Journal of Data Mining and Digital Humanities

Evaluating Deep Learning Methods for Tokenization of Space-less texts in Old French

Thibault Clérice[1]

[1]École nationale des Chartes, France
[2]Université Lyon 3, France

Corresponding author: Thibault Clérice , thibault.clerice@chartes.psl.eu

Abstract

Tokenization of modern and old Western European languages seems to be fairly simple as it stands on the presence mostly of markers such as spaces and punctuation. Although, when dealing with old sources like manuscript written in *scripta continua*, (1) such markers are mostly absent, (2) spelling variation and rich morphology makes dictionary based approaches difficult. We show that applying convolutional encoding to characters followed by linear categorization to word-boundary or in-word-sequence can be used to tokenize such inputs. Additionally, we release a software with a rather simple interface for tokenizing one's corpus.

Keywords

convolutional network; scripta continua; tokenization; Old French; word segmentation

I INTRODUCTION

Tokenization of space-less strings is a task that is specifically difficult for computer when compared to "whathumancando". *Scripta continua* is a writing phenomenon where words would not be separated by spaces and it appears to have disappeared around the 8th century (see **?**). Never the less, spacing can be somewhat erratic in later centuries writings, as show by Figure 1, a document from the 13th century. In the context of text mining of HTR or OCR output, lemmatization and tokenization of medieval western languages can be a pre-processing step for further research to sustain analyses such as authorship attribution **CITE JBCAMPS ?**.

We must stress in this study that the difficulty that we face is different for *scripta continua* than

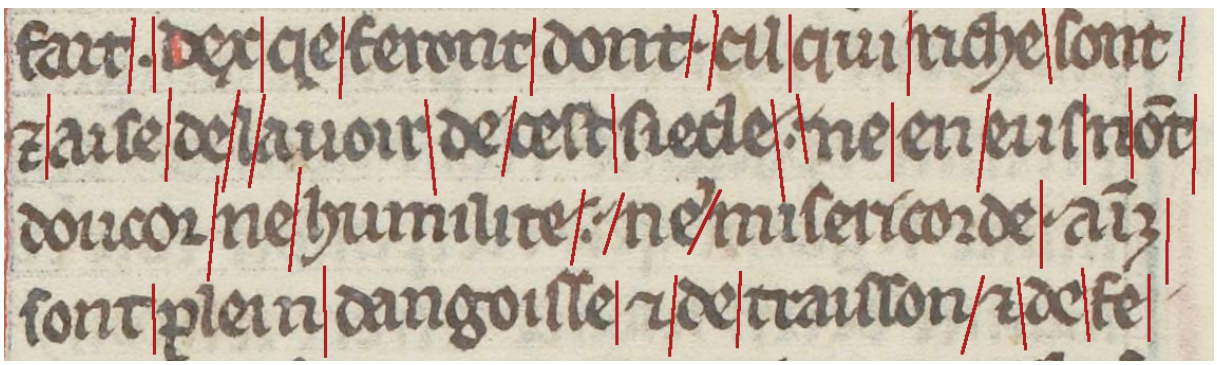

Figure 1: 4 lines from fol.103rb Manuscript fr. 412, Bibliothèque nationale de France. Red lines indicate word boundaries

	Sample
Input String	`Ladamehaitees'enparti`
Mask String	`xSxxxSxxxxxSxxxSxxxxS`
Output String	`La dame haitee s'en parti`

Table 1: Input, mask and human-readable output generated by the model. x are WC and S are WB

the ones researchers face languages such as Chinese for which an already impressive amount of work has been done as it. Indeed, Chinese word segmentation has lately been driven by deep learning methods, specifically ones based on *sequence to sequence translations*: **?** defines a process based on LSTM model, while **?** uses BiDirectional GRU and CRF. Actually, meanwhile redacting this article and producing the code-base, **?** took the same approach of encoding to linear classification to both word boundary (WB) and word content (WC) for Chinese word segmentation.

Indeed, while Chinese's issue seems to lie in the decomposition of relatively fix characters, Old French or medieval latin present heavy variation of spelling. In **?**, Camps notes, in the same corpus, the existence of not less than 29 spelling of the word *cheval* (horse in Old and Modern French) whose apparition counts span from 3907 to 1[1]. This makes a dictionary approach rather difficult as it would rely on a high number of different spelling and makes the computation highly complex.

II DESCRIPTION AND EVALUATION

2.1 Architecture

2.1.1 Encoding of input and decoding

The model is based on traditional text input encoding where each character is transcoded to an index. Output of the model is a mask that needs to be applied to the input: in the mask, characters are classified either as word boundary or word content (*cf.* Table 1.

For evaluation purposes, and to reduce the number of input classes, we propose two options for data transcoding: a lower-case normalization and a "reduction to the ASCII character set" feature (fr. 2). On this point, a lot of issues were found with transliteration of medieval paelographic characters that were part of the original datasets, as they are badly interpreted by the `unidecode` python package. Indeed, `unidecode` will simply remove characters it does not understand. I built a secondary package named `mufidecode` which precedes unidecode equivalency tables when the data is known of the Medieval Unicode Font Initiative (MUFI, **?**).

2.1.2 Model

Aside from normalizations of the input and output, three different structure of models were tested. Every model is composed by one encoder described below and one Linear Classifier which classifies into 5 classes : Start of Sentence (= SOS), End of Sentence (= EOS), Padding (= PAD), Masked Token (= Word Content), Space (= Word Boundary). For final scores, SOS, EOS and PAD were ignored.

The encoders are the following (configurations in parenthesis):

[1]These are *cheval*, *chevaus*, *cheual*, *ceval*, *chevals*, *cevaus*, *chival*, *ceual*, *cheuaus*, *cevals*, *chaval*, *chivaus*, *chiual*, *chevas*, *cheuals*, *chiuaus*, *ceuaus*, *chevaul*, *chiuau*, *chivals*, *chevau*, *kevaus*, *chavaus*, *cheuas*, *keval*, *cheua*, *cheuau*, *cheva*, *chiuals*

```
import mufidecode
import unidecode
"sot la gn̑t abstinence dess eintes uirges ele  p̃la"
mufidecode.mufidecode(" sot la gn̑t abstinence dess eintes uirges ele  p̃la")
# ' sot la gnat abstinence dess eintes uirges ele  pla'
mufidecode.mufidecode(" sot la gn̑t abstinence dess eintes uirges ele  p̃la", join=False
# (' ',  's',  'o',  't',  ' ',  'l',  'a',  ' ',  'g',  'n',  'a',  't',  ' ',  'a',  'b',  's',  't',  'i',
'n',  'e',  'n',  'c',  'e',  ' ',  'd',  'e',  's',  's',  ' ',  'e',  'i',  'n',  't',  'e',  's',  ' ',
'u',  'i',  'r',  'g',  'e',  's',  ' ',  'e',  'l',  'e',  ' ',  ' ',  'p',  'l',  'a')
unidecode.unidecode(" sot la gn̑t abstinence dess eintes uirges ele  p̃la")
# ' sot la gnat abstinence dess eintes uirges ele  la'
```

Figure 2: Different possibilities of pre-processing. The option with join=False was kept, as it keeps abbreviation marked as single characters. Note how `unidecode` loses the P WITH BAR

	Train dataset	Dev dataset	Test dataset
TIRONIAN SIGN ET	4367	541	539
CON	508	70	76
P WITH STROKE THROUGH DESCENDER	580	69	84

Table 2: Examples of some MUFI characters distributions

- LSTM encoder with hidden cell (Embedding (512), Dropout(0.5), Hidden Dimension (512), Layers(10))
- Convolutional (CNN) encoder with position embeddings (Embedding (256), Embedding(Maximum Sentence Size=150), Kernel Size (5), Dropout(0.25), Layers (10))
- Convolutional (CNN) encoder without position embeddings (Embedding (256), Kernel Size (5), Dropout(0.25), Layers (10))

2.2 Evaluation

2.2.1 Datasets

Datasets are transcription from manuscripts with unresolved abbreviation coming from different projects

- **Old French** based on **?**, **?**, **?**, **?**, and transcription from the TNAH master at the École Nationale des Chartes.
  - 193,734 training examples;
  - 23,581 validation examples;
  - 25,512 test examples

The input was generated by grouping at least 2 words and a maximum of 8 words together per sample. On a probability of 0.2, noise character could be added (noise character was set to DOT ('.')) and some words were kept randomly from a sample to another on a probability of 0.3 and a maximum number of word kept of 1. If a minimum size of 7 characters was not met in the input sample, another word would be added to the chain. A maximum input size of 100 was kept. The results corpora should be varied in sizes as shown by 3. The corpora is composed by 193 different characters when not normalized, in which some MUFI characters appears few hundred times 2.

2.2.2 Results

2.2.3 Example of outputs

The following input was not seen during training, dev or test, and is part of a different corpus :

- `Truth` : Aies joie et leesce en ton cuer car tu auras une fille qui aura .i. fil qui sera de molt grant merite devant Dieu et de grant los entre les homes.
- `Input` : Aiesjoieetleesceentoncuercartuaurasunefillequiaura.i.filquiserademoltgrantmeritedevantDieu
- `CNN` : Aiesjoie et leesce en ton cuer car tu auras une fille qui aura . i . fil qui sera de molt grant merite devant Dieu et de grant los entre les homes .
- `CNN lower`: Aies joie et leesce en ton cuer car tu auras une fille qui aura . i . fil qui sera de molt grant merite devant Dieu et de grant los entre les homes .
- `CNN without position`: Aiesjoie et leesce en ton cuer car tu auras une fille qui aura . i . fil qui sera de molt grant merite devant Dieu et de grant los entre les homes .
- `CNN Normalize`: Aies joie et leesce en ton cuer car tu auras une fille qui aura . i . fil qui sera de molt grant merite devant Dieu et de grant los entre les homes .

2.3 Discussion

We believe that, aside from a graphical challenge, word segmentation in OCR from manuscripts can actually be treated from a text point of view

2.4 Conclusion

While

2.5 Acknowledgement

Boudams has been made possible by two open-source repositories from which I learned and copied bits of implementation of certain modules and without which none of this paper would have been possible: **?** and **?**. This tool was originally intended for post-processing OCR for the presentation **?** at DH2019 in Utrecht.

A ANNEX 1

Pellentesque dignissim ultrices fringilla. Vivamus eu luctus ante, vel bibendum magna. Curabitur elit purus, tincidunt non dui vitae, elementum bibendum neque. Curabitur ullamcorper sit amet justo at hendrerit. Fusce ut arcu imperdiet nibh mollis tempus a aliquet tellus. Quisque pharetra cursus nisi, vel lobortis ante consectetur et. Vivamus sed congue neque. Proin pellentesque risus nec dui consequat rutrum. Vestibulum nunc diam, placerat quis auctor vel, faucibus non justo. Etiam dictum purus neque. Phasellus imperdiet mauris ligula, eu laoreet nisi elementum ut. Sed sed porta massa. Aenean faucibus risus ultrices ornare porta. Quisque faucibus ante a tincidunt vestibulum. Lorem ipsum dolor sit amet, consectetur adipiscing elit.

B ANNEX 2

Cras tristique vel nisi at aliquet. Proin egestas erat sit amet velit lobortis imperdiet. Integer et arcu sapien. Etiam id blandit sapien. Nam tempus lacus ac massa semper, vel laoreet turpis rutrum. Mauris eget nibh vitae justo porta imperdiet sed vel ligula. In imperdiet, augue vel condimentum convallis, neque augue imperdiet neque, eget dapibus nunc mauris ultricies tortor. Nam eget nunc egestas, blandit lectus non, aliquam nunc. Cras sed quam vitae arcu ornare lobortis. Ut ut lacus hendrerit, convallis orci sit amet, commodo nunc. Pellentesque eget tincidunt tortor. Nunc ornare molestie mauris id vehicula. Suspendisse pharetra tortor metus, sit amet fermentum tellus vehicula ut.

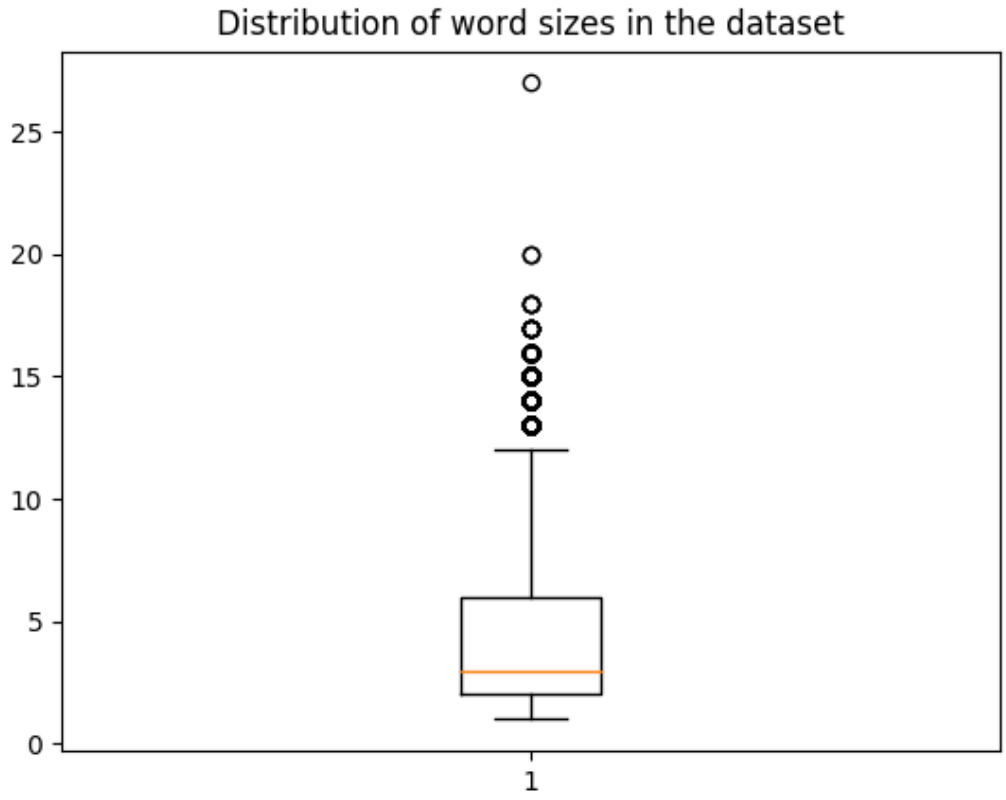

Figure 3: Distribution of word size over the train, dev and test corpora

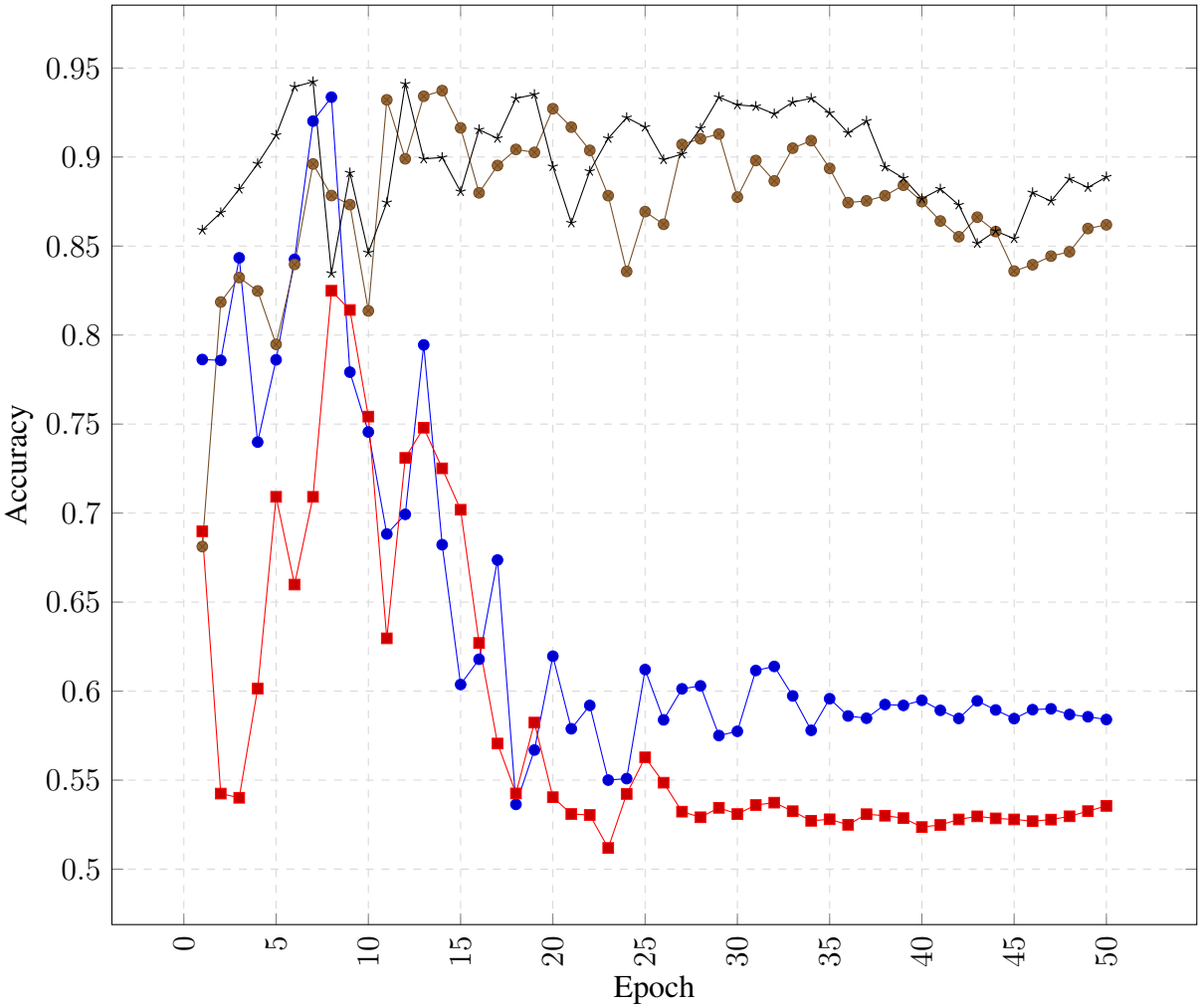

Figure 4: Training Accuracy over 50 epochs