Electronic Communications of the EASST
Volume 85 Year 2025

deRSE25 - Selected Contributions of the 5th Conference for Research Software Engineering in Germany

Edited by: René Caspart, Florian Goth, Oliver Karras, Jan Linxweiler, Florian Thiery, Joachim Wuttke

Infrastructures for a Community-Developed Text Processing Library

Florian Barth, George Dogaru, Tillmann Dönicke, Mathias Göbel

DOI: 10.14279/eceasst.v85.2684

Electronic Communications of the EASST (https://eceasst.org).
Published by **Berlin Universities Publishing**
(https://www.berlin-universities-publishing.de/)

Infrastructures for a Community-Developed Text Processing Library

Florian Barth[1], **George Dogaru**[2], **Tillmann Dönicke**[1], **Mathias Göbel**[1]

Göttingen State and University Library (SUB)[1],
Gesellschaft für wissenschaftliche Datenverarbeitung mbH Göttingen (GWDG)[2]

Abstract: This paper introduces MONAPipe, a modular natural language processing (NLP) pipeline developed within the German National Research Data Infrastructure (NFDI) and coordinated by the Göttingen State and University Library (SUB). Designed to support a wide range of text-based research disciplines, including literary studies, digital humanities, and (computational) linguistics, MONAPipe integrates both general-purpose NLP tools and community-developed components into a flexible, spaCy-based framework. MONAPipe supports reproducible and sustainable research through comprehensive documentation, versioned resource management via long-term repositories like GRO.data, and automated testing and deployment pipelines. Docker-based containerisation allows for scalable deployment on both local machines and high-performance computing infrastructures, with components being accessible via REST APIs. The pipeline is available as an installable Python package, complemented by example workflows and training materials. Future developments may include integration into the Jupyter4NFDI environment, online service provision via the KISSKI HPC infrastructure, as well as a graphical user interface.

Keywords: Natural Language Processing, Digital Humanities, High Performance Computing, Containerisation, National Research Data Infrastructure (NFDI)

1 Concept

Within the German National Research Data Infrastructure (NFDI), the NLP pipeline framework MONAPipe ("Modes of Narration and Attribution Pipeline", [DBVS22, BBT$^+$23]) is jointly developed by participating institutions under the coordination of the Göttingen State and University Library (SUB). MONAPipe is a standalone Python library[1] based on spaCy[2] [HMLB20]. The focus of the pipeline lies on integrating developments from several disciplines, such as literary studies, digital humanities, (computational) linguistics, and other text-based research areas. MONAPipe integrates these developments as custom components in an end-to-end pipeline based on spaCy, makes them accessible to a wide range of users, and guarantees a long-term durability.

The development of MONAPipe has been guided by use cases and user stories gathered within the NFDI consortium Text+.[3] The user stories in the area of digital collections and editions are diverse and range from individual projects, e.g. the enrichment of collections of poems by Heinrich von Kleist and Andreas Gryphius,[4] to the curation of larger data sets, like the Tomsk Toponyme Archive.[5] They also include demands for generic[6] and dedicated NLP services, such as a service for linking named entities.[7] MONAPipe integrates and deploys NLP developments from the community to address demands for a more flexible NLP infrastructure. Often, use cases from the DH community have specific requirements, e.g. aforementioned user story 352 describes in detail the need to attach the output of NLP tools to existing annotation layers—a task that currently can't be addressed by tools provided by the Clarin switchboard such as Weblicht. To solve such individual use cases,

[1] https://pypi.org/project/monapipe/
[2] https://spacy.io/
[3] https://text-plus.org/en/themen-dokumentation/user-storys-2020/
[4] User story 352, Claus Zinn: Collections of Heinrich von Kleist and Andreas Gryphius, https://www.text-plus.org/en/research-data/user-story-352/.
[5] User story 329, Timm Lehmberg: Routines and best practice for the curation/integration of research data coming from non-european affiliations, https://www.text-plus.org/en/research-data/user-story-329/.
[6] User story 344, Torsten Roeder: Generic services for digital editions offer great chances for small projects, https://www.text-plus.org/en/research-data/user-story-344/; Computational Linguistics Group, University of Konstanz: Towards improved research results in computational linguistics via Text+ https://www.text-plus.org/en/research-data/user-story-314/.
[7] User story 331, Daniel Burckhardt, Jana Keck: Named Entity Linking Service to enrich Textual Collections, https://www.text-plus.org/en/research-data/user-story-331/.

a more customisable NLP infrastructure is needed that also allows direct integration of user demands by the users themselves. With a Python library driven by spaCy's custom component functionality and integrated in a Jupyter environment, we believe to deliver a more versatile NLP environment that can be used, customised, and extended by researchers from community or developers from different institutions to solve current research questions as well as institutional tasks like the enrichment of large datasets in digital libraries. A typical task in digital humanities and GLAM[8] institutions is the detection of named entities and their linking to authority files like GND or knowledge bases like Wikidata—for this, an example workflow is shown in section 5.

MONAPipe is supported by rich documentation and guidance materials tailored to different user groups. For researchers and general users, the focus lies on accessibility and ease of use, including a simple pip installation and a complete overview of components, their usage, and attributes.[9] Furthermore, example workflows are given in interactive Jupyter notebooks to demonstrate the practical application of specific components.[10]

For developers, dedicated integration guidelines are provided to support the addition of new components following a standardised workflow.[11] The pipeline is continuously extended through contributions by the community. To facilitate this, a structured workflow has been established for the modular integration of components, ensuring long-term stability, maintainability, and scalability of the overall system. Component development is further supported through a continuous integration and delivery setup (CI/CD) based on GitLab pipelines, which automatically build, test, and deploy new software versions following a semantic release workflow based on semantic versioning. The CI/CD pipelines adhere to the internal quality standards of the SUB's research and development department, as described in the SUB technical reference.[12]

Dissemination takes place via community platforms such as the SSH Open Marketplace,[13] and MONAPipe has been presented in selected training contexts, including presentations and workshops [VBD$^+$23].

2 Components and Implementations

MONAPipe is based on and extends the functionality provided by spaCy, a Python library for building NLP pipelines. SpaCy offers trained pipeline models for multiple languages, including English and German, where such a pipeline model incorporates basic pipeline components, such as tokenizer, part-of-speech tagger, parser, named entity recognizer and lemmatizer. The outputs of these components are attached to a document object, which is constructed from the input text. Each pipeline component takes a document object as input and attaches new annotations to the document, its tokens or spans (of tokens). For example, the parser adds sentence spans to the document and dependency relations to tokens.

In addition to the built-in pipeline models, spaCy offers the possibilities to 1) train custom models and 2) add custom components. Hence the first contribution of MONAPipe is a custom model for German. While spaCy's built-in model for German produces dependency trees in the TIGER scheme [Smi03], our custom model produces trees in the more recent Universal Dependencies scheme.[14] Furthermore, MONAPipe extensively uses the possibility to add custom components to incorporate specialised tools from the computational linguistic and literary research communities, including tools to detect clauses [Dön20], named entities,[15] temponyms [SG15, BD21], speech [BTWJ20], events [VHGB21], or reflective passages [GBD$^+$24]. As of version 4.4.0, MONAPipe includes components for 23 different morphosyntactic, semantic or discourse-level/narrative NLP tasks (cf. Figure 1).[16]

[8] GLAM stands for galleries, libraries, archives, and museums.

[9] https://textplus.pages.gwdg.de/collections/mona-pipe/getting_started/component_overview/#table-of-components-and-implementations

[10] https://textplus.pages.gwdg.de/collections/mona-pipe/getting_started/getting_started/#example-notebooks

[11] https://textplus.pages.gwdg.de/collections/mona-pipe/development/custom_component/

[12] https://gitlab.gwdg.de/fe/technical-reference

[13] https://marketplace.sshopencloud.eu/tool-or-service/9xGRAg

[14] https://universaldependencies.org/u/dep/

[15] https://huggingface.co/LennartKeller/fiction-gbert-large-droc-np-ner

[16] https://textplus.pages.gwdg.de/collections/mona-pipe/getting_started/component_overview/

[17] The *Tokenizer* splits raw text into individual tokens (e.g., words or punctuation). The *Tok2Vec* component produces contextualised token embeddings for downstream tasks. The *Tagger* assigns part-of-speech tags to tokens. The *Lemmatizer* maps words to their base or dictionary form. The *Normalizer* standardises text (e.g., by lowercasing or correcting common variants). The *Morphologizer* identifies morphological features (e.g., tense, number, or gender). The *Sentencizer* segments text into sentences. The *Dependency-Parser* determines syntactic relations between tokens. The *Clausizer* identifies clauses and their boundaries within sentences. The *VerbAnalyzer* identifies compound verb structures and modal verbs. The *SemanticTagger* assigns meaning-based labels to tokens or

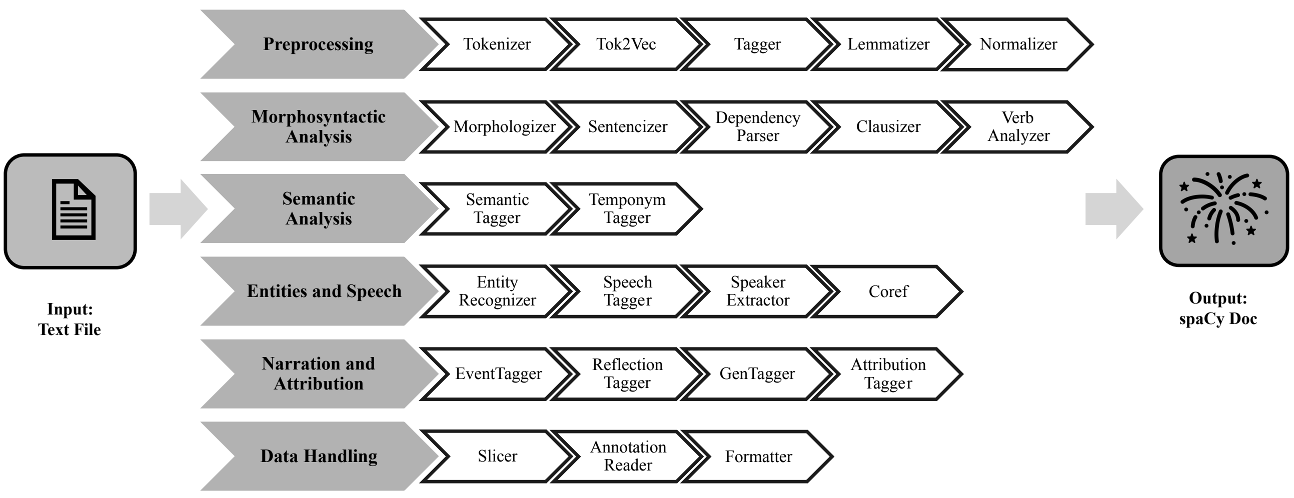


Figure 1: Components included in MONAPipe 4.4.0, grouped by levels of linguistic analysis.[17]

In the design of the MONAPipe architecture, a clear conceptual and technical distinction is made between *components* and their *implementations*. This separation follows established software design principles such as modularity and interface abstraction, and is reflected in the technical realisation of the pipeline. In MONAPipe, each component is defined as a superclass, specifying a common interface for a given task (e.g., named entity recognition, speech tagging, coreference resolution). Implementations are concrete subclasses within dedicated modules that inherit from the component classes and provide specific functionality—often based on different libraries, models, or data sources. This design enables dynamic loading of implementations, integration of community-contributed classifiers, and deployment in both local and containerised environments.

A concrete example of this architecture is the *EntityRecognizer* component, which currently includes two distinct implementations optimised for different textual domains (cf. Figure 2). The *standard implementation* provided by spaCy uses a model trained on newspapers and is designed to identify typical named entity types such as `PER`, `LOC`, `ORG`, and `MISC`, as commonly found in newswire or administrative text. In contrast, the *LiteraryCharacterNER* implementation extends the concept of named entities to include noun phrases and referential expressions that are more relevant for literary analysis.

Considering the example below, the standard implementation identifies only proper names such as *Gandalf* as persons (i), whereas the Literary Character NER additionally includes role-based references such as *Hobbit* and *Zauberer* (ii). This makes it better suited for downstream tasks like character network analysis. Both implementations conform to the same component interface and can be interchanged dynamically in the pipeline.[18]

(i) Der Hobbit sah [Gandalf]$_{\text{PER}}$ fragend an. Der Zauberer starrte schweigend ins Feuer.

(ii) Der [Hobbit]$_{\text{PER}}$ sah [Gandalf]$_{\text{PER}}$ fragend an. Der [Zauberer]$_{\text{PER}}$ starrte schweigend ins Feuer.

3 Resource Integration and Sustainability

The integration of large external resources—such as pre-trained models—into a modular NLP pipeline like MONAPipe presents specific challenges with regard to sustainability, reusability, and long-term maintenance.

spans. The *TemponymTagger* detects temporal expressions (e.g., “20 April, 2025”). The *EntityRecognizer* detects and classifies named entities (e.g., persons, locations, organisations). The *SpeechTagger* identifies direct, reported etc. speech. The *SpeakerExtractor* attributes speech spans to specific speakers. The *Coref* component resolves references to the same entity (e.g., linking “Edward” and “er” [= “he”]). The *EventTagger* categorises events described in the text. The *ReflectionTagger* marks reflective passages, i.e. passages that comment on the story or make abstract claims about the fictional or real world. The *GeneralisationTagger* identifies generalising statements (e.g., “all ravens are black”). The *AttributionTagger* detects who is responsible for specific statements in a text. The *Slicer* reduces the input text to a predefined substring. The *AnnotationReader* integrates external annotations into the pipeline. The *Formatter* structures the final output into a desired format (e.g., CoNLL-U).

[18] The MONAPipe framework generally supports using only one implementation per component at a time. In the specific case of NER, however, it is possible to use both implementations and merge the annotations.

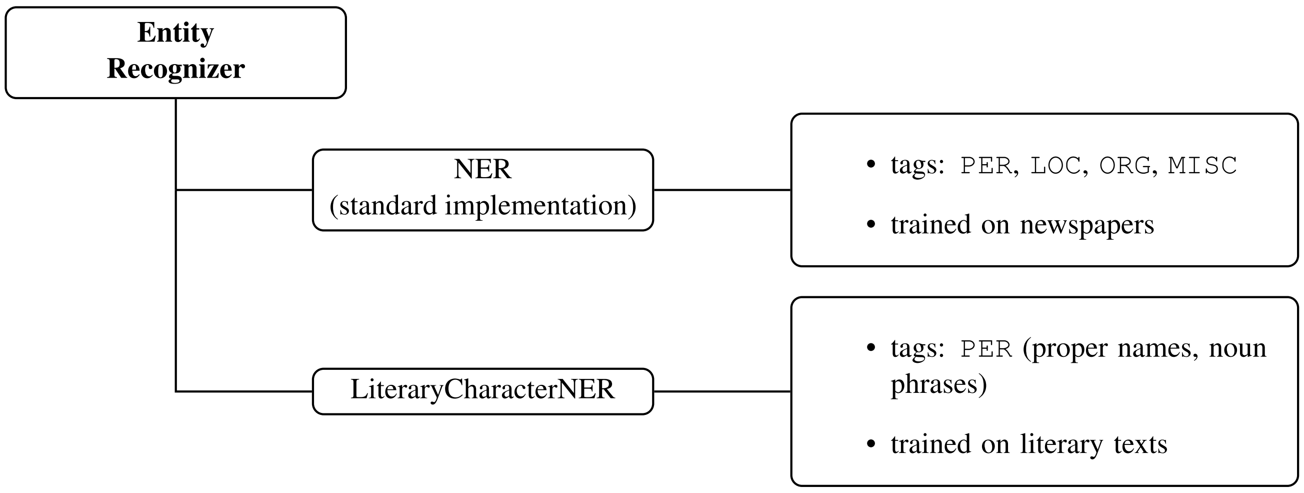

Entity Recognizer
NER (standard implementation)
tags: PER, LOC, ORG, MISC
trained on newspapers
LiteraryCharacterNER
tags: PER (proper names, noun phrases)
trained on literary texts

Figure 2: Component *EntityRecognizer* with two individual implementations.

To address these challenges, MONAPipe follows a strategy that separates software code from data resources and relies on established infrastructures for research data management.

The required resources are made available via a Dataverse repository hosted on GRO.data[19] [re323], the research data platform operated by the Gesellschaft für wissenschaftliche Datenverarbeitung mbH Göttingen (GWDG). Each resource is assigned a persistent identifier (DOI), versioned through the repository infrastructure, and accompanied by metadata that supports citability, transparency, and reuse in line with FAIR principles. During runtime, MONAPipe automatically downloads and caches the resources defined in a component's configuration, ensuring reproducibility through explicit version references. This setup allows the software to remain lightweight and modular, while preserving traceable and standards-compliant data integration. By decoupling code and data and integrating with existing research data management (RDM) infrastructures, MONAPipe supports reproducible research and aligns with broader efforts within the NFDI to build sustainable, interoperable, and community-oriented tools for working with textual data. This massively reduces the package size of the distributed library. But in addition to the library package, the software supports the usage of containers that themselves will, if required, access the larger binaries from GRO.data on demand.

4 Containerisation

During the recent development stage of MONAPipe, handling all dependencies became a technical challenge, especially when different dependencies required divergent versions of the Python interpreter. Containers offer a valid strategy to resolve version conflicts by isolating environments and allowing them to interact as standalone services within the MONAPipe system. In addition to solving this specific technical problem, containers enable portability and scalability across various infrastructures. For example, deployed containers can be used on more powerful computing platforms. To support containerisation, implementations were wrapped in functions utilising FastAPI [Ram18] to expose functionality via REST interfaces. Developers can use the containers for testing, and the continuous integration platform is configured to build, test, and deploy each container image. This process can be parallelised, making it highly efficient in terms of time consumption. Containerisation also simplifies the creation of environments that contain the specific software dependencies required by each implementation, thereby making application maintenance and updates more manageable.

The containerised implementations can be used both locally and via an online service (cf. Figure 3). For local usage, MONAPipe detects whether an implementation is container-based and builds all required containers on demand (using Docker). Furthermore, containers can be deployed to an external online service that communicates with the local Python library. By hosting the software on a High Performance Cluster (HPC) and interacting with components via the REST API, it becomes feasible to run the core application on a machine with limited resources, as the computationally intensive tasks are offloaded to the HPC. In addition, users are always able to add implementations via local containers or through third-party providers.

[19] https://data.goettingen-research-online.de/dataverse/mona-pipe-data

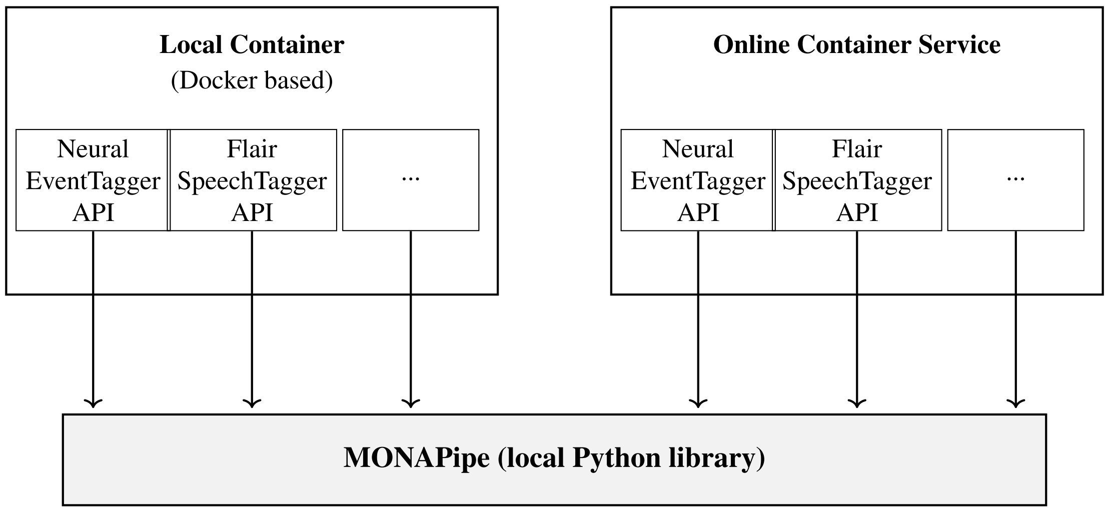


Figure 3: MONAPipe implementations (*NeuralEventTagger*, *FlairSpeechTagger*, ...) accessible via local containers and remote services.

5 Usage workflow

As stated in section 1, the research community for (digital) humanities is helped with a flexible NLP pipeline to address individual use cases as well as recurring problems that need specific adjustments.

A requirement that is stated in several user stories is the detection of named entities and in a next step the linking to authority files like GND or knowledge bases like Wikidata.[20] This ranges from the need of general tools (user stories 328, 404), the specific individual integration of entity annotations (user story 352, as described in section 1), and a dedicated entity linking service (user story 331, cf. section 1).

These user stories can be addressed by the NER models in MONAPipe. The pipeline package itself can be easily installed via pip and users are guided through the functionality in dedicated Jupyter notebooks (Figure 4). For specific demands, as described in user story 352, individual workflows can be set up by the user and the pipeline is expandable, e.g. more formatter components are planned, inter alia to directly export annotations to TEI as required in this user story.

The dedicated implementations allow users and data holders to enrich their data according to specific data domains. For the enrichment of literary texts with entities the *LiteraryCharacterNER* produce accurate results, for example, the enrichment of the *Digitale Bibliothek* within the *TextGrid repository* with entities is planned with this model.[21] More implementations from different Text+ partners for dedicated data domains will be added continuously which will bring a benefit for users as well. This includes entity linking models which are being integrated into the pipeline right now.[22] We aim to encourage more and more researchers to contribute their NLP developments to the MONAPipe architecture to maintain them in a productive infrastructure. This will increase the interoperability and long-term availability of these NLP models and make them easily available for other researchers.

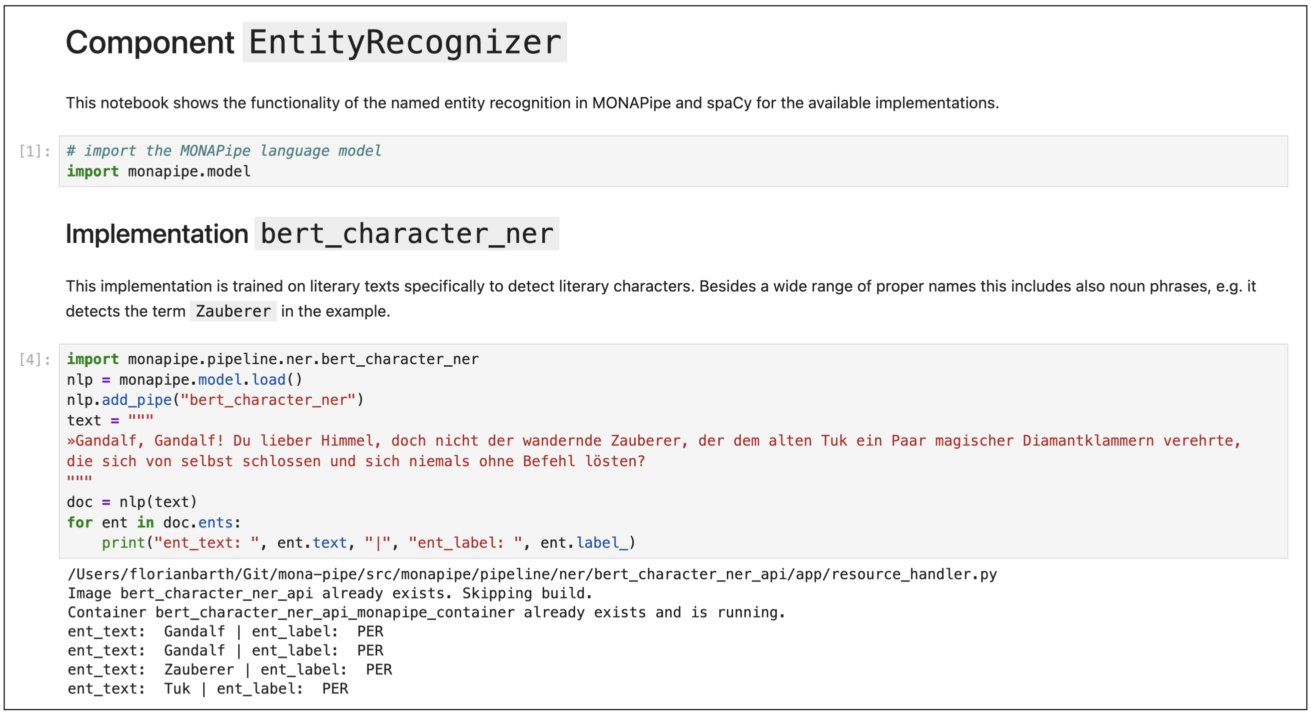
6 Conclusion

MONAPipe provides a modular NLP infrastructure tailored to the needs of research within the NFDI consortium Text+. Its component-based architecture enables the integration of both general-purpose and community-

[20] Entity recognition and linking is part of the following user stories: 328 (https://www.text-plus.org/en/research-data/user-story-328/), 331 (https://www.text-plus.org/en/research-data/user-story-331/), 344 (https://www.text-plus.org/en/research-data/user-story-344/), 352 (https://www.text-plus.org/forschungsdaten/user-story-352/), 404 (https://www.text-plus.org/en/research-data/user-story-404/).

[21] The *Digitale Bibliothek* is a literary corpus within the *TextGrid repository*: https://textgridrep.org/project/TGPR-372fe6dc-57f2-6cd4-01b5-2c4bbefcfd3c.

[22] Entity linking models were formerly integrated via spaCy external implementations into MONAPipe. These services provided by external institutions were shut down during the last months. This highlights even more the effort to establish a durable NLP tool that preserves resources in a productive state. Partners in Text+ will update formerly integrated models as well as new approaches for both GND and Wikidata linking.



Component `EntityRecognizer`

This notebook shows the functionality of the named entity recognition in MONAPipe and spaCy for the available implementations.

```
[1]: # import the MONAPipe language model
     import monapipe.model
```

Implementation `bert_character_ner`

This implementation is trained on literary texts specifically to detect literary characters. Besides a wide range of proper names this includes also noun phrases, e.g. it detects the term `Zauberer` in the example.

```
[4]: import monapipe.pipeline.ner.bert_character_ner
     nlp = monapipe.model.load()
     nlp.add_pipe("bert_character_ner")
     text = """
     »Gandalf, Gandalf! Du lieber Himmel, doch nicht der wandernde Zauberer, der dem alten Tuk ein Paar magischer Diamantklammern verehrte,
     die sich von selbst schlossen und sich niemals ohne Befehl lösten?
     """
     doc = nlp(text)
     for ent in doc.ents:
         print("ent_text: ", ent.text, "|", "ent_label: ", ent.label_)
```

```
/Users/florianbarth/Git/mona-pipe/src/monapipe/pipeline/ner/bert_character_ner_api/app/resource_handler.py
Image bert_character_ner_api already exists. Skipping build.
Container bert_character_ner_api_monapipe_container already exists and is running.
ent_text:  Gandalf | ent_label:  PER
ent_text:  Gandalf | ent_label:  PER
ent_text:  Zauberer | ent_label:  PER
ent_text:  Tuk | ent_label:  PER
```

Figure 4: Jupyter notebook for *LiteraryCharacterNER*.

developed tools, supported by structured documentation, versioned data management, continuous integration workflows, and a containerisation strategy.

In the next development stage, MONAPipe will be extended through the online API services provided via KISSKI, the AI service center of the GWDG.[23] The service is designed to host resource-intensive components on a high-performance computing infrastructure with GPU support. Existing example notebooks will be made available in the NFDI JupyterHub (Jupyter4NFDI). Further developments may include the semantic linking of annotations to existing linguistic ontologies, as well as the design of a graphical user interface to enable interactive use and exploration of pipeline results.

Bibliography

[BBT+23] F. Barth, Y. Bracke, J. C. Tello, G. Dogaru, T. Dönicke, K. Du, S. E. Funk, P. Genet, M. Göbel, L. Keller, D. Kurzawe, U. Veentjer, L. Weimer. MONAPipe: Modular Natural Language Processing Pipeline for Digital Humanities. Oct. 2023.
doi:10.5281/zenodo.8424925
https://doi.org/10.5281/zenodo.8424925

[BD21] F. Barth, T. Dönicke. Participation in the KONVENS 2021 Shared Task on Scene Segmentation Using Temporal, Spatial and Entity Feature Vectors. *Shared Task on Scene Segmentation*, 2021.

[BTWJ20] A. Brunner, N. D. T. Tu, L. Weimer, F. Jannidis. To BERT or not to BERT–Comparing contextual embeddings in a deep learning architecture for the automatic recognition of four types of speech, thought and writing representation. In *Proceedings of the 5th Swiss Text Analytics Conference (SwissText) & 16th Conference on Natural Language Processing (KONVENS)*. 2020.

[DBVS22] T. Dönicke, F. Barth, H. Varachkina, C. Sporleder. MONAPipe: Modes of Narration and Attribution Pipeline for German Computational Literary Studies and Language Analysis in spaCy. In Schaefer et al. (eds.), *Proceedings of the 18th Conference on Natural Language Processing (KONVENS 2022)*. Pp. 8–15. KONVENS 2022 Organizers, Potsdam, Germany, 12–15 Sept. 2022.
https://aclanthology.org/2022.konvens-1.2/

[23] https://kisski.gwdg.de/

[Dön20] T. Dönicke. Clause-Level Tense, Mood, Voice and Modality Tagging for German. In *Proceedings of the 19th International Workshop on Treebanks and Linguistic Theories*. Pp. 1–17. Association for Computational Linguistics, Düsseldorf, Germany, Oct. 2020.
doi:10.18653/v1/2020.tlt-1.1
https://aclanthology.org/2020.tlt-1.1

[GBD+24] B. Gittel, F. Barth, T. Dönicke, L. Gödeke, T. Schomacker, H. Varachkina, A. M. Weimer, A. Holler, C. Sporleder. Neither Telling nor Describing. Reflective Passages and Perceived Reflectiveness 1700–1945. *Journal of Computational Literary Studies* 3:1–24, 10 2024.
doi:10.48694/jcls.3861
https://jcls.io/article/id/3861/

[HMLB20] M. Honnibal, I. Montani, S. V. Landeghem, A. Boyd. spaCy: Industrial-strength Natural Language Processing in Python. 2020.
doi:10.5281/zenodo.1212303
https://dx.doi.org/10.5281/zenodo.1212303

[Ram18] S. Ramírez. FastAPI. https://fastapi.tiangolo.com, 2018. FastAPI framework, high performance, easy to learn, fast to code, ready for production.
https://github.com/fastapi/fastapi

[re323] re3data.org. GRO.data. 2023. Last accessed: 2025-04-09.
https://doi.org/10.17616/R31NJMQJ

[SG15] J. Strötgen, M. Gertz. A baseline temporal tagger for all languages. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*. Pp. 541–547. 2015.

[Smi03] G. Smith. A brief introduction to the TIGER Treebank. Technical report, Universität Potsdam, 2003.
https://www.ims.uni-stuttgart.de/documents/ressourcen/korpora/tiger-corpus/annotation/tiger_introduction.pdf

[VBD+23] H. Varachkina, F. Barth, T. Dönicke, J. Biermann, F. Altmann, T. Neitzke, C. Sporleder. Pipelines für Natural Language Processing und digitale Literaturanalyse in spaCy. Mar. 2023.
doi:10.5281/zenodo.7715520
https://doi.org/10.5281/zenodo.7715520

[VHGB21] M. Vauth, H. O. Hatzel, E. Gius, C. Biemann. Automated Event Annotation in Literary Texts. In Ehrmann et al. (eds.), *Proceedings of the Conference on Computational Humanities Research, CHR2021, Amsterdam, The Netherlands, November 17-19, 2021*. CEUR Workshop Proceedings 2989, pp. 333–345. CEUR-WS.org, 2021.
http://ceur-ws.org/Vol-2989/short_paper18.pdf