String Similarity Evaluation Data

Version: 1.0
Date: 23 July 2026

1. Background

As recommended in the Generic Names Supporting Organization (GNSO) Final Report on the New gTLD Subsequent Procedures Policy Development Process, the String Similarity Review is a part of the New gTLD Program: 2026 Round application evaluation. The objective of this review is to prevent user confusion and loss of confidence in the Domain Name System (DNS) resulting from delegation of similar strings. The Phase 1 Final Report on the Internationalized Domain Names Expedited Policy Development Process adds further details on this process and the scope of string similarity analysis to cover strings (e.g., other applied-for strings, existing top-level domains, blocked names) and their variant strings. It also requires the development of the String Similarity Review Guidelines for the String Similarity Panel (SSE panel). Details on String Similarity Evaluation are provided in Applicant Guidebook (AGB) Section 7.10.

The comparisons between strings are performed for String Similarity Evaluation (SSE). With variant strings included in the scope, the number of comparisons increases significantly. Therefore, the String Similarity Evaluation Tool (SSE tool) has been developed to perform prescreening and generate the potential contention sets as an input for the SSE panel to analyze and make decisions. The SSE panel can override the suggestions by the SSE tool. This methodology was suggested in the draft of the String Similarity Review Guidelines published for public comment in 2024.

2. Introduction

ICANN has gathered String Similarity Evaluation Data (SSE data) for determining visual string similarity for top-level domains (TLDs). As groundwork, data has been gathered at code point level for the full repertoire of the Root Zone Label Generation Rules (RZ-LGR). The task was extensive and required script-specific knowledge. This data defines the level of similarity between code points or code point sequences in the RZ-LGR repertoire and will be used in the SSE tool.

It is important to note that the data is designed to be able to identify the potential visually similar code points, beyond the variant code points. In case of ambiguity if the code point pair is similar or not, a conservative approach is used by including them into the similarity set. Additionally, some code points are included in the similarity set to maintain the transitivity of the set, even if they are marked with lower similarity scores.

The prescreening SSE tool uses this SSE data to determine the potentially similar strings which are presented in the potential contention sets generated by the tool.

These contention sets are not necessarily the final contention sets and will be considered by the SSE panel for the final determination of the contention sets. It is the responsibility of the SSE panel, through its manual review, to finalize the contention sets arising from visual string similarity. The SSE panel may change the contention sets identified by the SSE tool using the SSE data.

ICANN org has also developed the String Similarity Evaluation Guidelines (SSE guidelines) to guide the SSE panel to make their final determination using the SSE tool and the SSE data.

3. Methodology

3.1 Scope of Work

ICANN has collected input from the relevant script experts listed in Section 6. These experts analyzed the Root Zone Label General Rules (RZ-LGR) repertoire for their respective scripts, as well as cross-script comparisons with related scripts.The work had the following scope for a script:

A. Compare all elements[1] to ASCII characters, in uppercase and lowercase form (a-z, A-Z)
B. Compare all elements to each other
C. Compare all elements to elements in the repertoire of other related scripts (if any)
D. Compare all elements to:
    a. Capitalized elements of the related scripts (if applicable)
    b. Conjuncts or compounded characters (if applicable for the script)
    c. Simple shapes in ASCII, Latin, and other related scripts (e.g., circles “o”, lines “l”, curves “c”, “s”, “u”, “Ո”)
    d. Underlined elements, as used in URL links

3.2 Similarity Category

The similarity categorization consists of the following five levels:

1. Identical or near identical (homograph) or already defined variants
2. Highly confusable (strongly similar, or near homograph)
3. Similar
4. Distantly similar (weakly similar)
5. Distinct, not similar

[1] An element in this context refers to the visual representation of either a single code point or any valid sequence of code points within the repertoire of the script in RZ-LGR to be reviewed as a single unit within a domain label.

3.3 Data Collection and Preparation

The task was partitioned into multiple sub-tasks described below:

- Expert's Work
  - Each expert reviewed their respective script repertoire and compared all code point glyphs with each other and with the code point glyphs of related scripts (including ASCII characters).
  - Each expert noted the similar code points along with the degree of similarity and the reason for similarity based on the criteria presented above.
  - Some experts developed additional mechanisms to perform their work within the guidelines. These mechanisms are listed in the *Additional Notes on Script Data* section later in this document.

- Manual Review and Programmatic Checks
  - The similarity data received from the experts was reviewed based on a list of items ranging from syntactical checks (e.g., the glyph and code points match) to scope completeness checks (e.g., confirming that the script character and ASCII characters comparison has been conducted). Any issues were reported back to the expert for explanation or remediation. If necessary, this step was repeated.
  - Following syntactic and completeness checks, to review consistency, further cross-script alignment was conducted on the data. All cases were identified in which the cross-script classifications of the two script experts involved were different (e.g., one script expert marked a different category than another script expert on the same cross-script element pair). In such cases, ICANN org coordinated with relevant experts to align the data for relevant scripts.
  - Following the alignment, the transitivity check was done by integrating all similarity data across scripts as well as variant code point information defined in the RZ-LGR.
    - All variant pairs in Common RZ-LGR, which includes all variant definitions across scripts, were added as "Variant" and marked as Category 1.
    - In addition, pairs that were not identified explicitly by experts as similar were added as "Imposed" due to transitivity. For example, if A and B are marked as similar, A and C are marked as similar, then B and C will be "Imposed" into the same similarity set. In such cases, the imposed pairs were marked with similarity Category 4, because they had not been

identified by the script experts.

- Conversion to Data Files
  - The cumulative data for each script was converted into the XML structure in the LGR format (RFC 7940) to be processed by the SSE tool.
  - The equivalent HTML version was created to facilitate the manual review of these data files.

4. String Similarity Evaluation Data

The SSE data package includes one **Common Similarity Data** file and 26 **Script Similarity Data** files. Both kinds of files are needed for different purposes in the calculation of the string similarity as per the details below.

- The Common Similarity Data file is the merger of all Script Similarity Data files and the Common RZ-LGR. It is used for calculating the similar strings sets.
- Script Similarity Data files are generated based on the script expert input and variant definition in the Common RZ-LGR. They are used for calculating the level of similarity between strings within a similar strings set. To make these data files usable for the intended distance calculation, they were mechanically augmented by some of the mappings, tags, classes, and context rules found in the Common RZ-LGR.

All files can be collectively downloaded with this package.

Script	Similarity Data	
Common	HTML	XML
Arabic	HTML	XML
Armenian	HTML	XML
Bangla (Bengali)	HTML	XML
Chinese	HTML	XML
Cyrillic	HTML	XML
Devanagari	HTML	XML
Ethiopic	HTML	XML

Script	Similarity Data	
Georgian	HTML	XML
Greek	HTML	XML
Gujarati	HTML	XML
Gurmukhi	HTML	XML
Hebrew	HTML	XML
Japanese (Han + Hiragana + Katakana)	HTML	XML
Kannada	HTML	XML
Khmer	HTML	XML
Korean (Han + Hangul)	HTML	XML
Lao	HTML	XML
Latin	HTML	XML
Malayalam	HTML	XML
Myanmar	HTML	XML
Oriya	HTML	XML
Sinhala	HTML	XML
Tamil	HTML	XML
Telugu	HTML	XML
Thaana	HTML	XML
Thai	HTML	XML

4.1 Important Notes on Script-Specific Data

4.1.1 Hangul Script

The analysis of Hangul script was done by the following methodology. Pairs of similar Hangul syllable-initial letters are identified, e.g., (ㄹ, ㅌ). Then each pair of syllables (ㄹ, sp1, sf1) and (ㅌ, sp1, sf1) is compared where sp1 is one of 21 syllable-peak letters and sf1 is one of 27 syllable-final letters or null (i.e., no syllable-final letter).

A similar approach is adopted for similar syllable-final letters: each pair of syllables (si2, sp2,ㄹ) and (si2, sp2,ㅌ) is compared where si2 is one of 19 syllable-initial letters.

Examples of similar syllable-peak letters are (ㅗ, ㅡ), (ㅜ, ㅡ), (ㅚ, ㅟ). For these, each pair of syllables (si3, ㅗ, sf3) and (si3, ㅡ, sf3) is compared.

4.1.2 Han script

The Han script data was analyzed by both a Chinese expert and a Korean expert. There are 19,685 Han code points in Chinese RZ-LGR and 4,758 Han code points in Korean RZ-LGR. Of these, 4,744 code points overlap.

Due to the large size of Han script code points to compare, the Chinese script expert used automated machine learning tools, including a Hanzi similarity Natural Language Processing (NLP) tool and an image visual similarity tool, to identify and categorize code point pairs for experts to review and make final decisions.

The Hanja (Han character used in Korean language) script expert referred to several sources containing pairs or sets of similar Hanja characters. Sets of similar Hanja characters are converted to pairs of Hanja characters. Pairs of similar characters in shape are manually analyzed to determine how much the characters in each pair look similar. Small differences in the top or left component seem relatively well distinguished whereas minor differences in the middle, center, or bottom component do not seem so easily identified.

The inputs from the Chinese and Korean experts were then compared and aligned before publishing for Public Comment. After the Public Comment, the Chinese script expert performed additional analysis using Artificial intelligence (AI) powered factoring based on four core algorithms: Convolutional Neural Networks (CNNs), stroke structure analysis, Zernike moments, and grid histogram. This resulted in identification of 1,612 additional Han script similarity cases. Based on the conservatism principle, they have been incorporated into the Chinese, Japanese, and Korean script SSE data.

With the additional 1,612 cases, and after further consultation with script experts, the Han script Category 4 (not_confusing) cases have been removed from the Chinese, Japanese, Korean SSE data. This helps keep the file size manageable and does not impact the SSE tool calculation.

4.1.3 Neo Brahmi scripts

Neo Brahmi scripts in the RZ-LGR, including Devanagari, Bangla (Bengali), Gujarati, Gurmukhi, Kannada, Malayalam, Oriya, Tamil, and Telugu form strings following a particular mechanism, known as Akshar. These Akshars are typically formed by systematic combinations of the characters from the basic categories: Consonants (C), Matra/Vowel sign (M), Vowel (V), Anusvara/Bindu (B), Candrabindu (D), Visarga (X), Virama or Halant (H), Nukta (N). The potential basic akshar shapes formed by these characters can be achieved by following combinations, e.g.: C, CM, CB, CD, CX, CMB, CMD, CMX. In addition, there are special characters available in certain scripts which have their own systematic combinations i.e., Chillu in Malayalam, Addak in Gurmukhi, Khanda Ta in Bengali.

The consonant in the string can be C or a conjunct formed among the Cs joined by H, e.g. CHC, CHCHC, which creates conjuncted shapes which may or may not retain the basic shape of the consonants that participate in the conjunct formation.

Each script expert took into account these plausible combinations and the subsequent shapes they create into the similarity analysis.

A C and a C with a mark below may become visually similar in a string, when underlined. For example, ह (U+0939) vs. ह (U+0939 U+0956). The mark below often has an associated context rule and can have a cross-script similar or variant definitions with related scripts. So, in such cases, including similarity mapping between C and C+mark creates significant technical complexity. Therefore, such cases are not included in this version of SSE data for Neo Brahmi scripts. A careful manual review is required when a string includes any of the marks below, provided in the following list:

- Bengali script:
    - U+09BC (়) BENGALI SIGN NUKTA
    - U+09C1 (ু) BENGALI VOWEL SIGN U
    - U+09C2 (ূ) BENGALI VOWEL SIGN UU
    - U+09C3 (ৃ) BENGALI VOWEL SIGN VOCALIC R
    - U+09C4 (ৄ) BENGALI VOWEL SIGN VOCALIC RR
    - U+09CD (্) BENGALI SIGN VIRAMA

- Devanagari script:
    - U+093C (़) DEVANAGARI SIGN NUKTA
    - U+0941 (ु) DEVANAGARI VOWEL SIGN U
    - U+0942 (ू) DEVANAGARI VOWEL SIGN UU

    - U+0943 (ृ) DEVANAGARI VOWEL SIGN VOCALIC R
    - U+094D (्) DEVANAGARI SIGN VIRAMA
    - U+0956 (ॖ) DEVANAGARI VOWEL SIGN UE
    - U+0957 (ॗ) DEVANAGARI VOWEL SIGN UUE

- Gurmukhi script:
    - U+0A3C (਼) GURMUKHI SIGN NUKTA
    - U+0A41 (ੁ) GURMUKHI VOWEL SIGN U
    - U+0A42 (ੂ) GURMUKHI VOWEL SIGN UU
    - U+0A4D (੍) GURMUKHI SIGN VIRAMA

Further, the mapping of ঋ (U+09A4 U+09CD U+09A5) and ঌ (U+09A5) is not included in the SSE data due to the technical complexity caused in integration. A careful manual similarity review is required when strings contain these code point sequences.

4.1.4 Thai Script

There are some Thai characters which could be used to create strings similar to Latin strings (and vise versa) e.g. “ทนท” (U+0E17 U+0E19 U+0E17) can appear similar to “nun” (U+006E U+0075 U+006E) especially in some fonts where the ‘head’ of the Thai letter is not prominent. Such cases are included in the data as identified by the script expert.

However, there are some technical limitations found for integrating the similarity mapping between ท (U+0E17) and n (U+006E) due to complex cross-script interdependencies. Based on a consultation with the script expert, it was agreed that the mapping of ท (U+0E17) and n (U+006E), with Category 3, can be removed from the data set. A careful manual similarity review is required when reviewing a string containing ท (U+0E17).

5. File Structure and Conventions

These SSE data files present all the code points and sequences within and across the scripts that are deemed confusable in XML and HTML format. Though it uses the RFC7940 format in the XML and similar HTML representation as RZ-LGR, the contents of this document do not define Label Generation Rules but visual similarity data.

The HTML format is mechanically generated from the respective XML file for the string similarity data and contains the following structure.

5.1 Description

This section contains the overview of the SSE data file, the file format and conventions, overview of data collection and methodology

5.2 Repertoire

This section lists the repertoire by code point (or code point sequence). For any code point or sequence for which a confusable is defined, with additional information provided in the Confusables column. Some code points or sequences included in the repertoire table are not part of the repertoire itself. These code points are included for documenting out-of-repertoire confusable mappings as indicated in the Part of Repertoire column.

5.3 Confusables

This section lists all confusable sets derived from the data collected by the script expert. There are two types of the Confusable Set: Confusable Set for screening calculation and Confusable Set for documentation only.

5.3.1 Confusable set for screening calculation

This type is marked as "**Confusables Set {Set ID}.**" Confusables Set collects the mapping between code points and sequences, including the degree of their similarity. These sets are symmetric and transitive. They also include all variant mappings defined in the RZ-LGR along with the similar code points.

Each mapping has an associated "Reason" code that is a highlighted part of the Comment column. The reason codes used and their explanations are provided below:

- **Capitalization**: The similarity is between two uppercase forms or between an uppercase and unrelated lowercase form.
- **Confusing**: A majority of users familiar with the script of either target, source or both will confuse the pair.
- **Fallback**: One of the code points is a common fallback representation for the other.
- **Font**: The confusability depends on the choice of the font.
- **Handwriting**: The confusability is caused or strengthened by users being familiar with a shape, even though it is normally only used in handwriting,
- **Underlining**: The presence of underlining makes these confusable. This is typically the case when the distinction between two characters rests primarily on a mark below or a descender that might be partially or completely obscured when underlined.

- **Variant**: The mapping is defined as a variant in the RZ-LGR. Note: the category value for variant code points is set to 1 by definition.
- **Other**: None of the other reason codes apply. For example, this code is used when the category is homograph (1) for a mapping that exceptionally has not been defined as a variant in the RZ-LGR; or, when the category is level 4 for a mapping that is not confusing but is still used in the screening calculation. NOTE: This code is accompanied by a comment explaining the nature of the confusability and reasoning for choosing the assigned category value.

In addition, there are two special reason codes:

- **Imposed**: This Reason code is automatically generated when a mapping is mechanically added and imposed to make a confusable set transitive after variants have been injected. NOTE: All imposed mappings are assigned category 4 by default as these were not directly identified by the experts.
- **Injected confusable mapping**: This Reason code is automatically generated when a confusable mapping is injected from the Common Similarity Data file.

5.3.2 Confusable set for documentation only

This type is marked as "**Confusables Set {Set ID} [This set is for documentation only]**". Elements of this set are not processed by the SSE tool while performing the screening calculation, and are ignored. The documentation-only sets are included in the data primarily for reference and to document that it was considered by the expert team but explicitly resolved as not confusing despite some weak similarity.

Each mapping has an associated Reason code that is a highlighted part of the Comment column. The Reason code and its explanation are given below:

- **Imposed**: Same definition used in section 4.3.1.
- **Variant**: Same definition used in section 4.3.1.
  NOTE: Some variants may be added into a documentation-only set to make the set symmetric and transitive. All variant mappings injected from the RZ-LGR will have a Reason code of **Variant** with the category 1 even though it is present in the documentation-only set. These are also included in the set included for screening calculation where these are processed.
- **Not_Confusing**: Mappings that are marked with Reason code "Not_confusing" are listed here for reference only.

NOTE: For Han scripts, the listing of Confusable sets for documentation only is suppressed for reasons of file size. This has no effect on any processing and screening performed with the SSE data file by the SSE tool.

5.4 Classes and Rules

The section lists classes automatically generated from the tag on each code point and lists definitions of contextual rules. For the SSE data, there are two common rules.

- **excluded-similarity** — this special context rule is defined to match the empty string, which means it can never be satisfied. The mappings with this content rule are not used for calculation.
- f**ollowed-by...** — these rules ensure that confusable mappings are well-curated for index calculation consistency.

5.5 Table of References

The section lists the references cited for specific code points, variants, classes, rules or actions in the SSE data file.

6. Additional Significant Considerations

6.1 String Similarity Due to Uppercase Form

As some web browsers may accept domain names in uppercase forms and automatically convert them to lowercase form before resolving, the uppercase forms are included in the similarity consideration. This may result in some of the code points which are not visually similar in lowercase to be defined as similar if their uppercase forms are determined to be similar.

For example: n (U+006E, Latin Small Letter N) and ν (U+03BD, Greek Small Letter Nu) have visually the same uppercase letters.

n (U+006E, Latin) — uppercase → N (U+004E, Latin Capital Letter N)
ν (U+03BD, Greek) — uppercase → N (U+039D, Greek Capital Letter Nu)

Therefore, n (Latin Small Letter N) and ν (Greek Small Letter Nu) are defined as similar in the SSE data due to uppercase conversion. The SSE tool will calculate the similarity between strings accordingly e.g. the string “ana” (Latin) vs “ανα” (Greek) will be reported in the same potential contention set because their uppercase forms are

homoglyph. This potential contention set will be identified by the SSE tool, e.g. see below.

ana (Latin) and ανα (Greek) (lowercase)
ANA (Latin) and ANA (Greek) (uppercase)

6.2 String Similarity for ASCII Strings

In addition to the similarity data, variant definitions of all scripts in the RZ-LGR are combined into the SSE data to be able to identify the potential contention sets.

For example: v (U+0076, Latin Small Letter V) and ν (Greek Small Letter Nu) are variant code points as per the Latin script RZ-LGR and Greek script RZ-LGR.

Through transitivity, the code point v (U+0076, Latin Small Letter V) are included in the similarity set with n (U+006E, Latin Small Letter N) and ν (U+03BD, Greek Small Letter Nu) from the previous example.

Similarity set includes:

{ ν (U+03BD, Greek Small Letter Nu),
v (U+0076, Latin Small Letter V),
n (U+006E, Latin Small Letter N) }

Based on the data, the SSE tool will put the following strings in the same potential contention set: “ana” (Latin), “ava” (Latin), “ανα” (Greek). So, it is possible to have two ASCII strings identified by the SSE tool to be in the same potential contention set.

As per the RZ-LGR (Version 6) and the similarity information defined by the script experts, the following ASCII code points are identified in the same similarity sets

Set 1: {b, ss}
Set 2: {f, t}
Set 3: {h, n, ri, v}
Set 4: {i, l}
Set 5: {m, rn}
Set 6: {r, u, y}
Set 7: {s, un}
Set 8: {vv, w}

Even though being in the same potential similarity set, caused by transitivity and variant relationships within script or across script, the actual similarity between each pair of ASCII code points are all Category 4 (not confusing) or Category 5 (not defined), except the following pairs:

i (U+0069) and l (U+006C), Category 1
m (U+006D) and rn (U+0072 U+006E), Category 2
n (U+006E) and ri (U+0072 U+0069), Category 2
vv (U+0076 U+0076) and w (U+0077), Category 3

The i (U+0069) and l (U+006C) are considered similar because the uppercase i and lowercase l are homoglyphs in some fonts. Even though they are a mixed-case comparison for ASCII, they can create confusion especially when the case of the strings cannot be identified from adjoining code points, e.g. the string “lll”. Therefore, i and l are included in the similarity set which will be used to produce candidate similar strings by the SSE tool.

The m (U+006D) and rn (U+0072 U+006E) are defined by the script expert.

The n (U+006E) and ri (U+0072 U+0069) are included in the SSE data due to integration.The script expert defined the similarity between n (U+006E) and rı (U+0072 U+0131, Latin Small Letter R + Latin Small Letter Dotless I) with Category 2. As i (U+0069) and ı (U+0131) are variants as per the RZ-LGR. The n (U+006E) and ri (U+0072 U+0069) are included to ensure the transitivity of the set and the similarity level is Category 2 to align with the conservatism principle.

The vv (U+0076 U+0076) and w (U+0077) are defined by the script experts.

7. Contributors

Script Experts:

Akshat Joshi - Devanagari, Gujarati
Atiur Rahman Khan - Bengali (Bangla)
Dessalegn Yehuala - Ethiopic
Dmitry Belyavsky - Cyrillic
Gurpreet Singh Lehal - Gurmukhi
Harsha Wijayawardhana - Sinhala
Hiro Hotta - Japanese (Hiragana, Katakana)

Katarina Gevorgyan - Armenian
Kim Kyoungsok - Korean (Han, Hangul)
Kuldeep Patnaik - Oriya
Makara Sok - Khmer
Matitiahu Allouche - Hebrew
Michael Bauland - Latin
Naail Abdul Rahman - Thaana
Nabil Benamar - Arabic
Panagiotis Papaspiliopoulos - Greek
Parkpoom Tripatana - Thai
Pavanaja U B - Kannada
Saysomvang Souvannavong - Lao
Shanmugam Rajabadher - Tamil
Thin Zar Phyo - Myanmar
Uma Maheshwar Rao G - Telugu
Veena Solomon - Malayalam
Wang Wei - Chinese (Han)
Zakharia Pourtskhvanidze - Georgian

Contractor:

Akshat Joshi
Asmus Freytag
Michael Bauland

ICANN Staff:

Pitinan Kooarmornpatana
Sarmad Hussain