I J C R M
International Journal of Contemporary Research In Multidisciplinary

Research Article

HRS: A Devanagari-Aware Readability Metric for Hindi Text

Prabhat Chaudhary [1*], Dr. J. B. Singh [2], Rajesh Kumar Sharma [3]
[1] M. Tech Student, Department of CSE, Sagar Institute of Technology & Management Barabanki, Uttar Pradesh, India
[2] Professor, Department of Computer Science and Engineering, Sagar Institute of Technology and Management, Barabanki, Uttar Pradesh, India
[3] Assistant Professor, Department of Computer Science and Engineering, Sagar Institute of Technology and Management, Barabanki, Uttar Pradesh, India

Corresponding Author: * Prabhat Chaudhary

DOI: https://doi.org/10.5281/zenodo.20522230

Abstract

Automatic readability assessment for Hindi text has remained largely unsolved despite. Hindi being spoken by over 886 million internet users and around 14.7 lakh schools across India. Existing tools based on Flesch-Kincaid and related formulas fail on Devanagari script because they do not account for matra complexity (vowel diacritics), conjunct consonant density (virama-based fused consonants) or India's CBSE grade structure. This paper presents the Hindi Readability Score (HRS), a corpus-validated readability formula designed from scratch for the Devanagari script. HRS incorporates two novel features not found in any prior readability formula: "conjunct density detected via Unicode virama analysis (U+094D) and matra complexity based on guru/laghu syllable weight". We validate HRS against a 49 - sentence corpus drawn from NCERT Class 1-12 textbooks, Constitution of India, legal texts and Hindi news sources. HRS achieves Pearson r = 0.81 with human-assigned difficulty ratings used and a Mean Absolute Error of 1.67 school grades. We also present the Hindi Grade Level (HGL) formula mapping HRS to CBSE school grades (Class 1 to college) calibrated via least-squares regression. The complete implementation is released as an open-source Python package (pip install hindi-readability) with zero external dependencies.

Manuscript Information

- **ISSN No:** 2583-7397
- **Received:** 10-04-2026
- **Accepted:** 31-05-2026
- **Published:** 03-06-2026
- **IJCRM:**5(3); 2026: 702-705

- **Plagiarism Checked:** Yes
- **Peer Review Process:** Yes

How to Cite this Article

Chaudhary P, Singh J B, Sharma R K. HRS: A Devanagari-Aware Readability Metric for Hindi Text. Int J Contemp Res Multidiscip. 2026;5(3):702-705.

Access this Article Online

www.multiarticlesjournal.com

KEYWORDS: Hindi NLP, readability assessment, Devanagari script, text complexity, CBSE grade level, Indic language processing, conjunct consonants, matra complexity.

1. INTRODUCTION

Readability assessment, the automatic measurement of how easy or difficult a text is to read has been a solved problem in English for over 75 years. The Flesch Reading Ease formula was developed in 1948. It is embedded in Microsoft Word, used by the US Navy for training materials and cited in over 10,000 research papers. India, by contrast, has no equivalent tool for Hindi despite the language serving 24.8 crore school students (UDISE+ 2023-24) [8], 886 million internet users [9], and 14.7 lakh schools producing Hindi educational content.

The absence of Hindi readability tools has tangible consequences. A teacher assigning a Hindi passage has no automated way to verify whether it is appropriate for Class 5 or Class 10 students. EduTech platforms such as BYJU'S and Vedantu generate thousands of Hindi explanations daily with no systematic readability checking. Government portals serving rural populations must be written at a Class 5-6 level, but no tool exists to verify this.

English readability formulas cannot be applied to Hindi because Hindi has three structural features that English formulas are completely blind to. First, Devanagari Matras (vowel diacritics) create heavy and light syllables with different cognitive loads. Second, conjunct consonants formed when the virama character (U+094D) fuses two consonants into a single glyph appear almost exclusively in Sanskrit-origin tatsam vocabulary, which is significantly harder for younger readers. Third, India uses the CBSE/NCERT grade structure (Class 1-12) rather than the US K-12 system that Flesch-Kincaid maps to.

This paper makes the following contributions:

- We present HRS, the first corpus-validated readability formula for Hindi, incorporating two novel Devanagari-specific features.
- We present HGL, a CBSE-aligned grade level formula calibrated via regression on a 49-sentence human-graded corpus.
- We release a complete open-source Python implementation (hindi-readability v0.3.0) with zero external dependencies, available on PyPI and demonstrated on Hugging Face Spaces.

2. RELATED WORK

2.1 English Readability Formulas

The Flesch Reading Ease formula [1] remains the most widely used readability metric, computing ease from syllables per word and words per sentence. The Flesch-Kincaid Grade Level formula [2] maps this to US school grades. The Gunning Fog Index [3] identifies 'hard words' by syllable count. All three share a fundamental limitation. they model syllables using English phonological rules that do not apply to Devanagari script and their output US K-12 grade levels rather than CBSE grades.

2.2 Indic NLP Libraries

The IndicNLP Library [4] provides tokenization, morphological analysis and sentence segmentation for 12 Indic languages, but does not include readability scoring. iNLTK [5] offers pre-trained ULMFiT models for 9 Indian languages but focuses on classification and generation rather than readability measurement. MuRIL [6] and IndicBERT v2 [7] are transformer models for Indic languages that achieve strong performance on standard NLP benchmarks neither provides a readability metric or CBSE grade output.

2.3 Multilingual Readability

T Naous (2023) [11] presents ReadMe++ a multilingual readability dataset covering Arabic, English, French, Hindi and Russian with CEFR-level annotations. While this is the only prior work with Hindi readability annotations. It uses CEFR levels (A1-C2) rather than CBSE grades, provides no Python package and focuses on benchmark evaluation rather than a deployable tool. Our work is complementary: we provide a practical zero-dependency implementation with CBSE alignment that is directly usable by Indian educators.

2.4 RESEARCH GAP

No prior work provides:

- A corpus-validated readability formula using Devanagari-specific features (mantras and conjuncts).
- CBSE-aligned grade output.
- A publicly installable Python package and this work fills all three gaps.

3. Devanagari Script Features for Readability

Three Devanagari-specific features drive reading difficulty in Hindi beyond what syllable length and sentence length can capture.

3.1 Matra Complexity

In Devanagari, every consonant carries an implicit vowel (schwa). Matras (Unicode range U+093E - U+094C) are vowel diacritics that override the default vowel. Long mantras (U+093E), ii (U+0940), uu (U+0942), ai (U+0948), au (U+094C), e (U+0947), o (U+094B) create heavy (guru) syllables associated with more formal, complex vocabulary. Short Matras create light (laghu) syllables. Matra complexity is defined as the ratio of long Matras to total Matras in the text:

Matra complexity = count (long Matras) / count (total Matras)

A text with high matra complexity tends to use tatsam (Sanskrit-origin) vocabulary which is harder for younger readers.

3.2 Conjunct Consonant Density

The virama (halant, U+094D) suppresses the implicit vowel of a consonant and fuses it with the following consonant and creates a conjunct consonant. Examples include ksha (formed by ka + virama + sha), tra (ta + virama + ra), and jnya (ja + virama + nya). Conjunct consonants are the most reliable marker of textual difficulty in hindi. They appear almost exclusively in tatsam vocabulary used in formal, literary, legal and scientific register. We detect conjuncts by scanning for the pattern: consonant + virama (U+094D) + consonant. This requires no dictionary lookup or external library only Unicode character analysis: conjunct density = count (virama followed by consonant) / wordcount x 100.

To our knowledge, no prior readability formula for any language uses virama-based conjunct detection as a difficulty signal. This is the primary novel contribution of this work.

3.3 Devanagari Syllable Counting

Standard English syllable counting rules are inapplicable to Devanagari. We implement phonologically accurate syllable counting, each independent vowel (U+0904-U+0914) or consonant not killed by a virama constitutes one syllable nucleus. A virama suppresses the schwa and does not create a new syllable. Anusvara (U+0902) and visarga (U+0903) extend the preceding syllable but do not add new ones.

4. The HRS Formula

4.1 Hindi Readability Score (HRS)

The Hindi Readability Score is a 0 -100 ease score (higher = easier), inspired by Flesch Reading Ease but extended with Hindi-specific features:

HRS = 206.0 - (60.0 x avg_syllables/word) - (1.8 x avg_words/sentence)
- (70.0 x conjunct_density) - (8.0 x matra_complexity)

The weights reflect linguistic reasoning:

Syllable count per word is the dominant difficulty driver (weight 60, comparable to Flesch's coefficient of 84.6). Conjunct density receives the highest weight (70) because it is the strongest signal of Sanskrit-heavy vocabulary which is significantly harder for younger readers. Matra complexity receives a lower weight (8) as it provides a secondary signal. Sentence length (weight 1.8) follows the same role as in Kincaid but with a lower coefficient reflecting that Hindi sentence structure differs from English.

4.2 Hindi Grade Level (HGL)

The HGL formula maps HRS to CBSE school grades (1 to 13, where 13 = College+):

HGL = 10.083 - (HRS x 0.1088)

The coefficients 10.083 and 0.1088 were derived by least-squares linear regression on the 49-sentence validation corpus, minimising the sum of squared differences between predicted grade and human-assigned CBSE grade. The initial intuition-based coefficients (17.2 and 0.14) produced a MAE of 4.22 grades. The corpus-calibrated coefficients reduce MAE to 1.67 grades.

4.3 Hindi Complexity Index (HCI)

The HCI is a normalised 0-1 composite score useful for dataset labelling and ML pipeline features:

HCI = 0.40 x syl_score + 0.20 x sent_score + 0.25 x conjunct_score + 0.15 x matra_score

where each sub-score is normalized to [0,1] against empirical maxima.

5. Validation

5.1 Corpus Construction

We constructed a validation corpus of 49 sentences drawn from the following sources:

NCERT Hindi textbooks Class 1-12 (providing authentic grade-labelled text), Constitution of India (college-level legal text), legal documents, and Hindi news sources. Each sentence was assigned a human grade (1-13 on the CBSE scale) based on the source grade level and a difficulty score (1-10) based on reading judgment. This represents genuine human annotation. NCERT grade assignments are determined by expert curriculum designers and linguists, constituting a form of expert annotation analogous to teacher labelling.

5.2 RESULTS

Table 1 presents the validation results of the HRS formula against the human-graded corpus.

Table 1: HRS Validation Results on 49-Sentence Human-Graded Corpus

Metric	Result	Interpretation
Pearson r	0.81	Strong correlation with human judgment
Spearman rho	0.75	Consistent rank ordering
MAE (grades)	1.67	Less than 2 school grades error on average
Accuracy +/-1 grade	40.8%	20 of 49 sentences within 1 grade
Accuracy +/-2 grades	73.5%	36 of 49 sentences within 2 grades
Corpus size	49	NCERT Class 1-12, Constitution, Legal, News

5.3 Per-Level Analysis

Table 2 presents per-difficulty-level performance showing how accuracy varies across the difficulty spectrum.

Table 2: Per-Level Performance Breakdown

Difficulty Level	N	Avg. HRS	MAE	+/-1 Accuracy
Easy (Diff 1-2)	17	66.8	1.06	59%
Simple (Diff 3-4)	9	22.0	2.67	11%
Standard (Diff 5-6)	8	13.7	1.50	50%
Hard (Diff 7-8)	7	5.8	1.00	71%
Expert (Diff 9-10)	8	0.6	2.62	0%

6. Implementation

6.1 Package Architecture

The hindi-readability package (v0.3.0) is structured as three modules with clean separation of concerns. script.py

File implements Devanagari Unicode analysis matra counting, virama detection, conjunct identification and syllable counting using only the Python standard library and nothing. formulas.py implements the HRS, HGL and HCI formulas.scorer.py provides the readability scorer by public API with five methods: score (), compare (), batch_score(), is_appropriate_for_grade() and simplify_suggestions().

6.2 Key Design Decisions

- **Zero external dependencies:** All Unicode analysis use Python built-in Unicode data modules. This ensures the package installs in under 2 seconds and runs on all devices without GPU or internet access.
- **Python 3.8+ compatibility:** Tested on Python 3.8, 3.9, 3.10, 3.11, and 3.12 via GitHub Actions CI on every commit.

- **NFC normalization:** All input text is normalized to Unicode NFC form before analysis, ensuring consistent results regardless of how the Devanagari text was encoded.
- **CBSE grade mapping:** Grade output uses Indian school level names (Prathmik, Madhyamik, Uccha Madhyamik, Snatak) rather than generic numbers.

7. DISCUSSION

7.1 Strengths

The primary strength of this work is the conjunct density feature. No prior readability formula for any language has used virama-based conjunct detection as a difficulty signal. The linguistic motivation is strong: conjunct consonants are morphological markers of Sanskrit-origin vocabulary, and Sanskrit-derived words are a reliable indicator of formal, difficult text in Hindi. The Pearson r = 0.81 on a genuine human-annotated corpus is comparable to published validation results for English readability formulas on similar corpus sizes.

7.2 LIMITATIONS

The primary limitation is corpus size. While 49 genuinely human-graded sentences is comparable to early English readability validation work, a larger corpus of 200+ sentences annotated by multiple Hindi teachers with measured inter-rater reliability would strengthen the statistical claims. Additionally, the formula does not currently account for topic familiarity, sentence structure complexity (embedding depth, relative clauses), or domain-specific vocabulary. The formula has been validated on formal Hindi (educational, legal, news) but has not been tested on Hinglish (code-mixed Hindi-English), which is the dominant register on social media.

7.3 Comparison to Existing Work

Compared to ReadMe++ [11], this work differs in three keyways: (1) CBSE grade output rather than CEFR levels, (2) a zero-dependency installable package rather than a research dataset, and (3) formula-based computation rather than transformer fine-tuning, making it suitable for low-resource deployment. A direct comparison of HRS against a Hindi-fine-tuned IndicBERT model on the same corpus is planned as a dissertation extension.

8. Future Work

Three research directions are planned for the Year 2 dissertation. First, corpus expansion: collecting 200+ Hindi sentences from diverse sources (NCERT, news, social media, legal) and annotating them with 10 Hindi teachers, computing Cohen's Kappa for inter-rater reliability. Second, formula recalibration: applying least-squares regression on the expanded corpus to update formula weights, measuring improvement in MAE and Pearson r over the current baseline. Third, ML baseline comparison: fine-tuning IndicBERT v2 for readability regression and comparing its per-level MAE against HRS, establishing whether the additional computational cost of a transformer model is justified for this task.

9. CONCLUSION

We have presented HRS, the first corpus-validated readability metric for Hindi text, incorporating two novel Devanagari-specific features: virama-based conjunct density and matra complexity. The formula achieves Pearson r = 0.81 on a 49-sentence human-graded corpus and is available as a zero-dependency Python package (pip install hindi-readability). The Hindi Grade Level formula provides CBSE-aligned grade output calibrated by regression. We believe this work establishes an important baseline for Hindi text readability research and provides a practical tool for Indian educators, content creators, and NLP researchers.

REFERENCES

1. Flesch R. A new readability yardstick. Journal of Applied Psychology. 1948;32(3):221-233.
2. Kincaid JP, Fishburne RP Jr, Rogers RL, Chissom BS. Derivation of new readability formulas for Navy enlisted personnel. Memphis (TN): Naval Technical Training Command, Research Branch; 1975. Report No.: 8-75.
3. Gunning R. *The Technique of Clear Writing*. New York: McGraw-Hill; 1952.
4. Kunchukuttan A. The IndicNLP Library. arXiv [Preprint]. 2020. Available from: https://arxiv.org/abs/2005.00085
5. Arora G. iNLTK: Natural Language Toolkit for Indic Languages. arXiv [Preprint]. 2020. Available from: https://arxiv.org/abs/2009.12534
6. Khanuja S, Bansal D, Mehtani S, Khosla S, Dey A, Gopalan B, *et al*. MuRIL: multilingual representations for Indian languages. arXiv [Preprint]. 2021. Available from: https://arxiv.org/abs/2103.10730
7. AI4Bharat. IndicBERT v2: a robust multilingual language model for Indic languages. In: Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing; 2022; Abu Dhabi, United Arab Emirates.
8. Ministry of Education, Government of India. Unified District Information System for Education Plus (UDISE+) 2023-24. New Delhi: Ministry of Education; 2024.
9. Internet and Mobile Association of India. India Internet Report 2024. Mumbai: Internet and Mobile Association of India; 2024.
10. Unicode Consortium. The Unicode Standard, Version 15.0: Devanagari Block U+0900-U+097F [Internet]. Mountain View (CA): Unicode Consortium; 2022 [cited 2026 Jun 1]. Available from: https://unicode.org/charts/PDF/U0900.pdf

11. Naous T, Valentino C, Bontcheva K, Chen X. ReadMe++: benchmarking multilingual language models for multi-domain readability assessment. arXiv. 2023. Available from: https://arxiv.org/abs/2305.14463

About the Corresponding Author

Prabhat Chaudhary is an M. Tech student in the Department of Computer Science and Engineering at Sagar Institute of Technology & Management, Barabanki, Uttar Pradesh, India. His academic interests include computer science, software development, and emerging technologies. He is engaged in advanced technical learning and research-oriented studies in the field of engineering.