) precision and FPR at prevalences below 5%, with intervals; (2) the validation size and rule behind any operating point, preferably with a finite-sample guarantee; (3) recall under a graded random attack including an attack withheld from defence design; (4) evasion under a budgeted adaptive attacker; (5) FPR on benignly stylised genuine messages whenever a defence is used; and (6) compressed size and single-message latency.

VII. THREATS TO VALIDITY

- *External.* The corpus is English, mainly from the UK and Singapore, and over a decade old [2]. Indian smishing uses code-mixed text, Indic scripts, shortened URLs and brand impersonation. Recent smishing corpora [29] contain phishing only and cannot support FPR estimates. Absolute numbers should not be transferred; the methodological conclusions are more likely to hold.
- *Temporal.* Splits are not time-ordered, so lure drift is not measured.
- *Adversary.* The greedy attacker is limited to single-character edits with a fixed table and eight edits; stronger search or white-box access [13] would evade more. Five splits give interval estimates but no paired tests for the attack.
- *Construct.* The benign rewrites are synthetic approximations of informal writing. The monthly volume is illustrative. Latency was measured on a server core.
- *Scope.* Only linear models were studied; compact transformers may behave differently and cost more to run on a phone.

VIII. CONCLUSION

Five SMS filters with near-identical benchmark accuracy differ sharply once evaluation reflects deployment. Empirical thresholds routinely break their false-alarm targets, and a finite-sample NP rule shows that public-corpus validation sets cannot certify FPRs below 0.5%. A query-only greedy attacker with eight edits defeats word-level discriminative filters almost completely but a character n-gram SVM in fewer than one case in four. Augmentation improves robustness at the price of a measurable bias against stylised genuine messages. A 16 KB hashed character model is a credible starting point for very constrained devices. Future work will build a time-stamped, code-mixed Indian SMS corpus with genuine traffic, evaluate compact transformers under the same protocol, and study certified robustness for character n-gram models.

DATA AND CODE AVAILABILITY

All experiments use the public SMS Spam Collection. Three Python scripts (pandas, NumPy, SciPy, scikit-learn 1.8.0, joblib) reproduce every table and figure in about twelve minutes on two CPU cores. They are available from the author on request.

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 09 , September - 2026

REFERENCES

[1] Press Information Bureau, Government of India, "Chakshu facility of Sanchar Saathi enables citizens to report suspected fraud communications under various categories," press release, Feb. 2026. [Online]. Available: https://www.pib.gov.in/PressReleasePage.aspx?PRID=2223779

[2] T. A. Almeida, J. M. Gómez Hidalgo, and A. Yamakami, "Contributions to the study of SMS spam filtering: New collection and results," in Proc. 11th ACM Symp. Document Engineering (DocEng), 2011, pp. 259–262.

[3] S. J. Delany, M. Buckley, and D. Greene, "SMS spam filtering: Methods and data," Expert Systems with Applications, vol. 39, no. 10, pp. 9899–9908, 2012.

[4] J. M. Gómez Hidalgo, G. C. Bringas, E. P. Sánz, and F. C. García, "Content based SMS spam filtering," in Proc. ACM Symp. Document Engineering (DocEng), 2006, pp. 107–114.

[5] X. Tong, Y. Feng, and J. J. Li, "Neyman-Pearson classification algorithms and NP receiver operating characteristics," Science Advances, vol. 4, no. 2, eaao1659, 2018.

[6] R. Geirhos, J.-H. Jacobsen, C. Michaelis, R. Zemel, W. Brendel, M. Bethge, and F. A. Wichmann, "Shortcut learning in deep neural networks," Nature Machine Intelligence, vol. 2, pp. 665–673, 2020.

[7] D. Sculley and G. M. Wachman, "Relaxed online SVMs for spam filtering," in Proc. 30th Annu. Int. ACM SIGIR Conf. Research and Development in Information Retrieval, 2007, pp. 415–422.

[8] S. Kapoor and A. Narayanan, "Leakage and the reproducibility crisis in machine-learning-based science," Patterns, vol. 4, no. 9, 100804, 2023.

[9] S. Axelsson, "The base-rate fallacy and the difficulty of intrusion detection," ACM Trans. Information and System Security, vol. 3, no. 3, pp. 186–205, 2000.

[10] T. Saito and M. Rehmsmeier, "The precision-recall plot is more informative than the ROC plot when evaluating binary classifiers on imbalanced datasets," PLoS ONE, vol. 10, no. 3, e0118432, 2015.

[11] C. Scott and R. Nowak, "A Neyman-Pearson approach to statistical learning," IEEE Trans. Information Theory, vol. 51, no. 11, pp. 3806–3819, 2005.

[12] N. Dalvi, P. Domingos, Mausam, S. Sanghai, and D. Verma, "Adversarial classification," in Proc. 10th ACM SIGKDD Int. Conf. Knowledge Discovery and Data Mining, 2004, pp. 99–108.

[13] J. Ebrahimi, A. Rao, D. Lowd, and D. Dou, "HotFlip: White-box adversarial examples for text classification," in Proc. 56th Annu. Meeting of the Association for Computational Linguistics (ACL), 2018, pp. 31–36.

[14] J. Gao, J. Lanchantin, M. L. Soffa, and Y. Qi, "Black-box generation of adversarial text sequences to evade deep learning classifiers," in Proc. IEEE Security and Privacy Workshops (SPW), 2018, pp. 50–56.

[15] J. Li, S. Ji, T. Du, B. Li, and T. Wang, "TextBugger: Generating adversarial text against real-world applications," in Proc. Network and Distributed System Security Symp. (NDSS), 2019.

[16] D. Pruthi, B. Dhingra, and Z. C. Lipton, "Combating adversarial misspellings with robust word recognition," in Proc. 57th Annu. Meeting of the Association for Computational Linguistics (ACL), 2019, pp. 5582–5591.

[17] The Unicode Consortium, "Unicode Security Mechanisms," Unicode Technical Standard #39. [Online]. Available: https://www.unicode.org/reports/tr39/

[18] The Unicode Consortium, "Unicode Normalization Forms," Unicode Standard Annex #15. [Online]. Available: https://www.unicode.org/reports/tr15/

[19] I. J. Goodfellow, J. Shlens, and C. Szegedy, "Explaining and harnessing adversarial examples," in Proc. Int. Conf. Learning Representations (ICLR), 2015.

[20] J. Wei and K. Zou, "EDA: Easy data augmentation techniques for boosting performance on text classification tasks," in Proc. Conf. Empirical Methods in Natural Language Processing and 9th Int. Joint Conf. Natural Language Processing (EMNLP-IJCNLP), 2019, pp. 6382–6388.

[21] A. Joulin, E. Grave, P. Bojanowski, and T. Mikolov, "Bag of tricks for efficient text classification," in Proc. 15th Conf. European Chapter of the Association for Computational Linguistics (EACL), 2017, pp. 427–431.

[22] K. Weinberger, A. Dasgupta, J. Langford, A. Smola, and J. Attenberg, "Feature hashing for large scale multitask learning," in Proc. 26th Int. Conf. Machine Learning (ICML), 2009, pp. 1113–1120.

[23] B. Jacob et al., "Quantization and training of neural networks for efficient integer-arithmetic-only inference," in Proc. IEEE/CVF Conf. Computer Vision and Pattern Recognition (CVPR), 2018, pp. 2704–2713.

[24] F. Pedregosa et al., "Scikit-learn: Machine learning in Python," Journal of Machine Learning Research, vol. 12, pp. 2825–2830, 2011.

[25] E. B. Wilson, "Probable inference, the law of succession, and statistical inference," Journal of the American Statistical Association, vol. 22, no. 158, pp. 209–212, 1927.

[26] F. Wilcoxon, "Individual comparisons by ranking methods," Biometrics Bulletin, vol. 1, no. 6, pp. 80–83, 1945.

[27] J. Demšar, "Statistical comparisons of classifiers over multiple data sets," Journal of Machine Learning Research, vol. 7, pp. 1–30, 2006.

[28] S. Holm, "A simple sequentially rejective multiple test procedure," Scandinavian Journal of Statistics, vol. 6, no. 2, pp. 65–70, 1979.

[29] D. Timko and M. L. Rahman, "Smishing Dataset I: Phishing SMS dataset from Smishtank.com," in Proc. 14th ACM Conf. Data and Application Security and Privacy (CODASPY), 2024.