Recovering Text from Endangered Languages Corrupted PDF documents

Nicolas Stefanovitch
Joint Research Centre
European Commission
Ispra, Italy
nicolas.stefanovitch@ec.europa.eu

Abstract

In this paper we present an approach to efficiently recover texts from corrupted documents of endangered languages. Textual resources for such languages are scarce, and sometimes the few available resources are corrupted PDF documents. Endangered languages are not supported by standard tools and present even the additional difficulties of not possessing any corpus availabel over which to train language models to assist with the recovery. The approach presented is able to fully recover born digital PDF documents with minimal effort, thereby helping the preservation effort of endangered languages, by extending the range of documents usable for corpus building.

1 Introduction

Endangered languages usually have extremely scarce linguistic resources available, and even less in a directly usable Unicode-encoded text format. Often the only available resources are PDF documents produced by language preservation efforts or for proselyte purposes by different organisations. Despite the efforts of such organisations to provide text content in endangered languages, it is often the case that the documents they produced are only printable or displayable on screen but totally unusable for automatic text processing purposes. Sometimes the only documents available in these language are contained in such documents, thereby preventing a wider exposure (impossibility to find them with search engines) and study of that language (impossibility to build corpora). The problem this paper tackles is how to recover usable texts from such documents.

There are two main drives behind PDF documents not being exploitable by Natural Language Processing (NLP) systems: 1) the document has a corrupted font to Unicode value translation table - in which case the text can not be extracted as the content of the pasted text is either unavailable or gibberish; 2) the PDF actually contains scanned images - and therefore it is impossible to copy/paste from that document. In this paper we tackle only the first, simpler, problem of born digital PDF document recovery, leaving the second more complex problem for future works.

The alternative of using our approach for such corrupted document is a much more time consuming manual correction of incorrectly OCRed text, or an even more time consuming plain manual transliteration into Unicode of the document content. As such it can be viewed as a lightning fast alternative to manual recovery. Such a work can help speedup corpus creation efforts for endangered languages such as in (Mus and Metzger, 2021).

Despite the general applicability of our approach, we will specifically restrict our attention to the Universal Declaration of Human Right corpus (UDRH), as it is - outside the Bible - the most translated documents whose documents are openly accessible on the internet (Cabatbat et al., 2014). It has to be noted that corrupted PDF documents for endangered languages is a common phenomena: there exists no support for these languages, and very often they posses unique symbols or even use their own writing system. In order to write in these languages the creators of the documents must design their own ad-hoc fonts, which are not publicly available. For all these reasons, the approach we propose in this article particularly relevant for recovering text in endangered languages.

2 Background

2.1 UDHR corpus

The United Nations maintains a website collecting all the different translations of the Universal Declaration of Human Right (UDHR)[1]. The UDHR corpus presents the specificity that most of its texts are present only as PDF documents, without any

[1] https://www.ohchr.org/EN/UDHR/

Proceedings of the Fifth Workshop on the Use of Computational Methods in the Study of Endangered Languages, pages 78 - 82

Unicode text. At the time of writing of this article, there are 526 translations in the corpus, some of which are the only text openly available in that language. Out of these, 108 (20%) have a content only in form of scanned images and 21 (4%) are corrupted born digital documents containing unrecoverable characters.

A complementary effort has been done by the Unicode Consortium which aims at collecting the Unicode version of the UDHR corpus[2]. As such, many of the documents of the UDHR corpus without extractable content do actually possess a text version in the Unicode consortium repository.

2.2 Document Recovery

Extracting text from documents containing only images can be done with standard Optical Character Recognition (OCR) tools only if the alphabet of the endangered languages is exactly the same as the alphabet of an existing well supported language. However, it is rarely the case, as most endangered languages possess very specific symbols absent from more widely used languages, moreover, such languages sometimes use their own writing system. Finally, as an OCR process is always noisy, a fully automatic text recovery requires to correct the errors by relying on language models (D'hondt et al., 2017). As such even Tesseract-OCR, one of the most popular OCR tool which covers about 100 languages out of the box, is not a workable solution for most endangered languages.

Because of the scarcity of texts in such languages, it is not even possible to correctly train OCR systems, as the only existing realisations of some languages' characters exist only in the Unicode charts [3], and therefore severely lack in diversity as only every Unicode letter has in these charts only one realisation with one font. As such, an OCR system trained solely on Unicode charts would lack the flexibility of dealing with different fonts and realisation of the characters. For these reasons OCR techniques present significant difficulties when dealing with endangered languages, and in this paper we will tackle only with the simpler problem of corrupted fonts in born digital PDF documents.

2.3 PDF documents

In order to understand the solution designed to tackle the problem, it is important to understand how PDF files are structured. PDF documents do not contain string of Unicode characters that could be directly copied, there is not even an understanding of words as a semantic unit of text (Bandara, 2020). A PDF document actually contains I) a list of fonts, and for each there is a) a mapping between a CID (Characterd IDentifier) and the symbol as a 2D bitmap (glyph), and b) a mapping from CID to Unicode value; II) a list of physical lines, which are themselves made of an ordered list of tuples (page, font, character, bounding box) describing where to to draw each symbols in the 2D coordinates of each pages, as given by the bounding box of that symbol.

Figure 1: Subset of a corrupted font for Nenets, visualised using FontForge

2.4 Problem Description

When a PDF document is corrupted, resulting in gibberish being produced when trying to convert the document to text or when trying to copy/paste from it, it is actually only the translation map from CID to Unicode that is corrupted. When using a specialised PDF to text translation tool, such as PDFMiner, characters absent from the map (the map can be only partially corrupted) are extracted as the string `cid:<n>` where `n` is the actual numerical value of the CID. As such all the realisation of a symbol will be linked to the exact same CID, and the task of recovering the document is equivalent to the simpler task of recovering the Unicode translation table.

In Figure 1 we report an example of a corrupted font encoding for Nenets taken from the UDHR corpus: The letters in gray are the Unicode letters associated to the symbol below them. While the dot, the dash and Latin capital letter I are correctly encoded, all the other letters are problematic: Most Cyrillic symbols are not associated with any Unicode character, while some of them are wrongly associated to the characters `1`, `2`, `(` and `)`. Note how both the glyph of the number `2` and of the Cyrillic letter *En with hook* are both translated to the same character.

In Figure 2 we present and instance of a text

[2]https://www.unicode.org/udhr/
[3]https://www.unicode.org/charts/

Қатыгун сик правоғун Декларация
Ниғвң иғр раюд

:àòüãóí ñèê ïðàâî4óí Äåêëàðàöèÿ
Ни4в2 и43 раюд

Figure 2: Title of the UDHR in Nivkh, as appearing in the rendered PDF document (top), and as appearing when extracted with a text extraction tool (bottom)

excerpt from a Nivkh document, showing the incorrect result produced by the text extraction tool.

Nevertheless, using extraction tools it is possible to extract correctly the document as a string of CIDs, instead of as a string of Unicode characters. Such a content lends itself to statistical analysis, where the frequency of CID and character ngrams could be used to recover the encoding. However, because of the general unavailability of language models for endangered languages this approach is not possible.

An inspiration to our approach is the work of (Vol et al., 2018), however their system is designed to process documents of undetermined language, while we know the language of the document we want to process; and it relies on OCR of the document for a subset of well supported languages, requiring extensive training material, while there are no such resources in our case as we deal with endangered languages.

Because of all the above constraints, there is no possibility to automatically recover Unicode translation maps for endangered languages. Consequently, human intervention is required, and as such the system we propose is designed to make the recovery process as fast and convenient as possible.

3 Proposed System

The system we designed is an interactive tool that allows the user to recover the text of corrupted document, requiring from a few minutes to a few hours depending on the quantity of symbols to recover, on the knowledge of the user of the target language, and on its ability to input the required characters.

Our system proceeds in two phases: a first fully automatic one that recovers the symbols for space and dot; and a second interactive one that helps the user gradually build the minimal quantity of resources in order to decipher the text.

3.1 Automatic Recovery

Because a font can be only partially corrupted it means that sometimes part of the text can be recovered: several letters as well as spaces and punctuation. However, it may happen that the font is corrupted in a way that the space and dot characters are wrongly attributed to other symbols. As such a font that contains some unattributed CID can not be trusted, and we proceed to the initialisation from scratch of the Unicode translation map, meaning that all documents and all languages are treated the same.

In the automatic phase, the system heuristically recovers which CID corresponds to the space and the dot characters. The space character is determined as the CID which is appears on the most lines, the dot character is determined as the CID that appears the most frequently at the end of a line which ends not at the margin, but specifically between 20% and 80% of the right margin. The left and right margin are determined by the leftmost and rightmost position of a character of that font. Other heuristics to recover the space were tried, such as the most frequent character that do not appear at lines end/beginning, but they did not work consistently and were disregarded.

Because of that reliance on statistics this automatic recovery can not work on very short texts, in which case it can be deactivated and the recovery can proceed only with the interactive phase. However, in case enough text is available to compute the statistics (a few lines), it proves a major time-saver.

3.2 Interactive Recovery

During the interactive phase the user is asked to prompt the system in several ways text as he sees it anywhere in the text. This can be either tokens, in that understanding it is any character sequence separated by spaces, or token sequences. From there, the system automatically tries to match the Unicode sequences that the user inputs to the CID sequences that are extracted from the PDF. Initially the translation map is void but it is filled progressively until there remains no character to be decoded.

If the user inputs a single token, the system will build a regular expression for all the CID sequences of the same length, and substituting the already decoded CID with their Unicode translation. If there is an unique match, the system therefore infers the value of the previously undecoded CID by matching them one by one to the characters entered by

n	1	2	3	4	5	6
niv	10	33	72	93	97	98
yrk	17	29	60	86	98	100

Table 1: Proportion in percent of unique sequences of token lengths for sequence length n in the UDHR for two different languages

language	niv	yrk
unique characters	81	68
text length (words)	1430	1530
input length (words)	57	76

Table 2: Unique character count and total word length for the UDHR declaration in two languages, and the total number of input words necessary for full recovery of these texts

the user. If there is a contradiction with a previously learned translation, the system flags it and the user must review the error and correct its inputs.

Entering a single token is however ineffective when starting the deciphering of a document because of the potentially high number of CID sequence that are of the same size as the token. As such, ii is in practice useful only at latter stages when most of the characters have already been recovered. A much more precise way of matching CID sequences to Unicode sequence is needed.

This problem is solved by letting the user input sequences of consecutive tokens, appearing anywhere on any lines, giving therefore much flexibility to the user. While a token length is an imprecise way of retrieving text, a sequence of token lengths is much more likely to be unique. For non unique sequences it is nevertheless possible to cue to the system its line number. With this minimal information the system can find the correct CID sequence an decipher it in the same fashion as previously.

In Table 1 we report on the uniqueness of such sequences of token length for two languages. For the two illustrated languages, if the user inputs a sequence of 5 words there is at least 97% chance that this sequence is unique, and therefore that all the corresponding letters will be correctly decoded.

In order to help the user, the system determines which CID are the most frequent, and on which lines their are the most unrecovered CID. By using this information the user can reduce to the minimum the number of words he has to input to the system in order to guarantee a full recovery. It also saves considerable time to the user, as this one does not have to search manually through the document for unrecovered characters.

4 Experiments

In the experiments we consider only corrupted documents that do not have an Unicode version even on the Unicode Consortium website. Out of the dozen such documents, we focus our effort on four languages in order to demonstrate the capacity of our approach: Nenets (iso code: yrk, in Cyrillic script), Nivkh (also called Gyliak, iso code: niv - actually the Sakhalin island dialect (Gruzdeva, 2022), in Cyrillic script) is a language isolate spoken by only a few thousands people, Mundari (iso code: unr, in Devanagari script) and a Mongolian dialect (iso code: mvf, in Mongolian script). While there exists significant resources for Mongolian in Cyrillic script, it is not the case for Mongolian script, which is used only to write dialects spoken in China, moreover their difference makes it impossible to exploit transliteration in order to ease the recovery process. The Nivkh language has the particularity that one letter of its alphabet is not even present in Unicode and can be realised only through combining characters. Because of the lack of support for these languages in latex it is impossible to give concrete examples of the rules used when recovering the texts.

In Table 2 we report statistics on the documents: number of unique symbols, number of words; and statistics on the user input: number of words entered in the system. The number of words required by the system is between 4 to 5 % of the total, the number of letters actually input by the user is actually significantly less, as during the text recovery process, it is possible to copy/paste partially recovered sequences of text and only replace the unavailable CID with the correct characters. As such, our approach makes it very efficient to recover the text by requiring to input only a fraction of the original text in order to recover it.

5 Discussion

Our approach allowed us to quickly and efficiently recover the UDHR Unicode text for two languages, requiring less than an hour of work: Nivkh and Nenets. These two documents have been sent to the UDHR page of the Unicode consortium, and are now already publicly available.

One additional advantage of our approach, is that

when dealing with a document collection using the same corrupted font, it is necessary to recover it only once in order to process the other documents, thereby yielding additional time gains for linguists and experts striving to create corpora.

When recovering Mongolian, we have been confronted to the problem that this language is written top down, because PDF documents consider that lines are going exclusively from left to right or right to left, the text extraction tool is totally unable to recover the lines. Consequently, our method can not be applied directly for that language, and instead of relying on an external library, it requires a further ad-hoc vertical segmentation step. This is left for future work.

At the time of the writing, another document is in the process of being recovered: the UDHR in Mundari, which is written in Devanagari script. Devanagari presents one specific difficulty: the long vowel "i" is written in a Unicode text string after the character of the consonant it is attached to, but it is displayed before it when the string is visually rendered. This is because the CID sequence of the symbols of a line is extracted in increasing order of the bounding box coordinates of the characters it contains. In order to deal with that, the user is constrained to input the characters in the same order as the one expected by the CID sequence, and a post-processing step is required to swap the corresponding Unicode characters before rendering the final text.

6 Conclusion

We present an approach that is able to quickly guide a human expert in recovering text from corrupted born digital PDF documents containing text in rare or endangered languages. Such languages impose severe constraints, because often there exists no preexisting corpus to train on, or to compare to the extracted text. Our approach has been designed specifically to operates within these constraints and consists in reconstructing the CID to Unicode translation maps by efficiently leveraging user input in an interactive way. We applied this approach to 4 documents of the UDHR corpus for which there exists no Unicode text, 2 of which where fully recovered, the other ones needing some additional development related to particularities of the writing system they use. The tool is not yet available, but will be released on this repository[4].

Future work will deal in applying the approach to more languages of the UDHR, and to deal with the harder problem of recovering text of endangered language existing solely as pictures. To that intent, we consider exploring image augmentation techniques (Minaee et al., 2021) in order to train ad-hoc OCR system for any scripts solely based on the symbols present in the Unicode charts, with the aim of interactively presenting the user with potential choices.

Our approach is useful to help protect and study endangered languages for which document base exists in born digital PDF format, but for which some of documents, or all of them, are corrupted.

References

RMCV Bandara. 2020. *Content extraction from PDF invoices on business document archives*. Ph.D. thesis.

Josephine Jill T Cabatbat, Jica P Monsanto, and Giovanni A Tapang. 2014. Preserved network metrics across translated texts. *International Journal of Modern Physics C*, 25(02):1350092.

Eva D'hondt, Cyril Grouin, and Brigitte Grau. 2017. Generating a training corpus for ocr post-correction using encoder-decoder model. In *Proceedings of the Eighth International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1006–1014.

Ekaterina Gruzdeva. 2022. On the diversification of nivkh varieties. In *The 4th Annual Meeting of Japan*.

Shervin Minaee, Yuri Y Boykov, Fatih Porikli, Antonio J Plaza, Nasser Kehtarnavaz, and Demetri Terzopoulos. 2021. Image segmentation using deep learning: A survey. *IEEE transactions on pattern analysis and machine intelligence*.

Nikolett Mus and Réka Metzger. 2021. Toward a corpus of tundra nenets: stages and challenges in building a corpus. In *Proceedings of the Workshop on Computational Methods for Endangered Languages*, volume 2, pages 4–9.

Mark Vol, Andrey Krutko, Nicolas Stefanovitch, and Denis Postanogov. 2018. Automatic recovery of corrupted font encoding in pdf documents using cnn-based symbol recognition with language model. In *2018 13th IAPR International Workshop on Document Analysis Systems (DAS)*, pages 121–126. IEEE.

[4] https://github.com/nicolasst