FOREST: Phonetic Trie Ensembles with Logistic Regression Filtering for Low-Resource Approximate Named Entity Recognition

Tommy Nguyen · Marcos Didonet Del Fabro · Hegler Tissot

Working Manuscript — Drexel University, 2022

Abstract Named Entity Recognition (NER) in clinical, biomedical, and specialized technical domains frequently suffers from two practical bottlenecks: scarce annotated training corpora and high rates of typographical or phonetic misspellings. While contemporary deep learning models achieve strong accuracy on benchmark corpora, they require large labeled datasets and degrade under out-of-vocabulary spelling noise. Conversely, classical dictionary- and rule-based systems operate with zero annotated running text, yet exact lookups fail on misspelled mentions and naive approximate string search produces excessive false positives. We present **FOREST**, a rule-based approximate NER framework that builds an ensemble of tries over original surface strings and up to four phonetic encodings (Double Metaphone, Metaphone, NYSIIS, and Soundex), followed by a lightweight 10-feature Logistic Regression filter calibrated via precision–recall thresholding on synthetically perturbed dictionary terms. On a curated FDA pharmaceutical and medical lexicon (4,803 terms; 5,013 held-out perturbed test queries up to edit distance 4), a two-trie Double Metaphone and surface ensemble achieves reciprocal-rank-weighted precision of 0.9739, recall of 0.9270, and an F_1 score of 0.9499 at search edit distance 2. On a larger 58,966-term NASA ASRS vocabulary at edit distance 0, calibrated Logistic Regression filtering mitigates phonetic collision noise, lifting F_1 from 0.1435 to 0.5923 (Double Metaphone) and from 0.0183 to 0.5961 (five-trie phonetic ensemble). Complementary WordNet microbenchmarks (76,912 terms) show that active-node pruning accelerates edit-distance-2 queries by 7.0–9.1×, length-partitioned multi-trie search reduces latency by 18.1%, and binary serialization speeds index loading by 63.1×.

Keywords Named Entity Recognition · Trie · Fuzzy String Search · Phonetic Matching · Logistic Regression · Low-Resource NLP

T. Nguyen · H. Tissot
College of Computing and Informatics, Drexel University, Philadelphia, PA, USA
E-mail: nwyin@hey.com

M. Didonet Del Fabro
Department of Informatics, Federal University of Paraná (UFPR), Curitiba, PR, Brazil

1 Introduction

A variety of industries are generating text data at rapidly increasing scale. Clinical notes, electronic health records (EHRs), aviation safety logs, and scientific literature contain critical unstructured information whose analysis depends on natural language processing (NLP) techniques [13]. Named Entity Recognition (NER), a core subtask of NLP, identifies spans in free text referring to entities of interest—such as medications, symptoms, organizations, or locations—and assigns them semantic categories. Accurate NER underpins downstream pipelines including information retrieval [12], question answering [1], and relation extraction [4].

In EHRs and biomedical corpora, unstructured narratives capture symptoms, phenotypes, drug prescriptions, dosages, and treatment outcomes [12]. Mining specialized entities reliably with minimal annotation budgets remains a persistent challenge [13]. State-of-the-art NER models routinely reach F_1 scores in the mid-to-high 90th percentile on well-resourced benchmarks [9,8], and neural architectures can transfer across related domains with modest adaptation [12]. However, machine learning (ML) and deep learning (DL) systems

carry significant trade-offs in *low-resource domains*—specialized settings that possess only a curated domain lexicon of thousands to tens of thousands of terms and little or no annotated running text for supervised sequence labeling [9].

A second limitation of both neural and classical NER systems is sensitivity to typographical errors, phonetic misspellings, and adversarial character perturbations [9,7]. Generating and augmenting training sets to make deep sequence labelers robust to arbitrary misspellings is data- and compute-intensive [7], which is often impractical in niche domains. Rule- and dictionary-based NER offers an attractive alternative: rules and gazetteers require no annotated corpus to initialize, remain transparent and deterministic to modify, and yield predictable behavior without stochastic training variance [9,15]. Even so, standard dictionary lookups fail whenever input mentions deviate from canonical spellings—for instance, due to fast clinical charting or phonetic approximations—while hand-authoring exhaustive misspelling rules is prohibitively expensive [9,8]. Moreover, many domain entities span multi-word phrases (such as "Palace of Versailles" or multi-token disease names), compounding single-token spelling noise with boundary ambiguity.

Trie-based approximate string matching has emerged as a promising foundation for dictionary-driven NER. By tracking active prefix states bounded by edit distance, interactive fuzzy trie search supports millisecond-scale lookup over large lexicons [10]. Incorporating phonetic encodings further recovers misspellings that are far apart in character edit distance yet sound alike [6, 16].

In this work, we introduce **FOREST**, a rule-based approximate NER framework designed for low-resource domains where only a target dictionary is available. Our contributions are threefold. First, we combine an ensemble of surface and phonetic tries—supporting Double Metaphone, Metaphone, NYSIIS, and Soundex alongside the original string—with a lightweight 10-feature Logistic Regression (LR) filter trained on synthetically perturbed dictionary terms and calibrated via precision–recall thresholding, improving single-word recovery under spelling noise. Second, we fix a mid-word character-omission corner case in active-node trie traversal and quantify the precision–recall and runtime trade-offs of active-node pruning, phonetic ensemble size, length-partitioned tries, and binary serialization across three corpora: FDA pharmaceutical and medical terms, a NASA ASRS vocabulary, and Princeton WordNet. Third, we release three curated approximate-matching benchmarks alongside the open-source implementation to support reproducible evaluation of low-resource NER baselines. Extending the single-word engine to full multi-word expression boundary detection over continuous prose is framed as ongoing and future work.

2 Related Work

A *trie* (prefix tree) is a tree-structured index for fast key retrieval in which each node encodes a single character of a string, allowing keys with common prefixes to share storage paths. Tries are widely used when systems must perform rapid dictionary lookups across large vocabularies [15,6]. Prior database and systems research has optimized trie memory layout and cache locality—for example via cache-conscious HAT-tries [2] and Height Optimized Tries (HOT) [3]—demonstrating how structural optimizations directly improve lookup throughput and memory footprint.

To move beyond exact matching, active-node trie search tracks candidate prefix states along paths in the trie using pointers [10]. Let edit distance (ED) denote the minimum number of single-character insertions, deletions, or substitutions required to transform one string into another [11]. By pruning any node whose accumulated edit distance exceeds a maximum threshold k, active-node traversal finds all dictionary entries within bounded Levenshtein distance of a query prefix on the fly [10].

Several clinical and biomedical NLP systems adapt tries for dictionary-driven NER [6,15]. For example, *n-trie* accelerates lexicon-based clinical context rules by orders of magnitude over conventional implementations while maintaining deterministic behavior as rule sets grow [15]. Meanwhile, surveys of neural NER document high accuracy on standard benchmarks [20,9], alongside comparative EHR studies showing that rule-based systems remain competitive when labeled in-domain corpora are scarce [8].

In clinical free text, spelling errors frequently reflect how unfamiliar drug or anatomical names sound rather than pure keyboard adjacency slips [14,16]. Earlier work indexed phonetic keys using hash-based structures such as *PhoneticMap* [17] and combined character and phonetic similarity for Portuguese medication names [16]. Integrating phonetic normalization directly into fuzzy trie lookup—compressing a lexicon into a phonetic trie that is roughly 72% smaller than a surface trie—further improved approximate entity matching and secondary similarity re-ranking [6].

Finally, partitioning keys across multiple specialized tries—a "compact trie forest"—has proven effective for reducing memory overhead and search breadth in hardware IP lookup [5]. FOREST adapts this multi-trie

perspective to NLP by coordinating complementary surface and phonetic tries alongside length-partitioned tries, unifying their candidate sets through a calibrated linear classifier.

3 Materials and Methods

Rule-based NER offers lower computational cost and stronger interpretability than end-to-end deep neural architectures, allowing domain experts to inspect matched rules, adjust lexicons, and achieve consistent, deterministic updates without retraining large models [9,15]. To retain these advantages while overcoming exact-match brittleness and phonetic ambiguity, FOREST combines exact-prefix/active-node trie search across multiple string and phonetic representations with a lightweight Logistic Regression candidate filter. Figure 1 summarizes the end-to-end single-word query pipeline.

3.1 Active-Node Fuzzy Trie Engine

At the core of FOREST is an active-node trie traversal algorithm adapted from prior fuzzy keyword and phonetic trie search formulations [10,6]. Given a dictionary $\mathcal{D}$ inserted character-by-character into a prefix tree, each node v tracks its depth $\ell(v)$ and current running edit distance $d(v)$. Before processing a query string $q = q_1 q_2 \dots q_n$ under maximum edit distance $k \geq 0$, all nodes within depth $\ell(v) \leq k$ are initialized into depth-indexed active buckets $A[\ell]$ with $d(v) = \ell(v)$.

For each incoming query character q_i, active nodes are traversed top-down from maximum trie depth down to root 0:

1. *Downward expansion and substitution/match:* For every active node v, each child u labeled with character c receives updated distance

$$d(u) \leftarrow \min(d_{\text{prev}}(u),\, d(v) + \mathbb{I}[c \neq q_i])\,, \tag{1}$$

   and recursive character-skip expansion propagates $d(w) \leftarrow \min(d_{\text{prev}}(w), d(u)+1)$ to descendants w as long as $d(w) \leq k$.
2. *Deletion penalty and active-set pruning:* After expanding v, staying at v without consuming an edge (a query insertion / dictionary omission) increments $d(v) \leftarrow d(v) + 1$. Any node whose distance exceeds k is immediately pruned from $A[\ell]$.

During re-implementation of Ferri et al. [6], we identified and resolved a corner case in which valid dictionary words containing *mid-word character omissions* failed to remain reachable after non-matching intermediate steps; adding recursive downward propagation (`update_further_children`) together with strict depth-bucket cleanup restored expected recall across arbitrary insertion and omission positions. In addition, pruning active-node buckets immediately whenever $d(v) > k$ (rather than deferring filtering until query completion) avoids wasted expansion over high-ED branches. A configurable Jaro–Winkler similarity floor may also be enforced at leaf collection. Finally, to reduce lookup breadth on large dictionaries, FOREST optionally partitions entries across multiple length-bounded tries: if a trie only indexes entries with lengths in $[L_{\min}, L_{\max}]$ under edit bound k, any query outside $[L_{\min} - k,\, L_{\max} + k]$ is skipped in $O(1)$ time.

3.2 Phonetic Trie Ensemble and Logistic Regression Filtering

Phonetic encoders collapse spelling variations that sound similar into shared canonical codes, letting misspelled queries align closely—or even exactly—in phonetic space [6,16]. `PhoneticTrie` constructs an ensemble of up to five tries over the same lexicon $\mathcal{D}$: one over the unencoded *original* strings and up to four over distinct phonetic encodings:

- **Double Metaphone** (primary code),
- **Metaphone**,
- **NYSIIS** (New York State Identification and Intelligence System), and
- **Soundex**.

For each phonetic encoder ϕ, every entry $w \in \mathcal{D}$ is transformed into $\phi(w)$, indexed in an inverse dictionary mapping $\phi(w) \mapsto \{w' \in \mathcal{D} : \phi(w') = \phi(w)\}$, and inserted into ϕ's trie.

At query time, searching proceeds in two phases:

1. **Phase 1 (Candidate Generation):** The query q (or $\phi(q)$) is searched up to edit distance k in each trie of the ensemble. Phonetic matches are dereferenced via their inverse maps back to original dictionary strings, and results across all tries are deduplicated into a unified candidate set $C(q)$.
2. **Phase 2 (Supervised Scoring and Thresholding):** Because broad fuzzy or phonetic matching over multiple tries admits spurious collisions, FOREST pairs each (q, c) candidate ($c \in C(q)$) with a 10-dimensional feature vector $\mathbf{x}(q, c) \in \mathbb{R}^{10}$ consisting of the Jaro–Winkler similarity [19] and Levenshtein edit distance [11] evaluated across all five string/phonetic representations (original, Double Metaphone, Metaphone, NYSIIS, and

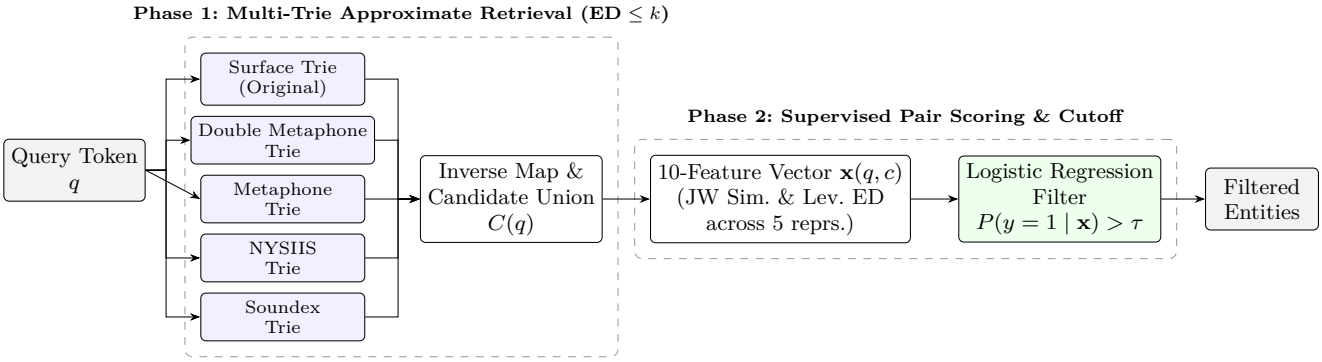


Fig. 1 Two-phase FOREST single-word matching architecture. In Phase 1, a query token is encoded and searched up to edit distance k across surface and phonetic tries, returning a deduplicated candidate set via inverse phonetic maps. In Phase 2, each candidate is scored by a 10-feature Logistic Regression classifier and retained only if its predicted match probability exceeds threshold τ.

Table 1 Summary of dictionaries and query splits used for single-word matching evaluation and runtime microbenchmarks.

Dataset	Terms	Train	Tuning	Test
FDA Drug & Medical	4,803	17,693	2,346	5,013
NASA ASRS Vocab.	58,966	210,929	30,877	60,052
WordNet Lexicon	76,912	—	—	48[†]

[†]Fixed benchmark set (8 queries per length 4–9).

Soundex). A binary Logistic Regression classifier predicts the match probability $p = P(y = 1 \mid \mathbf{x}(q, c))$, and candidates are accepted if $p > \tau$.

To calibrate τ without manually guessing weights, we run Phase 1 retrieval on a held-out *tuning* split, sweep thresholds along the precision–recall curve over validation probabilities, and pick the threshold $\tau^\star$ maximizing candidate-pair F_1. When no trained Logistic Regression model is attached, FOREST falls back to averaging weighted Jaro–Winkler scores across the four phonetic representations against a user-defined cutoff.

3.3 Datasets and Synthetic Perturbation Protocol

Because niche domains typically provide clean entity gazetteers without annotated free-text corpora, we construct controlled single-word matching benchmarks by introducing synthetic character perturbations (insertions, deletions, substitutions, and transpositions) up to edit distance 4. Table 1 summarizes the three datasets used in our experiments.

FDA Drug and Medical Terms. We curated 4,803 unique lowercase single-word entries primarily consisting of pharmaceutical drug names and medical terms from Food and Drug Administration (FDA) lists (word lengths 3–23 characters; mean 8.19, median 8). Target entities were randomly partitioned with *zero target-word overlap* across splits into training (3,387 unique targets; 17,693 queries), tuning (451 targets; 2,346 queries), and test (965 targets; 5,013 queries). Each split spans edit distances 0 through 4—for instance, the 5,013 test queries comprise 965 exact (ED = 0), 2,813 ED = 1, 926 ED = 2, 124 ED = 3, and 185 ED = 4 variants.

NASA ASRS Vocabulary. To stress-test ensemble scaling and phonetic filtering on a noisier, 12× larger vocabulary, we evaluated a second lexicon of 58,966 terms extracted from the NASA ASRS dataset, split into 210,929 training, 30,877 tuning, and 60,052 test queries across ED 0–4 (11,746 exact, 33,407 ED 1, 11,179 ED 2, 1,495 ED 3, and 2,225 ED 4 in test).

Princeton WordNet Lexicon. A 76,912-word dictionary (812,599 bytes in plain text) paired with 48 benchmark queries (eight words each of lengths 4 through 9) supports controlled systems microbenchmarks isolating candidate fan-out, active-node pruning speedups, length-partitioned multi-trie speedups, and pickle serialization overhead.

3.4 Evaluation Metrics

Approximate search frequently returns multiple candidate entities for a single query. Simply counting binary hit rate ignores whether the true entity appears at rank 1 or buried behind dozens of false matches. Following information retrieval evaluation practice, we adopt *reciprocal-rank-weighted* precision, recall, and F_1.

Let each test query q have ground-truth dictionary target $w^\star$ and return an ordered candidate list $C(q) = [c_1, c_2, \ldots, c_m]$. We accumulate true positives

(TP), false positives (FP), and false negatives (FN) across all test queries as:

$$TP \mathrel{+}= \sum_{r=1}^{m} \frac{1}{r} \mathbb{I}[c_r = w^\star], \tag{2}$$

$$FP \mathrel{+}= \sum_{r=1}^{m} \frac{1}{r} \mathbb{I}[c_r \neq w^\star], \tag{3}$$

$$FN \mathrel{+}= \mathbb{I}[w^\star \notin C(q)]. \tag{4}$$

Reciprocal-rank weighting ($1/r$) gives full weight to exact top-ranked predictions and penalizes false positives at high ranks more heavily than those further down the list, while incrementing FN by 1 directly penalizes failing to retrieve the target entity anywhere in $C(q)$. Aggregate Precision, Recall, and F_1 are then computed as:

$$\text{Precision} = \frac{TP}{TP + FP}, \quad \text{Recall} = \frac{TP}{TP + FN}, \tag{5}$$

$$F_1 = \frac{2 \cdot \text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}} = \frac{2TP}{2TP + FP + FN}. \tag{6}$$

4 Results

We evaluate FOREST along four axes: (1) end-to-end single-word matching accuracy on the FDA drug and medical dataset across search edit distances 0–2; (2) an ablation of phonetic ensemble size and Logistic Regression calibration on the larger NASA ASRS vocabulary; (3) query latency and candidate growth across word lengths on WordNet; and (4) length-partitioned multi-trie and binary serialization optimizations.

4.1 FDA Single-Word Matching Performance

As a baseline sanity check before approximate matching, we queried all 17,693 training perturbations at exact edit distance ED = 0 on the surface FDA trie. Queries averaged 0.106–0.109 ms per word (≈ 2.0 s total) and produced only two collision false positives where a synthetic edit turned one valid term into another dictionary entry (*ovral* → *oral* and *pellets* → *pellet*).

Table 2 reports held-out test performance (5,013 queries, including 1,235 queries at ED ≥ 2) for the two-trie FOREST configuration combining the surface trie and Double Metaphone trie with a calibrated Logistic Regression filter. Increasing the trie search radius from ED = 0 to ED = 2 produces a sharp rise in recall—from 0.6166 to 0.9009 at ED = 1 and 0.9270 at ED = 2—while Logistic Regression filtering keeps precision high across all radii (0.9877 at ED = 0 to 0.9739 at ED = 2). Consequently, F_1 climbs by +0.1907 points from 0.7592 to 0.9499. Notice also that even at search ED = 0—where exact surface lookup alone would only cover the 19.3% unperturbed (ED = 0) test fraction—incorporating the Double Metaphone trie already lifts recall to 0.6166 because many character typos preserve the underlying Double Metaphone code.

Table 2 Test precision, recall, and F_1 (reciprocal-rank-weighted) on the FDA dataset using surface + Double Metaphone tries with Logistic Regression filtering across search edit distances 0–2.

Search Max ED	Precision	Recall	F_1
0	0.9877	0.6166	0.7592
1	0.9802	0.9009	0.9389
2	0.9739	0.9270	0.9499

Table 3 Ablation of phonetic trie ensemble size and Logistic Regression calibration on the NASA ASRS test split (58,966 terms; 60,052 queries; trie ED = 0).

Ensemble (Tries)	Filter	$\tau^\star$	Prec.	Rec.	F_1
Surface (1)	Untuned	—	0.1956	0.1956	0.1956
Surf. + DMet. (2)	Untuned	—	0.0906	0.3457	0.1435
Surf. + DMet. (2)	Tuned LR	0.3876	**0.7909**	0.4734	0.5923
Surf. + 4 Phon. (5)	Untuned	—	0.0101	0.1026	0.0183
Surf. + 4 Phon. (5)	Tuned LR	0.3534	0.7473	**0.4958**	**0.5961**

4.2 Phonetic Ensemble and Calibration Ablation on ASRS

To isolate how phonetic trie ensembles and Logistic Regression calibration interact on a much larger, noisier vocabulary, we conducted an ablation on the 58,966-term NASA ASRS benchmark (60,052 test queries) with trie edit distance fixed to ED = 0. Under ED = 0, every fuzzy match must come solely from exact collisions in phonetic space, maximizing the burden on candidate filtering.

Table 3 reveals three findings. First, a surface-only trie at ED = 0 recovers only the 11,746 unperturbed queries (11,746/60,052 = 0.1956 precision, recall, and F_1); attempting to fit LR on exact-match-only pairs at ED = 0 fails (zero predictions at $\tau^\star = 0.8385$) because all retrieved pairs share zero edit distance and identical Jaro–Winkler similarity of 1.0, leaving no negative feature variance. Second, adding phonetic tries *without* calibration degrades overall F_1 sharply: unfiltered Double Metaphone collisions drop precision to 0.0906 ($F_1 = 0.1435$), and adding all four phonetic encodings drops untuned precision to 0.0101 ($F_1 = 0.0183$) due to coarse encodings such as Soundex matching hundreds of unrelated words and drowning out top-rank scores. Third, enabling the 10-feature Logistic Regression fil-

Table 4 Mean candidates returned and query latency (ms) on WordNet (76,912 terms; 8 queries per length), comparing unpruned vs. pruned active-node traversal (at ED = 0, every length averaged 1.00 candidate in ≤ 0.5 ms).

Len	Avg. Cand.		ED=1 Time (ms)		ED=2 Time (ms)	
	ED=1	ED=2	Unpr.	Pruned	Unpr.	Pruned
4	11.00	192.00	26.0	4.7	2,670.6	380.2
5	4.75	57.88	27.6	4.7	2,817.2	372.0
6	5.00	41.00	32.4	5.0	3,139.8	406.6
7	2.00	11.75	31.6	6.7	4,168.6	467.1
8	2.00	7.25	30.4	5.0	3,216.8	355.3
9	1.25	5.88	41.6	5.7	3,630.8	480.6

ter with validation-calibrated $\tau^\star$ resolves this collapse: precision rebounds to 0.7909 (Double Metaphone, $F_1 = 0.5923$) and 0.7473 (five-trie ensemble, $F_1 = 0.5961$), with the full five-trie ensemble achieving the highest recall (0.4958) and F_1 at ED = 0. Fitting the Logistic Regression classifier itself takes only 0.70 s once candidate pairs are extracted (whereas generating all 5-trie candidate feature tables over 210,929 training and 30,877 tuning queries took 2,503 s and 366 s, respectively, compared to 857 s and 120 s for Double Metaphone alone).

4.3 Word-Length Scaling and Active-Node Pruning

Microbenchmarks on the 76,912-term WordNet trie (Intel Core i7-8550U at 1.8 GHz, 12 GB RAM) quantify how candidate fan-out and query latency vary by word length and edit distance, and isolate the effect of pruning out-of-budget active nodes mid-traversal (Table 4).

At ED = 2, short 4-letter queries generate an average of 192.00 candidates per query (compared to 11.00 at ED = 1), whereas 9-letter queries return just 5.88 candidates on average. Pruning nodes as soon as their edit distance exceeds k reduces ED = 1 query latency from 26.0–41.6 ms down to 4.7–6.7 ms and accelerates ED = 2 queries from 2,671–4,169 ms down to 355–481 ms—a consistent **7.0–9.1× speedup** that cuts total benchmark execution time from 228.1 s to 81.5 s while returning identical candidates.

4.4 Length-Partitioned Multi-Trie and Serialization

Two additional structural optimizations target query latency and startup overhead on the 76,912-term WordNet dictionary:

- **Length-partitioned multi-trie routing:** Splitting the lexicon into two overlapping tries—a short-word trie (len ≤ 7, searched at ED = 1 for queries of length ≤ 6) and a long-word trie (len ≥ 5, searched at ED = 2 for queries of length ≥ 7)—produces assertion-verified identical outputs to querying a monolithic trie while reducing 3-run average batch search time over the 48 queries from 9.67 s to 7.92 s (**18.1% reduction**).
- **Binary trie serialization:** Serializing the constructed WordNet trie to disk expands storage size from 812,599 bytes (raw plain text) to 11,274,297 bytes ($\approx$ 13.9× footprint), but deserializing the pickle index takes only 1.38 s compared to 87.15 s for parsing text and inserting every word—a **63.1× startup speedup** (with 0.35 s serialization write time).

5 Discussion

When abundant annotated training corpora and compute are available, modern deep learning sequence labelers generally surpass purely rule-based pipelines [9, 8, 20]. FOREST addresses the complementary regime in which practitioners hold only a domain lexicon and cannot afford large-scale manual annotation. In such settings, a phonetic trie ensemble paired with an ultra-lightweight linear classifier trained in under a second on synthetic dictionary perturbations offers an immediate, interpretable baseline. Because candidate retrieval relies on explicit dictionary entries and transparent edit-distance and Jaro–Winkler features, domain experts can audit false positives, add or remove dictionary entries, or adjust the cutoff threshold τ deterministically without destabilizing downstream behavior.

Our experiments highlight two key empirical interactions. First, phonetic representations and supervised filtering are strictly complementary. Using Double Metaphone on the FDA dataset at ED = 0 triples recall relative to an exact-match ceiling (0.6166 vs. the 19.3% unperturbed split share), while extending trie search to ED = 2 reaches 0.9499 F_1. Yet on a larger 58,966-term vocabulary (ASRS), uncalibrated phonetic collisions collapse precision (to 0.0101 when combining all four phonetic algorithms) because coarse encodings group many semantically unrelated tokens. A 10-feature Logistic Regression filter calibrated on a small held-out tuning set restores high precision (0.7473–0.7909) while preserving the recall gain from phonetic diversity. Second, dictionary word length strongly shapes ambiguity: 4-character tokens return 32× more ED = 2 candidates than 9-character tokens on WordNet (192.0 vs. 5.9). Length-aware edit budgets and length-partitioned tries therefore cut redundant traversal without altering search semantics.

5.1 Limitations and Future Work

Several directions remain outside the scope of the present empirical evaluation and are left for future work:

1. **Multi-word expression recognition:** A prototype multi-word `Forest` class and synthetic sentence generator exist in the project repository, indexing concept identifiers (CUIDs), spaCy tokenization, and sliding-window token–concept context features (±2 neighboring tokens), alongside preliminary exploration on UMLS disease phrases and clinical text. Full end-to-end benchmarking on standard annotated multi-word NER corpora (e.g., CoNLL-2003 [18] or clinical evaluation suites), expression boundary disambiguation, and flexible span matching remain future work.
2. **Morphological normalization and runtime scaling:** Integrating lemmatization to normalize inflectional variants prior to trie lookup, extending experiments to ED ≥ 3 using parallel trie execution or compiled Cython/Rust extensions (as ASRS 5-trie candidate extraction at ED = 0 still requires 41.7 minutes in pure Python), and resolving occasional non-ASCII/numeric phonetic encoding edge cases.
3. **Cross-domain classifier transfer:** Evaluating whether the 10-feature Logistic Regression coefficients learned on one domain's perturbed lexicon (e.g., FDA drug names) transfer zero-shot to new target vocabularies (such as ASRS or UMLS subsets), eliminating even synthetic tuning on unseen domains.

6 Conclusions

We presented **FOREST**, a rule-based approximate Named Entity Recognition framework that combines surface and phonetic trie ensembles (Double Metaphone, Metaphone, NYSIIS, and Soundex) with a 10-feature Logistic Regression filter calibrated on synthetic dictionary perturbations. Experiments on a 4,803-term FDA pharmaceutical and medical lexicon demonstrated that combining Double Metaphone and surface tries with Logistic Regression filtering achieves 0.9499 F_1 (precision 0.9739, recall 0.9270) at edit distance 2 on out-of-vocabulary perturbed queries, while experiments on a 58,966-term NASA ASRS vocabulary showed that calibrated filtering overcomes coarse phonetic collision noise (F_1 rising from 0.0183 untuned to 0.5961 at ED = 0). Systems benchmarks on WordNet (76,912 words) confirmed that active-node pruning yields a 7.0–9.1× speedup at ED = 2, length-partitioned tries cut search time by 18.1%, and binary serialization accelerates dictionary loading by 63.1×. Code and datasets for this work are available at `https://github.com/nwyin/forest`.

Acknowledgements This work was conducted as part of the Senior Design Project sequence (CI 491–493) and Independent Study in Natural Language Processing (CS I499) at the College of Computing and Informatics, Drexel University.

References

1. Allam, A.M.N., Haggag, M.H.: The question answering systems: A survey. International Journal of Research and Reviews in Information Sciences (IJRRIS) **2**(3) (2012)
2. Askitis, N., Sinha, R.: HAT-trie: a cache-conscious trie-based data structure for strings. In: Proceedings of the Thirtieth Australasian Conference on Computer Science (ACSC), vol. 62, pp. 97–105 (2007)
3. Binna, R., Zangerle, E., Pichl, M., Specht, G., Leis, V.: HOT: A height optimized trie index for main-memory database systems. In: Proceedings of the 2018 International Conference on Management of Data, pp. 521–534 (2018)
4. Cohen, A.M., Hersh, W.R.: A survey of current work in biomedical text mining. Briefings in Bioinformatics **6**(1), 57–71 (2005)
5. Erdem, O., Carus, A., Le, H.: Compact trie forest: Scalable architecture for IP lookup on FPGAs. In: 2012 International Conference on Reconfigurable Computing and FPGAs, pp. 1–6 (2012). DOI 10.1109/ReConFig.2012.6416734
6. Ferri, J., Tissot, H., Del Fabro, M.D.: Integrating approximate string matching with phonetic string similarity. In: A. Benczúr, B. Thalheim, T. Horváth (eds.) Advances in Databases and Information Systems, pp. 173–181. Springer International Publishing, Cham (2018)
7. Goodfellow, I.J., Shlens, J., Szegedy, C.: Explaining and harnessing adversarial examples (2015)
8. Gorinski, P.J., Wu, H., Grover, C., Tobin, R., Talbot, C., Whalley, H., Sudlow, C., Whiteley, W., Alex, B.: Named entity recognition for electronic health records: A comparison of rule-based and machine learning approaches (2019)
9. Hahn, U., Oleynik, M.: Medical information extraction in the age of deep learning. Yearbook of Medical Informatics **29**, 208–220 (2020). DOI 10.1055/s-0040-1702001
10. Ji, S., Li, G., Li, C., Feng, J.: Efficient interactive fuzzy keyword search. In: Proceedings of the 18th International Conference on World Wide Web, WWW '09, pp. 371–380. Association for Computing Machinery, New York, NY, USA (2009). DOI 10.1145/1526709.1526760. URL `https://doi.org/10.1145/1526709.1526760`
11. Levenshtein, V.I.: Binary codes capable of correcting insertions, deletions, and reversals. Soviet Physics Doklady **10**(8), 707–710 (1966)
12. Meystre, S.M., Savova, G.K., Kipper-Schuler, K.C., Hurdle, J.F.: Extracting information from textual documents in the electronic health record: a review of recent research. Yearbook of Medical Informatics pp. 128–144 (2008)
13. Ross, M.K., Wei, W., Ohno-Machado, L.: "Big Data" and the electronic health record. Yearbook of Medical Informatics **23**, 97–104 (2014). DOI 10.15265/IY-2014-0003. URL `https://www.ncbi.nlm.nih.gov/pmc/articles/PMC4287068/`

14. Ruch, P., Baud, R., Geissbühler, A.: Using lexical disambiguation and named-entity recognition to improve spelling correction in the electronic patient record. Artificial Intelligence in Medicine **29**, 169–184 (2003). DOI 10.1016/s0933-3657(03)00052-6
15. Shi, J., Hurdle, J.F.: Trie-based rule processing for clinical NLP: A use-case study of n-trie, making the ConText algorithm more efficient and scalable. Journal of Biomedical Informatics **85**, 106–113 (2018). DOI 10.1016/j.jbi.2018.08.002. URL https://www.sciencedirect.com/science/article/pii/S1532046418301576
16. Tissot, H., Dobson, R.: Combining string and phonetic similarity matching to identify misspelt names of drugs in medical records written in Portuguese. Journal of Biomedical Semantics **10**(1), 17 (2019). DOI 10.1186/s13326-019-0216-2. URL https://doi.org/10.1186/s13326-019-0216-2
17. Tissot, H., Peschl, G., Del Fabro, M.D.: Fast phonetic similarity search over large repositories. In: H. Decker, L. Lhotská, S. Link, M. Spies, R.R. Wagner (eds.) Database and Expert Systems Applications, pp. 74–81. Springer International Publishing, Cham (2014)
18. Tjong Kim Sang, E.F., De Meulder, F.: Introduction to the CoNLL-2003 shared task: Language-independent named entity recognition. CoRR **cs.CL/0306050** (2003). URL http://arxiv.org/abs/cs/0306050
19. Winkler, W.E.: String comparator metrics and enhanced decision rules in the Fellegi-Sunter model of record linkage. In: Proceedings of the Section on Survey Research Methods, pp. 354–359. Washington, DC (1990)
20. Yadav, V., Bethard, S.: A survey on recent advances in named entity recognition from deep learning models. CoRR **abs/1910.11470** (2019). URL http://arxiv.org/abs/1910.11470