



HAL
open science

Renard: A Modular Pipeline for Extracting Character Networks from Narrative Texts

Arthur Amalvy, Vincent Labatut, Richard Dufour

► **To cite this version:**

Arthur Amalvy, Vincent Labatut, Richard Dufour. Renard: A Modular Pipeline for Extracting Character Networks from Narrative Texts. 2024. hal-04611122

HAL Id: hal-04611122

<https://hal.science/hal-04611122>

Preprint submitted on 13 Jun 2024

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.



Distributed under a Creative Commons Attribution 4.0 International License

Renard: A Modular Pipeline for Extracting Character Networks from Narrative Texts

Arthur Amalvy¹, Vincent Labatut¹, and Richard Dufour²

1 Laboratoire Information d'Avignon 2 Laboratoire des Sciences du Numérique de Nantes

DOI: [N/A](#)

Software

- [Review](#) ↗
- [Repository](#) ↗
- [Archive](#) ↗

Editor: [Open Journals](#) ↗

Reviewers:

- [@openjournals](#)

Submitted: 01 January 1970

Published: 01 January 1970

License

Authors of papers retain copyright and release the work under a Creative Commons Attribution 4.0 International License ([CC BY 4.0](#)).

Summary

Renard (*Relationships Extraction from NARrative Documents*) is a Python library that allows to define modular natural language processing (NLP) pipelines to extract character networks from narrative texts. Contrarily to the few existing tools, Renard can extract *static* as well as *dynamic* networks. Renard pipelines are modular: the user can choose the implementation of each NLP subtask needed to extract a character network. This allows to specialize pipelines to particular types of texts and to study the impact of each subtask on the extracted network.

Statement of Need

Character networks (that is, graphs where nodes represent characters and edges represent their relationships) extracted from narrative texts are useful in a number of applications, from visualization to literary analysis (Labatut & Bost, 2019). There are different ways of modeling relationships (co-occurrences, conversations, actions...), and networks can be static or dynamic, which allows extracting different kinds of networks depending on the targeted applications. While some authors extract these networks by relying on manually annotated data Park, Kim, & Cho (2013), it is a time-costly endeavor, and the fully automatic extraction of these networks is therefore of interest. Unfortunately, there are only a few existing software and tools that can extract character networks (Marazzato & Sparavigna, 2014; Métrailler, 2023), and none of these can output dynamic networks. Furthermore, automatically extracting a character network requires solving several successive natural language processing tasks, such as named entity recognition (NER) or coreference resolution, and algorithms carrying these tasks are bound to make errors. To our knowledge, the cascading impact of these errors on the quality of the extracted networks has yet to be studied extensively. This is an important issue since knowing which tasks have more influence on the extracted networks would allow prioritizing research efforts.

Renard is a fully configurable pipeline that can extract static and dynamic networks from narrative texts. We design it so that it is as modular as possible, which allows the user to select the implementation of each extraction step as needed. This has several advantages:

1. The pipeline can be specialized for a specific type of texts, allowing for better performance.
2. The pipeline can easily incorporate new advances in NLP, by simply implementing a new step when necessary.
3. One can study the impact of the performance of each step on the quality of the extracted networks.

Design and Main Features

Renard is centered about the concept of *pipeline*. In Renard, a pipeline is a series of sequential *steps* that are run one after the other in order to extract a character network from a text. When using Renard, the user simply *describes* this pipeline in Python by specifying this series of steps, and can apply it to different texts afterwards.

```
from renard.pipeline import Pipeline
from renard.pipeline.tokenization import NLTKTokenizer
from renard.pipeline.ner import NLTKNamedEntityRecognizer
from renard.pipeline.character_unification import GraphRulesCharacterUnifier
from renard.pipeline.graph_extraction import CoOccurrencesGraphExtractor

with open("./my_doc.txt") as f:
    text = f.read()

pipeline = Pipeline(
    [
        NLTKTokenizer(),
        NLTKNamedEntityRecognizer(),
        GraphRulesCharacterUnifier(min_appearance=10),
        CoOccurrencesGraphExtractor(co_occurrences_dist=25)
    ]
)

out = pipeline(text)
```

To allow for custom needs, Renard is designed to be very flexible. If a step is not available in Renard, we encourage users to either:

- Externally perform the computation corresponding to the desired step, and inject the results back into the pipeline,
- Integrate their custom processing into Renard by implementing their own step by subclassing the existing `PipelineStep` class.

The flexibility of this approach introduces the possibility of creating invalid pipelines because steps often require information computed by previously ran steps: for example, solving the NER task requires a tokenized version of the input text. To counteract this issue, each step therefore declares its requirements and the new information it produces, which allows Renard to check whether a pipeline is valid, and to explain at runtime to the user why it may not be.

Task	Step	Supported Languages
Preprocessing	CustomSubstitutionPreprocessor	any
Tokenization	NLTKTokenizer	eng, ces, dan, nld, est, fin, fra, deu, ell, ita, nor, pol, por, rus, slv, spa, swe, tur
Quote Detection	QuoteDetector	any
NER	NLTKNamedEntityRecognizer	eng, rus
	BertNamedEntityRecognizer	eng, fra
Coreference Resolution	BertCoreferenceResolver	eng
	SpacyCoreferenceResolver	eng
Character Unification	NaiveCharacterUnifier	any
Character Unification	GraphRulesCharacterUnifier	eng, fra

Task	Step	Supported Languages
Speaker Attribution	BertSpeakerDetector	
Graph Extraction	CoOccurrencesGraphExtractor	any
	ConversationalGraphExtractor	any

Renard lets the user select the targeted language of its custom pipeline. A pipeline can be configured to run in any language, as long as each of its steps supports it. Table shows the supported languages for all the available steps in Renard.

References

- Labatut, V., & Bost, X. (2019). Extraction and analysis of fictional character networks : A survey. *ACM Computing Surveys*, 52, 89. <https://doi.org/10.1145/3344548>
- Marazzato, R., & Sparavigna, A. C. (2014). Extracting networks of characters and places from written works with CHAPLIN. *arXiv, cs.CY*, 1402.4259.
- Métraiiller, C. (2023). *Charnetto*. https://gitlab.com/maned_wolf/charnetto
- Park, G., Kim, S., & Cho, H. (2013). Structural analysis on social network constructed from characters in literature texts. *Journal of Computers*, 8. <https://doi.org/10.4304/jcp.8.9.2442-2447>
- Park, G., Kim, S., Hwang, H., & Cho, H. (2013). Complex system analysis of social networks extracted from literary fictions. *International Journal of Machine Learning and Computing*, 107–111. <https://doi.org/10.7763/IJMLC.2013.V3.282>
- Rochat, Yannick. (2014). *Character networks and centrality* [PhD thesis, Université de Lausanne]. https://serval.unil.ch/resource/serval:BIB_663137B68131.P001/REF.pdf
- Rochat, Y. (2015). Character network analysis of émile zola's les rougon-macquart. *Digital Humanities 2015*. <https://infoscience.epfl.ch/record/210573?ln=en>
- Rochat, Y., & Triclot, M. (2017). Les réseaux de personnages de science-fiction : Échantillons de lectures intermédiaires. *ReS Futurae*, 10, 1183. <https://doi.org/10.4000/resf.1183>