IMPLEMENTATION OF CONVERTING UNICODE TEXT TO ORIGINAL TEXT

Mr. Prashanth Kumar HM[1] & Dr. Subramanya Bhat S[2]
[1]Research Scholar, College of Computer Science, Srinivas University, Mangalore, India
[2] Professor, College of Computer Science, Srinivas University, Mangalore, India
E-mail: prashanth.hm02@gmail.com, itsbhat@gmail.com

ABSTRACT

In regional plagiarism checker initially, we must understand for character conversion, almost we have 3000+ regional conversion font available in market. More fonts may refer to Unicode character, UTF or encoding format. Other font won't create any abjection while converting but we may more concentrate on Unicode. Because most regional language text creation is under Unicode format. Unicode is an international standard for the representation of text and characters in computers and other devices. It maps characters to unique numerical values, called code points, allowing computers to store, process, and display text in a consistent way, regardless of the language or script in which it was written. Unicode supports a wide variety of languages, scripts, symbols, and special characters, including emoji, mathematical symbols, and bidirectional text. It also supports character encoding, allowing characters to be stored, processed, and displayed in a way that is compatible with different platforms and devices. In our implementation part Unicode conversion is an essential tool for text processing, storage, and display, making it easier and more consistent for the need to use text in different languages. Finally, it's challenging work for converting Unicode text to original format into the next level of plagiarism process.

Keywords: Unicode, Jumbled Text, Conversion, ASCII, Encoding, Interface, Index.

1. INTRODUCTION

Unicode is an international standard for the representation of text and characters in computers and other devices. It is designed to support the representation of all written characters used in the world, including those used in alphabets, scripts, and symbols. Unicode is also designed to be able to handle a wide variety of languages, scripts, and symbols, which makes it ideal for use in a globalized world. "At its core, Unicode is a mapping of characters to specific numerical values, called code points. Each code point is a unique number that represents a specific character, regardless of the language or script in which it is used."[1] This allows computers and other devices to store, process, and display text in a consistent way, regardless of the language or script in which it was written.

For example, consider the "characters 'A' and 'a'. In a traditional computer character set, such as

ASCII, these characters are represented by different numerical values. In Unicode, however, both 'A' and 'a' are assigned the same code point, allowing them to be treated as equivalent characters regardless of the language or script in which they are used." [2]

In addition to standard characters, Unicode also includes a wide variety of symbols, including emoji, mathematical symbols, and other special characters. This makes it ideal for use in a wide range of applications, including text processing, web development, and software development. It also includes several features that are designed to support the representation of text in different languages and scripts. For example, it includes support for right-to-left writing systems, such as Arabic, Hebrew, and Persian, as well as support for complex scripts, such as Devanagari, Thai, and Khmer. It also includes support for bidirectional text, which allows text in different scripts to be displayed and processed correctly, regardless of their direction. Another important feature of Unicode is its ability to handle multi-lingual text. "This means that Unicode can represent text in multiple languages, scripts, and symbols, even within the same document. This makes it ideal for use in applications that need to support multiple languages, such as multilingual websites, multilingual software, and multilingual databases." [5]

Unicode also includes support for combining characters, which allows multiple characters to be combined into a single character. For example, in Unicode, an accented letter, such as 'é', can be represented as a single character, rather than as a combination of a base letter and an accent character. This makes it easier to process text, as well as to display text consistently across different platforms and devices. Finally, Unicode includes support for character encoding, which allows characters to be represented in a way that can be stored, processed, and displayed by computers and other devices. Unicode includes several different encodings, including UTF-8, UTF-16, and UTF-32, each of which is designed for specific use cases. For example, "UTF-8 is designed for use in web development, while UTF-16 is designed for use in software development. It's a vital tool for the representation of text and characters in the modern world. Its ability to handle a wide variety of languages, scripts, and symbols, its support for right-to-left writing systems, complex scripts, bidirectional text, multi-lingual text, combining characters, and character encoding, make it ideal for use in a wide range of applications and contexts." [6] Whether you're a web developer, software developer, or just someone who needs to use text in different languages, Unicode is an essential tool that makes text processing, storage, and display easier and more consistent. Here UTF-8 is not created any issue during processing, but Unicode font is highly transformable to other method, here extraction data from user file to next level all text data is stored in jumbled text, this test useless for next processing level, so avoiding this type of problem for the valid plagiarism process, we should be extract to ascii value form jumbled text. The ascii only format to support human readable text other than Unicode and encoding methods.

2. OBJECTIVES

2.1 Standard Characters: Standard characters refer to a set of commonly used letters, numbers, and symbols that are standardized and widely recognized across different countries and cultures. "Standard characters are used in a variety of contexts, including writing systems, computer character sets, and encoding standards. These characters are typically defined by international standards organizations, such as Unicode, to ensure consistency and compatibility across different platforms and devices." [7] Examples of standard characters include the English alphabet, numerals, punctuation marks, and common symbols, such as @ and &. The use of standard characters makes it easier to process, store, and display text, as well as to communicate effectively in different languages and scripts.

2.2 Jumbled Text: Jumbled text refers to text that has been rearranged or mixed up, resulting in a confusing and difficult-to-read message. Jumbled text can occur due to various reasons, such as technical issues with a computer system, errors in software, or the deliberate rearrangement of text for effect. In some cases, jumbled text can be difficult to decipher and may require manual correction or the use of specialized software tools to restore it to its original form.

Ex: ¯ÉÃR£ÀUÀ¼ ÀÄ ª ÀÄÄÄvÁzÀª ÀÅUÀ¼ À£ ÀÄß v ÉUÉz ÀÄP ÉÆAqÀÄ,

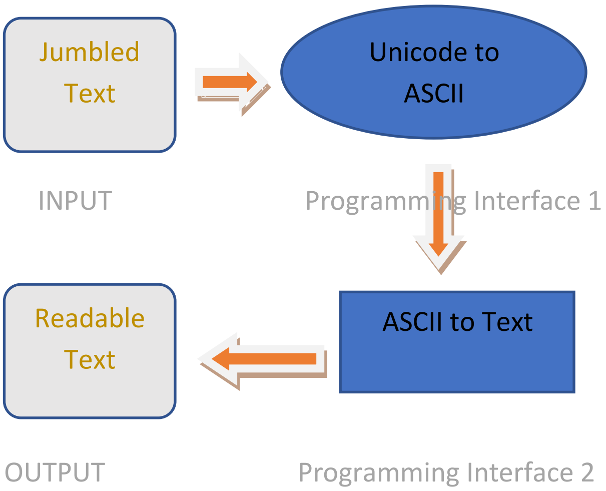
Jumbled text can be particularly problematic in fields where clear and accurate communication is essential, such as in scientific or legal documents, where even small errors or inaccuracies can have significant consequences.

2.3 Intermediate Operation: Intermediate operation refers to a processing step or operation that is performed during the processing of data in a computer system. Intermediate operations are typically performed between the input and output stages of a system and are used to manipulate, transform, or otherwise process data. The purpose of intermediate operations is to prepare the data for its intended use or to perform a specific function on the data, such as compression, encryption, or sorting. In software development, intermediate operations are often performed using libraries, frameworks, or APIs. For example, when processing large data sets, intermediate operations may be used to sort, filter, or aggregate data before it is written to disk or sent to an external system.

2.4 Encoding Basic: "For the standard ASCII (0-127) characters, the UTF-8 codes are identical. This makes UTF-8 ideal if backwards compatibility is required with existing ASCII text, in plagiarism majority of English language and English related languages support only here in UTF-8. Other characters require anywhere from 2-4 bytes. This is done by reserving some bits in each of these bytes to indicate that it is part of a multi-byte character. In particular, the first bit of each byte is 1 to avoid clashing with the ASCII characters." [2]

3. PROCESS

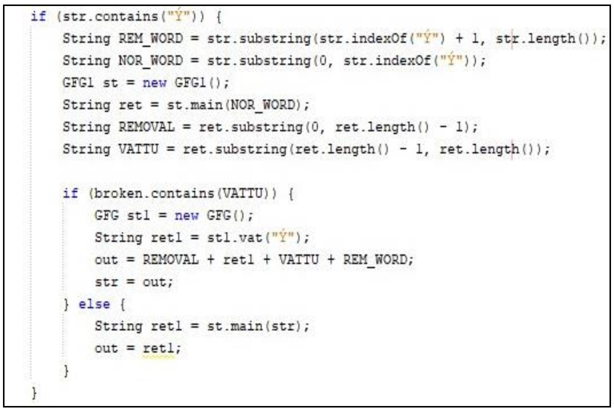
1. Unicode to ASCII Converter
2. ASCII to Text Converter



Unicode to ASCII converter is a programming language that converts Unicode characters to their equivalent ASCII characters. Unicode is a character encoding standard that includes a wide range of symbols, letters, and characters from different scripts and languages. ASCII, on the other hand, is an 8-bit character or 16-bit decimal character mainly used standard format conversion. The conversion from Unicode to ASCII is necessary when dealing with systems that only support ASCII and when you need to remove any non-ASCII characters from a text. In the next level ASCII to text converter is a programming language that converts ASCII characters to their equivalent text representation. "The conversion from ASCII to text is necessary when you want to display the content of an ASCII file or data stream in a human-readable form. The output of the conversion will be a plain text representation of the original ASCII characters, with each character being mapped to its equivalent textual representation." [8] Finally readable text refers to a text that is well-structured, easy to understand, and clear in its meaning. This can refer to written text in any format, including books, articles, emails, and more. Readable text is characterized by its clear and concise language, proper use of punctuation, and good organization of information. The goal of readable text is to make the information it contains accessible and easy to understand for the intended audience. This can be achieved through clear writing, use of appropriate font styles and sizes, and effective use of headings, bullet points, and other formatting tools.

4. IMPLEMENTATION

4.1 Characters Conversion: The purpose of character conversion is to ensure compatibility between systems and applications that use different encodings to represent text. Some common character encodings include ASCII, Unicode, UTF-8, and UTF-16.



```
if (str.contains("Ý")) {
    String REM_WORD = str.substring(str.indexOf("Ý") + 1, str.length());
    String NOR_WORD = str.substring(0, str.indexOf("Ý"));
    GFG1 st = new GFG1();
    String ret = st.main(NOR_WORD);
    String REMOVAL = ret.substring(0, ret.length() - 1);
    String VATTU = ret.substring(ret.length() - 1, ret.length());

    if (broken.contains(VATTU)) {
        GFG st1 = new GFG();
        String ret1 = st1.vat("Ý");
        out = REMOVAL + ret1 + VATTU + REM_WORD;
        str = out;
    } else {
        String ret1 = st.main(str);
        out = ret1;
    }
}
```

This implementation part contains input text to receive from user with selected language finally it finds relevant character to original. Above example special jumbled character 'Ỳ' (ASCII 237) is selected and taking relevant original character(ರ) from 'Kannada' languages with supporting indexed levels.

4.2 Executable Binding: Executable binding, also known as dynamic linking or run-time linking, is a process in computer programming where a compiled executable file is bound to libraries or modules at run-time, as opposed to linking them at compile-time. This means that the executable file does not include the code for the libraries it depends on. Instead, the libraries are loaded at run-time when the executable file is executed. This allows for more flexibility, as different versions of the libraries can be used without recompiling the executable file, and libraries can be updated without affecting the executable file.

Executable binding is commonly used in operating systems and other software that needs to run multiple programs or applications simultaneously. It helps to save memory and allows for efficient use of resources.

4.3 Intermediate APIs: APIs that sit between the client and the server in a client-server architecture. The primary purpose of intermediate APIs is to act as a bridge between the client and server, enabling communication between them and facilitating the exchange of data. Intermediate APIs perform various tasks, such as:

Data transformation: converting data from one format to another to make it compatible between the client and server.

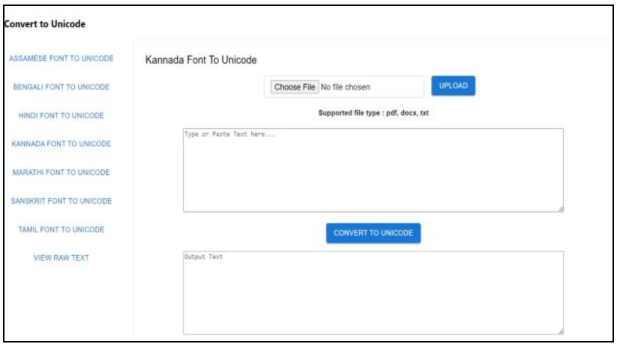
Security: providing security features such as authentication, authorization, and encryption to ensure the safety of data being transferred.

Caching: improving performance by temporarily storing data in memory so that it doesn't have to be fetched from the server every time it is needed.

Rate limiting: Controlling the rate at which the client can access the server to avoid overloading the server and ensure fair usage for all clients.

Intermediate APIs are commonly used in microservice architectures, where multiple services are deployed in a distributed environment, and in modern web and mobile applications to improve scalability and performance. In plagiarism process it's very useful to carry user data or file to backend level, or processed data (file) to user level.

4.4 User Interfaces: A user interface (UI) is a means by which a person interacts with a computer, mobile device, or other electronic device. It is the space where users interact with a system and complete tasks. The UI provides a visual representation of the device's functions and capabilities, and users interact with the UI using input devices such as a keyboard or mouse. Below image shows conversion of Kannada Unicode.



There are several types of user interfaces, including:

Graphical user interface (GUI): A visual interface that presents graphical elements such as windows, icons, and buttons. Command-line interface (CLI): A text-based interface where users enter commands to perform actions. Voice user interface (VUI): A UI that allows users to interact with a device using voice commands.

Web user interface (WUI): A UI that is accessed through a web browser.

The goal of a UI is to make the device's functions and capabilities easy to understand and use. In this process UI design takes into consideration the needs and preferences of the target users, as well as the devices and platforms the UI will run on. A well-designed UI can enhance the user experience and increase user satisfaction with a device or system.

5. ISSUES AND HANDLING

5.1 Lengthy Process: A conversion text lengthy process refers to a procedure, task, or activity that takes a long time to conversion and implementation part took many lengthy steps to complete. It may involve multiple steps, require significant resources or effort, or face obstacles that slow progress. These processes can be time-consuming and may require patience and persistence to complete successfully.

5.2 Text Ambiguity: During conversion text sometimes, software cannot assist similarity text, this creates text ambiguity, this refers to the existence of multiple possible meanings or interpretations of a given text. This can occur due to the use of vague or imprecise in same text language, multiple definitions of a characters, or the context in which the text is being read to convert. Text ambiguity can lead to confusion, misunderstanding, or misinterpretation of the intended meaning, which can have important consequences in various contexts.

5.3 Required Text Normalization: Text normalization is the process of transforming text into a standard or canonical form, to reduce variations, inconsistencies, and redundancies that can make text processing, analysis, and retrieval difficult. This process involves several techniques such as sometimes converting all text to lowercase, removing punctuation, expanding contractions, replacing abbreviations with their full forms. Text normalization is an important step in Unicode language processing and information retrieval tasks, as it enables more accurate and efficient analysis and comparison of textual data.

5.4 Lack of Data Compatibility: Data compatibility refers to the ability of different data systems or applications to work together effectively and efficiently, without causing conflicts, errors, or inconsistencies. This includes ensuring that data formats, structures, and definitions are compatible across different systems, as well as establishing protocols and standards for data exchange and sharing. Data compatibility is important for organizations that need to integrate and analyse data from different sources, such as in Unicode data warehousing, Unicode text relationship management. It also enables interoperability between different levels of conditional statements, platforms, and devices, allowing for seamless data communication and collaboration.

6. OTHER ADVANTAGES

6.1 Text formatting: Text formatting is an important element of communication and can be used to make text easier to read encoding characters, emphasize important information, or create a distinct brand identity. Sometimes same jumbled text refers from input fields, but destination should be referred by intermediated selected languages, it is depending on language level and indexing types.

6.2 Easy for indexing: Text indexing is the process of creating an index or catalog of the content of a collection of texts, to facilitate efficient searching, retrieval, and analysis. This process involves identifying the key concepts, terms, or entities that are important for describing the content of the texts, and creating an index that maps these concepts to the locations in the texts where they occur. This allows users to search for specific words or phrases and retrieve relevant texts quickly, rather than having to read through the entire collection. Text indexing is a fundamental technique in many information retrieval systems.

6.3 Core Indexing: In Unicode conversion the core indexing refers to the process of identifying and selecting the most important or relevant information from a larger set of data or information, to create a condensed and manageable representation of that data. This process involves choosing key characters that are considered essential for describing or understanding the relevant text data set, and assigning relevant keywords, descriptors, or codes to them, to facilitate searching, retrieval, and analysis.

CONCLUSION

The plagiarism checker works on text matching mechanisms; it finds the originality of data between student files and the internet or repository archives. Here, the development of plagiarism checkers in regional languages is the most complicated; there are many levels of challenging tasks we must resolve. In the list, Unicode conversion is one of the major tasks; we should solve this task in customer input file conversion, internet archive conversion, and indexed data conversion according to priority if text is available for Unicode conversion. In this section, we resolved our own intermediate format of Unicode to original text conversion. This is the process of converting jumbled text data to original text, which is required to understand the text between two sides of matching mechanisms.

REFERENCES

[1]. *The Unicode Consortium, The Unicode Standard,* Version 4.0, Reading, MA, Addison-Wesley, 2003.

[2]. "*ANSI X3.4: Coded Character Set-7-Bit American National Standard Code for Information Interchange*," New York, 1986.

[3]. *K.F. Lee, H.W. Hon and R. Reddy, "An overview of the SPHINX speech recognition system*", IEEE Transactions on Acoustics, Speech and Signal Processing, vol. 38, no. 1, pp. 35-45, January 1990.

[4]. *G. Brown and K. Woods, "Born Broken: Fonts and Information Loss in Legacy Digital Documents*," Int. J. Digit. Curation, vol. 6, no. 1, pp. 5–19, 2011 [Online]. Available: http://www.ijdc.net/index.php/ijdc/article/view/159. [Accessed: 31-Jan-2015]

[5]. *S. Young, 'The HTK hidden markov model toolkit: design and philosophy*," Cambridge University Engineering Department, UK, Tech. Rep. CUEDIF-lNFENG/TRI52, September 1994.

[6]. *G. S. Mahi, A. Verma, K. S. Bajwa, and G. Singh, "Information Loss in Digital Documents*," in IEEE international symposium on emerging trends and technologies in libraries and information services, 2015, pp. 118–121.

[7]. *A Lee, T. Kawahara and K. Shikano,"Julius - An open-source real-time large vocabulary recognition engine*", in Proc. 7th European Conference on Speech Communication and Technology (EUROSPEECH2001), Aalborg, Denmark, pp. 1691-1694, September 2001.

[8]. "*The Unicode Consortium."* [Online]. Available: www.unicode.org

[9]. *A. Bharati, N. Sangal, V. Chaitanya, R. Sangal, and G. U. M. Rao, "Generating Converters between Fonts Semi-automatically*," in SAARC conference on Multi-lingual and Multi-media Information Technology, 1998.

[10]. *V.K. Samaranayake, S.T. Nandasara, J.B. Dissanayake, AR. Weerasinghe and H. Wijayawardhana. (2003) "An introduction to UNICODE for Sinhala characters"*. Department of Sinhala University of Colombo, UCSC technical report 03/01.

[11]. *S. S. Rahaman, M. R. Islam, and M. a. H. Akhand, "Design and development of a Bengali unicode font converter,"* in 2013 International Conference on Informatics, Electronics and Vision (ICIEV), 2013.

[12]. *R. Arokia, A. Anand, and K. Prahallad, "Identification and Conversion on Font-Data in Indian Languages,"* in ICUDL, 2007.