Standardizing Heterogeneous Corpora with DUUR: A Dual Data- and Process-Oriented Approach to Enhancing NLP Pipeline Integration

Leon Hammerla and Alexander Mehler and Giuseppe Abrami
{ hammerla · mehler · abrami }@em.uni-frankfurt.de
Goethe University, Frankfurt am Main, Germany

Abstract

Despite their success, LLMs are too computationally expensive to replace task- or domain-specific NLP systems. Yet, the diversity of corpus formats across domains makes it difficult to reuse or adapt these specialized systems. As a result, the NLP field faces a trade-off between the efficiency of domain-specific systems and the generality of large language models, underscoring the need for an interoperable NLP landscape. We address this challenge by pursuing two objectives: standardizing corpus formats and enabling massively parallel corpus processing. We present a unified conversion framework embedded in a massively parallel, microservice-based, programming language-independent NLP architecture designed for modularity and extensibility. It allows for the integration of external NLP conversion tools and supports the addition of new components that meet basic compatibility requirements. To evaluate our dual data- and process-oriented approach to standardization, we (1) benchmark its efficiency in terms of processing speed and memory usage, (2) demonstrate the benefits of standardized corpus formats for NLP downstream tasks, and (3) illustrate the advantages of incorporating custom formats into a corpus format ecosystem.

1 Introduction

The use of specialized NLP systems is complicated by the heterogeneity of the corpus formats with which the systems are trained, the formats they require as input or generate as output (e.g. CoNLL (Buchholz and Marsi, 2006), TigerXML (Konig and Lezius, 2000), UIMA (Ferrucci and Lally, 2004), ANNIS (Krause and Zeldes, 2014) and TEI (Consortium, 2025)). This heterogeneity is a disadvantage when NLP systems must be used in a pipeline where the output of one system serves as input for another. The situation is even worse when there are no suitable libraries for the conversion. Using resource- and cost-intensive LLMs to replace such specialized NLP systems can be prohibitively expensive (Luccioni et al., 2024; Xiao et al., 2025; Ling et al., 2024). Thus, we should maintain access to a variety of specialized NLP systems that perform specific tasks within particular disciplines or with specific language data. It would be pointless to dispense with this heritage, also because downstream LLMs can use their outputs for improvement (Wu et al., 2025; Lewis et al., 2021). This raises the question of how to solve the problem of heterogeneous data formats.

This task requires deployable and interoperable NLP systems (Moreno-Schneider et al., 2022). However, it involves more than just automatically processing heterogeneous annotations. It also requires designing the underlying schemas and evaluating their instances. This may include manually creating annotations, especially when the objective is to generate data for training NLP systems on new tasks. From this perspective, standardization of the formats used in NLP and related disciplines is central. It would allow us to combine the results of different NLP systems into a format that a third system can process without encountering conversion issues. This would create an interoperable NLP landscape, eliminating the need for conversion.

This scenario outlines the long-standing dream of interoperability (Ide and Romary, 2004), which refers to the standardization of data formats used by participating systems. Although this concept has been around for a long time, it has yet to be realized (Aufrant, 2022). This is especially problematic for approaches that aim to explicitly annotate corpora for purposes such as modeling, evaluation, and intellectual exchange across scientific disciplines. We aim to fill this research gap in the context of interoperability and the efficient parallelization of NLP systems. To this end, we address two interdependent tasks. (1) standardizing corpus annotation and (2) massively parallelizing corpus processing. The paper has three major outcomes: (1) It develops a new approach to unify the heterogeneous landscape of annotation formats (Section 4). (2) It does so by integrating our unified format into an interoperable, massively parallizable NLP system that allows the integration of any NLP system that meets a list of minimal requirements (Section 3.2). (3) To assess the effectiveness of our dual data- and process-oriented approach to standardization, we provide three evaluation scenarios (Section 5).

2 Related Work

Several initiatives have sought to manage the growing diversity of linguistic formats by converting them into a standard format. For example, the Pepper and Salt framework (Zipser and Romary, 2010) converts various formats into the Salt Meta-Model; it represents linguistic data as directed graphs and supports export to other formats. A second example is FINTAN (Fäth et al., 2020): it employs a graph representation to transform heterogeneous linguistic resources into RDF graphs, thereby enabling uniform, graph-based transformations through update scripts in SPARQL. FINTAN was followed by Spicy Salmon (Fäth and Chiarcos, 2022), which integrated Pepper into the FINTAN workflow, enabling the conversion among 50 linguistic formats, including many CoNLL and RDF standards. A recent addition to the “spicy” ecosystem of Pepper, Salt, and ANNIS (Krause and Zeldes, 2014) is Annatto (Krause and Klotz, 2025). Like Pepper and FINTAN, it has a modular architecture with dedicated importer and exporter components, and utilizes a graph-based internal representation based on the graphANNIS (Krause et al., 2016) data model. Beyond format conversion, Annatto includes a suite of graph manipulation modules supporting tasks such as filtering, editing, consistency checking, and debugging. It also provides integrated tools for graph visualization and corpus search, making it a powerful resource for both corpus exploration and linguistic data processing workflows. More specialized is the *Universal Dependencies* (Nivre et al., 2017) project, which standardizes syntactic annotation by unifying the myriad variations of CoNLL formats. Similarly, the *Natural Language Interchange Format* (NIF) project (Hellmann et al., 2013) strives to establish a universal, RDF-based standard for linguistic data and, with its 2.0 update, introduces enhanced capabilities for corpus conversion. There are numerous third-party converters that provide a range of conversion capabilities as, e.g., openConvert (Bakker, 2015) from the CLARIN-NL project and the Python-based CoNLL-U Parser (Stenström, 2016), which can handle various variations of CoNLL-U files. There are many other tools, each of which caters to specific linguistic formats and conversion needs.

All these efforts aim at overcoming the heterogeneity of annotation formats by mapping them to a unified format. This includes aggregating annotations of the same resource to make them available as a whole for further processing (vertically) or to facilitate automatic annotation by making them available to each other (horizontally), especially in scenarios of data scarcity due to data acquisition problems. However, these approaches lack an integrative perspective that aims at a unified format within a procedurally unified system that enables massive parallelization of NLP systems and enables new NLP pipelines based on unified data to meet the requirements of text-based disciplines. Thus, unlike our work, these approaches are not primarily oriented toward dual, data-, and process-based standardization. With DUUR we address this research gap: we propose a unified framework that harmonizes heterogeneous formats and allows extensions to be added independently of any programming language. Our framework adheres to a unified interface and integrates format conversion into a reproducible, modular pipeline architecture. This provides three extensions to existing research:

1. We map the dual nature of NLP workflows in terms of data and process orientation by integrating our corpus conversion framework into an NLP pipeline infrastructure.

2. We provide a standard for conversion modules that uses a containerized, microservice-based architecture. This allows new modules to be developed independently of programming languages and requires only minimal knowledge of the underlying system. As a result, it becomes feasible for third parties to extend the framework with their own components.

3. We support the integration of conversion tools into a common architecture that enables their combined use, promoting interoperability and reuse across annotation formats.

3 Preliminaries

To explain our standardization framework DUUR for orchestrating containerized, microservice-based

READER-COMPONENTS, we first introduce the corpus format used as a base representation, and then present the NLP platform into which it is integrated, namely the Docker Unified UIMA Interface (DUUI) (Leonhardt et al., 2023).

3.1 UIMA

The Unstructured Information Management Architecture (UIMA) is a framework originally developed by IBM to support large-scale analysis of unstructured data (Ferrucci and Lally, 2004). The UIMA format defines how data and annotations are represented and exchanged within this framework. Its core data model, the Common Analysis Structure (CAS), stores the original unstructured content together with structured annotations. Through its offset-based annotation model, multi-view architecture, and extensible type system, the UIMA format can represent virtually any kind of annotation. This standardized representation ensures interoperability among NLP components and enables seamless integration of tools from diverse sources into a unified processing pipeline, preserving all annotation information.

3.2 The DUUI Platform

DUUI standardizes the integration, not the formats, of corpora and NLP systems across implementations through platform-independent interfaces. It addresses five areas: (1) horizontal and vertical scaling, (2) heterogeneous annotation and implementation landscapes, (3) managing reproducible and reusable annotations, (4) monitoring and error reporting, and (5) usability. DUUI uses a UIMA-based type system for text processing. It encapsulates NLP systems using containerized services such as Docker (Merkel, 2014) or Kubernetes, ensuring cross-environment compatibility while resolving dependency conflicts (Abrami et al., 2025). To facilitate scalable, flexible, and error-tolerant processing of corpora, DUUI employs DRIVERS to control NLP processes across different runtime environments. DRIVERS are orchestrated by a top-level COMPOSER, which manages their execution. Communication between the COMPOSER and COMPONENTS is implemented via RESTful web services. Through the ensemble of COMPOSER, DRIVERS, and COMPONENTS, DUUI models NLP pipelines. These pipelines process UIMA documents sequentially, with the computation taking place within the COMPONENTS, ensuring a modular, reusable workflow.

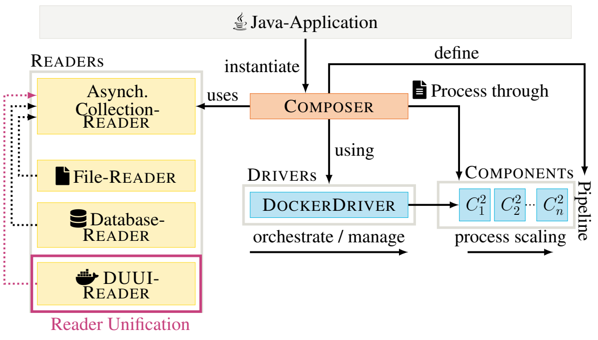


Figure 1: Integrating READERS into DUUI.

DUUI bridges the gap between NLP systems that run on different platforms or are implemented in different programming languages and harmonizes annotation schemes through its type system. However, it is still challenged by the ever-increasing diversity of data formats. Annotations must be converted into UIMA documents used by DUUI's internal corpus representation. DUUI is implemented in Java, which is the most efficient way to use UIMA. Therefore, format-specific readers for DUUI (see Figure 1) must also be implemented in Java, which requires expert knowledge of DUUI itself. This makes it difficult to develop READERS for custom corpus formats. Thus, we are faced with a situation in which DUUI makes it easier to use more and more NLP systems while simultaneously making it difficult to adapt or generate READERS for ever-new formats: DUUI is procedurally standardized, but not with regard to the formats made accessible by this process-related standardization. We need a dual, data- and process-oriented standardization. This is exactly what we develop in terms of DUUR based on DUUI.

4 DUUR

DUUR is a reader framework operating similarly to DUUI's pipeline framework. It features containerized, microservices-based reader components for custom formats and conversion tools.

4.1 Software Design

DUUR consist of a COMPOSER that manages multiple DRIVERS each controlling specialized READER-COMPONENTS that extend the standard DUUI component and inherit its functionality. A READER-COMPONENT retains all of the standard COMPONENT's RESTful API calls to facilitate communication between the COMPOSER and individual COMPONENTS. The most critical request in this interaction is the process request. Within DUUI, this POST request transfers a UIMA document, serialized using a particular LUA context, from the COMPOSER to a COMPONENT. The COMPONENT processes the serialized UIMA document, applies its operations, and returns the modified document. Upon receipt, the COMPOSER deserializes the document back into its UIMA format before passing it on to the next COMPONENT in the pipeline. In the DUUR framework, we extend the use of the POST request process, with the difference that an empty UIMA document is sent to the READER-COMPONENT, which then populates the document based on the specified corpus that it parses and converts. This process requires two conditions to be met: the READER-COMPONENT must have access to the target corpus, and the COMPOSER must know the total number of documents that can be extracted from it. This ensures that the COMPOSER can generate and send the appropriate number of empty documents to the READER-COMPONENT before exhausting the available content. To implement this, we introduce an additional RESTful API call to the READER-COMPONENT, called init. This init POST request allows the COMPOSER to send a compressed directory (e.g. a ZIP archive) containing one or more corpora to the READER-COMPONENT. Upon receiving the archive, the READER-COMPONENT extracts the relevant data based on its format conversion capabilities. Once the extraction is complete, the COMPONENT determines the total number of documents that can be derived from the provided corpus and returns it to the COMPOSER. With this information, the COMPOSER proceeds to send the appropriate number of empty UIMA documents to the READER-COMPONENT, which populates them with the extracted content.

4.2 Features

A READER-COMPONENT can be specialized to parse and convert a specific format, or act as a wrapper for an existing or custom converter that can handle multiple formats. This design allows for integrating tools such as Pepper and Annatto into our workflow, enabling reuse and reducing implementation overhead. In its current state, DUUR supports the conversion of 47 corpus formats (see Table 1) (not counting format variants). A list of all supported formats can be found in Appendix A.

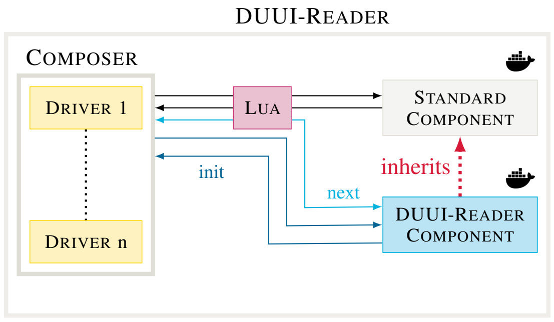


Figure 2: DUUI-READER architecture, illustrating the communication flow between the COMPOSER and COMPONENTs, with special emphasis on the modified interaction for the READER-COMPONENT.

4.2.1 Pepper Component

The Java-based conversion tool Pepper facilitates interoperability between widely used formats such as TigerXML (Lezius, 2002), EXMARaLDA (Schmidt and Wörner, 2014), and PAULA (Zeldes et al., 2013). It uses Salt, a graph-based model, as a mapping layer: first, input formats are mapped to Salt, before they are converted to the desired output format. To integrate Pepper into DUUI, we encapsulate it in a Docker container along with a RESTful API that adheres to our READER-COMPONENT guidelines. This approach allows use of Pepper's full conversion capabilities.

4.2.2 ANNIS Component

ANNIS (Krause and Zeldes, 2014) is a web-based search and visualization tool for linguistically annotated corpora, developed within the Pepper & Salt ecosystem. It uses a proprietary format that supports multi-layer annotation and querying, provides corpus visualization features and a number of richly annotated resources, such as reference corpora for the historical phases of German (Klein et al., 2016; Zeige et al., 2025; Wegera et al., 2021). For a long time, there was no standardized way to convert data from ANNIS to other formats, even within the Pepper & Salt framework. Recently, the Annatto project (Krause and Klotz, 2025), which continues the legacy of the defunct Pepper project, introduced initial conversion capabilities for the relANNIS 3.3 format. However, converting ANNIS data poses challenges due to fundamental conceptual differences: while ANNIS relies on virtual tokens as its primary unit of analysis, the UIMA framework is based on the Sofa (Subject of Analysis) model. These differences make direct annotation mapping difficult. To address this, we develop an ANNIS

Reader-Component that allows mapping between these different representations. This allows ANNIS corpora to be incorporated into NLP workflows while preserving and enabling the extraction of annotations in the ANNIS format.

4.2.3 Annatto

Annatto (Krause and Klotz, 2025) is a corpus conversion and exploration tool that continues the line of Salt, Pepper and ANNIS. Conceptually, it can be seen as a modernized successor to Pepper, offering extended functionality, especially in the area of corpus exploration. Unlike Pepper, which is based on the Salt metamodel, Annatto is built on the graphANNIS (Krause et al., 2016) infrastructure and its data model. Annatto provides a rich set of importers, exporters, and graph operations modules that facilitate corpus conversion, manipulation, exploration, and visualization. To leverage Annatto's conversion capabilities within DUUI, we encapsulate it in a Reader-Component that adheres to our standardized interface.

4.2.4 OpenConvert Component

OpenConvert (Bakker, 2015) is a Java-based conversion tool that supports various text formats such as TEI, DOC, and DOCX. To integrate it into DUUR, we take the same approach as with Pepper: encapsulate it in a Docker container and provide it with a RESTful API that follows our Reader-Component guidelines. This ensures compatibility with DUUI while leveraging OpenConvert's conversion capabilities.

4.2.5 SketchEngine Component

Sketch Engine (Kilgarriff and Computing, 2003–2023) is a corpus management and text analysis system used by lexicographers and linguists to analyze large text corpora through queries. Named for its core feature, word sketches, it generates concise summaries of a word's grammatical and collocational behavior. With support for over 90 languages and a suite of tools including concordance searching, thesaurus generation, and term extraction, it is a resource for research, dictionary building, and language teaching. To improve interoperability between Sketch Engine and the broader UIMA ecosystem, which includes advanced corpus visualization and annotation tools such as DUUI, the Unified Corpus Explorer (UCE) (Bönisch et al., 2025), and the TextAnnotator (TA) (Abrami et al., 2020), we generate a Sketch Engine component within DUUR. This allows linguists working with Sketch Engine to use the range of tools available in the UIMA ecosystem.

4.2.6 Leipzig Glossing Component

We provide a module for reading linguistic resources encoded in the Leipzig Glossing Rules (LGR) (Max Planck Institute, 2008), especially in the TEX and TXT formats, since existing tools like lingglosses (Moroz, 2021) and pyigt (List et al., 2021) already support parsing of the CSV and XML variants. A significant number of linguistic datasets, especially for underrepresented languages, are available in this format. Our goal is to make these resources available to the research community, while allowing linguists already working with LGR to take advantage of the full range of NLP.

4.2.7 Negation Components

The landscape of language resources related to negation, especially those annotating negation cues, scopes, events, and foci, is heterogeneous. Such resources are limited in number and use proprietary formats. In a low-resource domain such as negation detection, it is even more important to unify datasets into a common format. We consider the available corpora in this area, with a focus on the compilation of English negation corpora proposed in (Jiménez-Zafra et al., 2020). We selected a subset of the publicly available corpora because some of those listed as accessible in (Jiménez-Zafra et al., 2020) are no longer available (namely BioInfer (Pyysalo et al., 2007), NEG-DrugDDI (Bokharaeian et al., 2013), NegDDI-DrugBank (Bokharaeian et al., 2014), and Product Review (Councill et al., 2010) datasets). We provide a reader component for each supported corpus format to standardize and unify data representation [1].

5 Evaluation

Our approach optimizes workflows for processing heterogeneous formats. Its modular architecture supports integrating existing converters and developing custom components. This allows corpora to be incorporated into UIMA, making numerous NLP systems accessible for linguistic data processing. However, this diversity also makes evaluating our approach challenging. We pursue an efficiency- and a machine learning-related evaluation:

[1] See Table 3 for a list of all corpora and their annotation types, the number of sentences in each corpus and how annotations are distributed across categories.

Type	**Component**	**#Formats**
Wrapped Converter		
	Pepper	25
	Annatto	15
	openConv	6
Standalone		
	ANNIS	1
	Sketch Engine	1
	SOCC	1
	SFU	1
	Conan Doyle	1
	PB-FOC	1
	Bioscope	1
	DT-NEG	1
	LGR	1
Total[*]		**47**

[*]In total, there are 47 corpus formats that we can convert. Since some of the pre-existing converters built into our framework have overlapping conversion capabilities, these duplicates have been subtracted from the total.

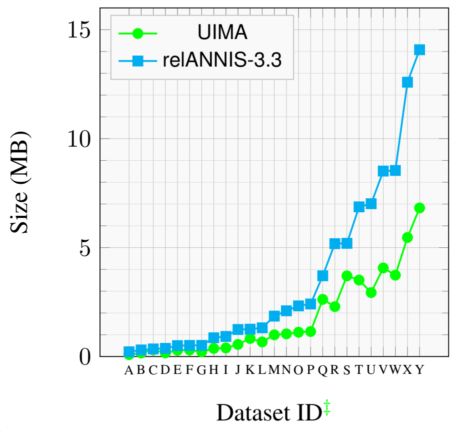
Table 1: Overview of READER-COMPONENTS: Three of the components are pre-existing converters integrated into DUUR, while the rest are dedicated to specific corpus formats. Variants of the formats are excluded from the count. See the Appendix for a list of all the specific corpus formats that each component can convert.

1. **Time and memory:** With relANNIS-3.3 as a case study, we demonstrate that our framework outperforms format-specific importers in terms of processing speed, while also producing more storage-efficient UIMA documents.
2. **Machine learning:** We refer to the task of negation detection, i.e. of the key components of negation as described in (Jiménez-Zafra et al., 2020): the cue, the scope, the focus, and the negated event. For each publicly available negation detection corpus, we develop and publish a separate reader component that integrates them into a unified format. We then use the resulting cue detection dataset to illustrate the performance gains that can be achieved by using this larger, more diverse dataset. All training is conducted within the DUUI framework, highlighting the benefits of linking existing resources within a broad, interoperable NLP infrastructure.
3. **Method extension:** Third, we briefly discuss the availability of NLP systems within the UIMA ecosystem as made possible by the format conversion provided by DUUR.

5.1 Time and Memory Efficiency

Using a case study with relANNIS-3.3, we evaluate the time and memory efficiency of DUUR and show that it offers advantages in both areas. We compare its import and conversion speed with that of two systems designed for importing and converting the relANNIS-3.3 format: the official ANNIS corpus tool (Krause and Zeldes, 2014) and the Annatto tool (Krause and Klotz, 2025). The relANNIS-3.3 format serves as a benchmark, since several importers are available and the format is well established. Our evaluation was carried out using the Reference Corpus of Old High German (Zeige et al., 2025), a collection consisting of several subcorpora of different sizes (see Table 5).

Our evaluation shows that DUUR outperforms both Annatto and the official ANNIS tool in almost all cases, achieving significantly faster processing times. On average, our method required 3.8 seconds per corpus, compared to 17 seconds for Annatto and 7.4 seconds for ANNIS (see Figure 4). Moreover, the resulting corpora, when stored as UIMA documents, require less disk space than their counterparts in relANNIS-3.3 format. After compressing each corpus using the BZ2 algorithm, we observed an average size of 1.8 MB for the UIMA representations, as opposed to 3.6 MB for those in relANNIS-3.3 format, representing a 50% reduction in storage requirements (see Figure 3).



‡A complete mapping of dataset IDs to their dataset names can be found in the appendix B. All datasets originate from the reference corpus of Old High German (Zeige et al., 2025).

Figure 3: A comparison of dataset sizes in UIMA and relANNIS-3.3 formats, ordered by relANNIS-3.3 size (ascending). Both datasets were compressed using the same compression algorithm, namely BZ2.

5.2 Utility through Use Case Integration

Creating larger, more balanced datasets by unifying corpora across diverse domains is crucial for improving the performance of downstream NLP tasks, especially in settings with limited resources. These

Train \Test	SOCC	BIOSCOPE	CONAN	PB-FOC	SFU	DT-NEG	Avg.
SOCC	0.9423	0.7585	0.7422	0.9115	0.7484	0.8293	0.8220
BIOSCOPE	0.2119	0.8702	0.3006	0.8139	0.3782	0.5134	0.5147
CONAN	0.8569	0.6952	0.8511	0.8616	0.7011	0.8277	0.7989
PB-FOC	0.5230	0.7500	0.5985	0.9721	0.6199	0.8065	0.7116
SFU	0.7219	0.7956	0.6888	0.9173	0.8037	0.8620	0.7982
DT-NEG	0.1617	0.7801	0.4074	0.8774	0.3398	0.9363	0.5838
ALL (except Test)	0.5729	0.7986	0.7149	0.9199	0.7570	0.8876	0.7751
ALL	0.9516	0.8473	0.7745	0.9184	0.7868	0.9354	0.8690
Few-Shot Qwen3-0.6B	0.5743	0.4983	0.5753	0.6514	0.5218	0.6997	0.5868
ICL GPT-5	0.6416	0.5751	0.7945	x	0.5478	0.5401	0.6198

Table 2: Cross-dataset binary F1 scores of cue detection models. Each cell shows the F1 score of a model trained on the dataset in the row and evaluated on the dataset in the column. In-domain results are highlighted in gray. For each test dataset, the highest and second-highest out-of-domain scores are highlighted in green and light green, respectively. Models trained on ALL use the combined data from all datasets, representing joint in-domain performance across datasets, whereas ALL (except Test) refers to models trained on all datasets except the test dataset, indicating joint out-of-domain generalization. We additionally include a few-shot baseline using a small LLM (Qwen3-0.6B) and an pure ICL example with GPT-5 to illustrate the gap between large pretrained models and task-specific fine-tuning for cue detection.

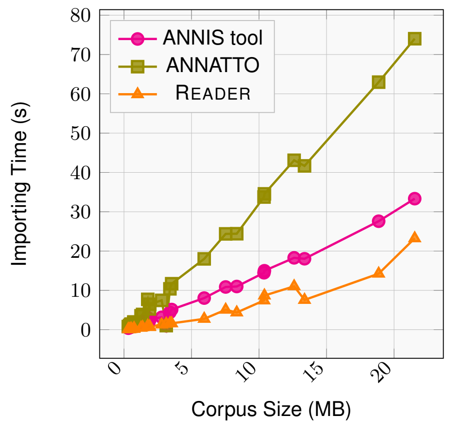


Figure 4: Comparison of import times for ANNIS, ANNATTO and READER over different corpus sizes in terms of uncompressed versions in the relANNIS-3.3 format. Tested corpora are part of the Reference Corpus for Old High German (Zeige et al., 2025). The performance differences visible on the y-axis show that READER has consistently lower import times.

unified corpora enable the training of generalized models that can effectively handle linguistic phenomena. For negation detection, which primarily involves identifying negation cues and their scopes, (Truong et al., 2022) and (Khandelwal and Sawant, 2020) demonstrate that models trained on a single corpus often generalize poorly to other corpora. This underscores the need for more generalized and diverse datasets in negation detection, given that linguistic variation significantly impacts the identification of cues, scopes, events, and foci. We propose this as a use case to evaluate DUUR. There are few existing resources for detecting negation, usually in incompatible, isolated formats. To address this issue, we collect publicly available negation corpora and implement a READER-COMPONENT for each (see Section 4.2.7). Finally, we convert all annotations into a unified UIMA format, resulting in components and datasets for the following corpora:

1. The **SOCC** dataset (Kolhatkar et al., 2019) is a corpus of user comments on online news articles, primarily from opinion pieces, annotated for negation cues and their scopes.

2. The **BioScope** corpus (Vincze et al., 2008) consists of biomedical texts, including scientific abstracts and clinical reports, annotated for negation and uncertainty.

3. The **PropBank Focus** corpus (Blanco and Moldovan, 2011) contains sentences from the Wall Street Journal section of the Penn Treebank, annotated with verbal negations.

4. **ConanDoyle-neg** (Morante and Daelemans, 2012) contains stories by Arthur Conan Doyle annotated with negation cues, their scope, and the events or properties being negated.

5. The **SFU Review** Corpus (Konstantinova et al., 2012) consists of reviews from various fields, annotated for negation and speculation.

6. The **DeepTutor Negation** corpus (Banjade and Rus, 2016) consists of student-tutor dia-

logues, annotated for the presence, scope and communicative function of negation.

To assess the impact of domain variation, we replicate the NEG-BERT cue detection architecture proposed by (Truong et al., 2022), using BERT tiny as the backbone, a compact model with 4.4 million parameters (Turc et al., 2019; Bhargava et al., 2021). We train one model on each individual dataset we collected. For each target dataset, we also train a model on the combination of all other datasets, excluding the target, to simulate out-of-domain performance. This setup yields a full cross-dataset evaluation matrix (see Table 2): each model is trained on one dataset and tested on all datasets, along with an additional row where models are trained on the combination of all datasets except the test set. We further train one model on the combination of all datasets to assess whether incorporating additional out-of-domain data can enhance in-domain performance. All models are trained on a single NVIDIA GeForce RTX 4060 Ti using identical hyperparameters: a learning rate of 2e-5, a batch size of 16, and the AdamW optimizer (weight decay = 0.01). Training runs for up to 20 epochs with early stopping (patience = 3). Across all experiments, we report the binary F1 score as the primary evaluation metric. Consistent with (Truong et al., 2022), we observe that models trained on individual corpora perform significantly worse when evaluated out-of-domain. This highlights the challenges of generalization in negation cue detection across domains. In contrast, the combined training approach achieves the highest out-of-domain F1 scores on four (BIOSCOPE, PB-FOC, SFU, and DT-NEG) of the six datasets and the second best on CONAN, demonstrating the most robust performance across domains (see Table 2). The only exception is the SOCC corpus, where the combined training approach performs worse. The likely reason is that SOCC contains the most diverse and extensive range of annotated cues, including auxiliaries, modals, conjunctions, negation words, negative pronouns, prepositions and determiners, comparatives, adjectives, adverbs, negative predicates, and adjuncts. This variety spans syntactic, lexical, and morphological domains. It is only partially shared by CONAN, which also includes annotations for negative predicates and adverbs or adjectives with affixal negation. This feature combination is not present in the other datasets. As a result, a pattern emerges between SOCC and CONAN: models trained on one consistently achieve the best performance when evaluated on the other. The in-domain performance for SOCC ranks second among all in-domain scenarios, while the out-of-domain performance for this corpus varies the most among all six out-of-domain scenarios: the standard deviation of the out-of-domain performance on SOCC is 0.3, the highest among all evaluated corpora (followed by CONAN with 0.196). In comparison, the average standard deviation across all corpora is 0.154, which is approximately 50% lower than that of SOCC. These features point to a peculiarity of SOCC that distinguishes it from the other corpora. Notably, the SFU Review corpus serves as a strong single-corpus baseline, delivering the second-best out-of-domain performance on four datasets: The likely reason is that this corpus is the largest among all the corpora considered and is explicitly designed to provide a balanced representation of multiple domains. Although it contains only the second-highest number of sentences with cue annotations, with PB-Foc having the most, it primarily features syntactic negation cues. Remarkably, '*not*' and '*no*' account for over 50% of all negation cue annotations across all datasets. As a result, strong performance in detecting syntactic cues contributes significantly to overall performance, which the SFU Review corpus appears to support particularly well. For the in-domain experiment, where we trained a single model on all datasets to examine whether incorporating additional data could improve in-domain performance, we observed gains only for the SOCC dataset. For all other datasets, the jointly trained model did not surpass the performance of models specialized on their respective in-domain data. Nevertheless, the model trained on all datasets achieved the highest overall average performance across all test datasets, with an average F1 score of 0.8690, representing an absolute improvement of 4.7 percentage points over the strongest single-dataset model (SOCC, with an average of 0.8220). To further support our claim that current LLMs cannot yet fully replace highly specialized NLP pipelines for specific tasks, we also evaluated a small-scale LLM (Qwen3-0.6B (Yang et al., 2025)) in a few-shot setting and GPT-5 in a pure in-context learning (ICL) setup[2]. Both models performed substantially worse on the cue detection task, reaching

[2]The prompts for both scenarios are provided in Appendix C.

Corpus	Sentence Sentence	S. w/ Cue	S. w/ Scope	S. w/ Focus	S. w/ Event
SFU	17 263	2853	2715	0	0
PB-Foc	10 641	4830	0	3547	3547
Bioscope	14 462	2094	2094	0	0
SOCC	1043	644	633	633	0
DT-NEG	1180	590	346	336	0
Conan-Doyle	5520	1227	1135	0	805
TOTAL	50 109	12 238	6923	4516	4352

Table 3: NEG-English corpus statistics. The 'S. w/ {Cue |Scope|Focus|Event}' column indicates the number of sentences containing at least one of the resp. annotation.

average binary F1 scores of 0.5868 (Qwen3-0.6B) and 0.6198 (GPT-5)[3], respectively. Additionally, these models incur substantially higher computational costs and runtimes. While training the all-datasets model for 20 epochs (plus evaluation) took only 321 seconds for training and 10 seconds for evaluation on the test sets, we measured an evaluation runtime of 451,660 seconds per test set with Qwen3-0.6B, both experiments were conducted on the same NVIDIA H200 NVL GPU. We provide the combined dataset, used for this evaluation, as NEG-English[4] (see Table 3); it contains 12 238 sentences with cue, 6 923 with scope, 4 516 with focus and 4 676 with event annotation.

5.3 Ecosystem Benefits

Corpus work relies on functionalities such as *acquisition* and *management*, *annotation*, *search*, *visualization*, and *analysis*. All of these require flexible and scalable NLP pipelines that can be leveraged through DUUI. DUUR contributes to this by converting corpora from various formats and by extending the pipelines with customizable converter components. As part of our dual standardization approach, DUUI provides an infrastructure that integrates TA for creating and managing manual annotations and UCE for searching, visualizing, and analyzing corpora. Here, DUUR provides standardized corpora for such services. It enables heterogeneous corpora to be embedded in DUUI's method landscape, allowing researchers to benefit from a wide range of NLP methods and work scenarios without knowing the specifics of the underlying corpora. This approach is particularly interesting in the context of interoperability: using a uniform format opens up a wider range of methods for downstream NLP operations (e.g., negation detection), where the results of these methods are available for mutual improvement.

[3]We excluded the PB-FOC dataset from the GPT-5 experiment due to licensing restrictions and limited public availability.

[4]Dataset available at OSF: `https://osf.io/cq5ky/?view_only=b7ff3cdc5fc34e42a09c9aadf45fc7fe`

6 Conclusion

Regarding the heterogeneity of corpus formats in NLP, we proposed a solution in the form of a conversion framework. This framework is embedded in a massively parallel, microservice-based, programming language-independent NLP architecture that is modular and extensible. Our framework supports the conversion of 47 formats. It integrates many NLP format conversion tools and introduces components developed specifically for certain application areas and their data formats. It also provides developers with guidelines for extending the system with their own modules. NLP systems based on explicitly annotated corpora will continue to be relevant in areas such as digital humanities and related disciplines. It is unlikely that LLMs will take over this type of work. Therefore, standardizing formats is a central task, and with DUUR, we provide an efficient solution. Our evaluation demonstrates how unifying corpus formats can benefit LLM-based workflows. E.g., fine-tuning on multiple unified corpora can result in more robust and generalizable models than training on a single source. To support this research, we publish our unified dataset. Our framework supports a variety of annotation formats and NLP systems. It connects isolated systems, reducing redundancy and enabling interoperability. Future work will expand coverage to underrepresented formats. Ultimately, we envision an interoperable NLP ecosystem in which specialized tools and large-scale models interact through shared, standardized corpora.

Limitations

Our framework is currently focused on text-based corpus formats. While its architecture is, in principle, extensible to multimodal corpora (e.g. audio or video (Bundan et al., 2025)), no READER-COMPONENTS for handling such modalities have been implemented to date. In addition, although the framework currently supports 47 distinct corpus formats, some legacy or highly domain-specific annotation standards remain outside the implementation scope. Integration of these formats would require the development of additional conversion modules. Furthermore, in certain cases, complete fidelity of annotation conversion cannot be guaranteed due to structural mismatches between format assumptions. For example, the ANNIS format allows for virtual tokens as the primary unit of analysis, whereas UIMA requires a fully interpretable sofa (Subject of Analysis) string as the base analytical unit. These fundamental differences can lead to unavoidable losses or approximations during format transformation, particularly when converting between fundamentally divergent representational models.

Acknowledgments

This work was supported by the German Research Foundation (DFG) as part of Project INF within CRC 1679 "Negation in Language and Beyond" (NegLaB[5]).

References

Giuseppe Abrami, Markos Genios, Filip Fitzermann, Daniel Baumartz, and Alexander Mehler. 2025. Docker unified uima interface: New perspectives for nlp on big data. *SoftwareX*, 29:102033.

Giuseppe Abrami, Manuel Stoeckel, and Alexander Mehler. 2020. Textannotator: A uima based tool for the simultaneous and collaborative annotation of texts. In *Proceedings of The 12th Language Resources and Evaluation Conference*, pages 891–900, Marseille, France. European Language Resources Association.

Lauriane Aufrant. 2022. Is NLP ready for standardization? In *Findings of the Association for Computational Linguistics: EMNLP 2022*, pages 2785–2800, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Jan Theo Bakker. 2015. Openconvert.

Rajendra Banjade and Vasile Rus. 2016. DT-neg: Tutorial dialogues annotated for negation scope and focus in context. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16)*, pages 3768–3771, Portorož, Slovenia. European Language Resources Association (ELRA).

Prajjwal Bhargava, Aleksandr Drozd, and Anna Rogers. 2021. Generalization in nli: Ways (not) to go beyond simple heuristics. *Preprint*, arXiv:2110.01518.

Eduardo Blanco and Dan Moldovan. 2011. Semantic representation of negation using focus detection. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 581–589, Portland, Oregon, USA. Association for Computational Linguistics.

Behrouz Bokharaeian, Alberto Díaz, and Miguel Ballesteros. 2013. Extracting drug-drug interaction from text using negation features. *Procesamiento del lenguaje natural*, 51:49–56.

Behrouz Bokharaeian, Alberto Diaz, Mariana Neves, and Virginia Francisco. 2014. Exploring negation annotations in the drugddi corpus. In *Fourth workshop on building and evaluating resources for health and biomedical text processing (BIOTxtM 2014)*, pages 1–8. Citeseer.

Kevin Bönisch, Giuseppe Abrami, and Alexander Mehler. 2025. Towards unified, dynamic and annotation-based visualisations and exploration of annotated big data corpora with the help of unified corpus explorer. In *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (System Demonstrations)*, pages 522–534, Albuquerque, New Mexico. Association for Computational Linguistics. Best Demo Award.

Sabine Buchholz and Erwin Marsi. 2006. CoNLL-X shared task on multilingual dependency parsing. In *Proceedings of the Tenth Conference on Computational Natural Language Learning (CoNLL-X)*, pages 149–164, New York City. Association for Computational Linguistics.

Daniel Bundan, Giuseppe Abrami, and Alexander Mehler. 2025. Multimodal docker unified UIMA interface: New horizons for distributed microservice-oriented processing of corpora using UIMA. In *Proceedings of the 21st Conference on Natural Language Processing (KONVENS 2025): Long and Short Papers*, KONVENS '25, pages 257–268, Hannover, Germany. HsH Applied Academics.

TEI Consortium. 2025. Tei p5: Guidelines for electronic text encoding and interchange.

Isaac Councill, Ryan McDonald, and Leonid Velikovich. 2010. What's great and what's not: learning to classify the scope of negation for improved sentiment analysis. In *Proceedings of the workshop on negation and speculation in natural language processing*, pages 51–59.

Christian Fäth and Christian Chiarcos. 2022. Spicy salmon: Converting between 50+ annotation formats with fintan, pepper, salt and powla. In *Proceedings of the 8th Workshop on Linked Data in Linguistics within the 13th Language Resources and Evaluation Conference*, pages 61–68, Marseille, France. European Language Resources Association.

Christian Fäth, Christian Chiarcos, Björn Ebbrecht, and Maxim Ionov. 2020. Fintan - flexible, integrated transformation and annotation eNgineering. In *Proceedings of the Twelfth Language Resources and Evaluation Conference*, pages 7212–7221, Marseille, France. European Language Resources Association.

[5]https://www.neglab.de/

David Ferrucci and Adam Lally. 2004. Uima: an architectural approach to unstructured information processing in the corporate research environment. *Natural Language Engineering*, 10(3–4):327–348.

Sebastian Hellmann, Jens Lehmann, Sören Auer, and Martin Brümmer. 2013. Integrating nlp using linked data. In *The Semantic Web – ISWC 2013*, pages 98–113, Berlin, Heidelberg. Springer Berlin Heidelberg.

Nancy Ide and Laurent Romary. 2004. International standard for a linguistic annotation framework. *Natural language engineering*, 10(3-4):211–225.

Salud María Jiménez-Zafra, Roser Morante, María Teresa Martín-Valdivia, and Luis Alfonso Ureña-López. 2020. Corpora annotated with negation: An overview. *Computational Linguistics*, 46(1):1–52.

Aditya Khandelwal and Suraj Sawant. 2020. Negbert: A transfer learning approach for negation detection and scope resolution. *Preprint*, arXiv:1911.04211.

Adam Kilgarriff and Lexical Computing. 2003–2023. Sketch engine. Corpus manager and text analysis software, first released in 2003.

Thomas Klein, Klaus-Peter Wegera, Stefanie Dipper, and Claudia Wich-Reif. 2016. Reference Corpus of Middle High German (1050–1350) (Version 1.0). Rheinische Friedrich-Wilhelms-Universität Bonn, Ruhr-Universität Bochum.

Varada Kolhatkar, Hanhan Wu, Luca Cavasso, Emilie Francis, Kavan Shukla, and Maite Taboada. 2019. The sfu opinion and comments corpus: A corpus for the analysis of online news comments. *Corpus Pragmatics*, 4(2):155–190.

Esther Konig and Wolfgang Lezius. 2000. A description language for syntactically annotated corpora. In *COLING 2000 Volume 2: The 18th International Conference on Computational Linguistics*.

Natalia Konstantinova, Sheila CM De Sousa, Noa P Cruz Díaz, Manuel J Mana López, Maite Taboada, and Ruslan Mitkov. 2012. A review corpus annotated for negation, speculation and their scope. In *Lrec*, pages 3190–3195.

Thomas Krause and Martin Klotz. 2025. Annatto.

Thomas Krause, Ulf Leser, and Anke Lüdeling. 2016. graphannis: A fast query engine for deeply annotated linguistic corpora. *Journal for Language Technology and Computational Linguistics*, 31(1):1–25.

Thomas Krause and Amir Zeldes. 2014. Annis3: A new architecture for generic corpus query and visualization. *Digital Scholarship in the Humanities*, 31(1):118–139.

Alexander Leonhardt, Giuseppe Abrami, Daniel Baumartz, and Alexander Mehler. 2023. Unlocking the heterogeneous landscape of big data NLP with DUUI. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 385–399, Singapore. Association for Computational Linguistics.

Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen tau Yih, Tim Rocktäschel, Sebastian Riedel, and Douwe Kiela. 2021. Retrieval-augmented generation for knowledge-intensive nlp tasks. *Preprint*, arXiv:2005.11401.

Wolfgang Lezius. 2002. *Ein Suchwerkzeug für syntaktisch annotierte Textkorpora*. Ph.d. thesis, University of Stuttgart, Stuttgart, Germany.

Chen Ling, Xujiang Zhao, Jiaying Lu, Chengyuan Deng, Can Zheng, Junxiang Wang, Tanmoy Chowdhury, Yun Li, Hejie Cui, Xuchao Zhang, Tianjiao Zhao, Amit Panalkar, Dhagash Mehta, Stefano Pasquali, Wei Cheng, Haoyu Wang, Yanchi Liu, Zhengzhang Chen, Haifeng Chen, Chris White, Quanquan Gu, Jian Pei, Carl Yang, and Liang Zhao. 2024. Domain specialization as the key to make large language models disruptive: A comprehensive survey. *Preprint*, arXiv:2305.18703.

Johann-Mattis List, Nathaniel A. Sims, and Robert Forkel. 2021. Toward a sustainable handling of interlinear-glossed text in language documentation. *ACM Trans. Asian Low-Resour. Lang. Inf. Process.*, 20(2).

Sasha Luccioni, Yacine Jernite, and Emma Strubell. 2024. Power hungry processing: Watts driving the cost of ai deployment? In *Proceedings of the 2024 ACM Conference on Fairness, Accountability, and Transparency*, FAccT '24, page 85–99, New York, NY, USA. Association for Computing Machinery.

Max Planck Institute. 2008. The leipzig glossing rules: Conventions for interlinear morpheme-by-morpheme glosses.

Dirk Merkel. 2014. Docker: lightweight linux containers for consistent development and deployment. *Linux journal*, 2014(239):2.

Roser Morante and Walter Daelemans. 2012. ConanDoyle-neg: Annotation of negation cues and their scope in conan doyle stories. In *Proceedings of the Eighth International Conference on Language Resources and Evaluation (LREC'12)*, pages 1563–1568, Istanbul, Turkey. European Language Resources Association (ELRA).

Julian Moreno-Schneider, Rémi Calizzano, Florian Kintzel, Georg Rehm, Dimitris Galanis, and Ian Roberts. 2022. Towards practical semantic interoperability in NLP platforms. In *Proceedings of the 18th Joint ACL - ISO Workshop on Interoperable Semantic Annotation within LREC2022*, pages 118–126, Marseille, France. European Language Resources Association.

George Moroz. 2021. *lingglosses: Linguistic glosses and semi-automatic list of glosses creation*.

Joakim Nivre, Daniel Zeman, Filip Ginter, and Francis Tyers. 2017. Universal Dependencies. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Tutorial Abstracts*, Valencia, Spain. Association for Computational Linguistics.

Sampo Pyysalo, Filip Ginter, Juho Heimonen, Jari Björne, Jorma Boberg, Jouni Järvinen, and Tapio Salakoski. 2007. Bioinfer: a corpus for information extraction in the biomedical domain. *BMC Bioinformatics*, 8(1).

Thomas Schmidt and Kai Wörner. 2014. 402exmaralda. In *The Oxford Handbook of Corpus Phonology*. Oxford University Press.

Emil Stenström. 2016. conllu: A python library for conll-u parsing. Accessed: 2025-03-27.

Thinh Hung Truong, Timothy Baldwin, Trevor Cohn, and Karin Verspoor. 2022. Improving negation detection with negation-focused pre-training. *Preprint*, arXiv:2205.04012.

Iulia Turc, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Well-read students learn better: The impact of student initialization on knowledge distillation. *CoRR*, abs/1908.08962.

Veronika Vincze, György Szarvas, Richárd Farkas, György Móra, and János Csirik. 2008. The bioscope corpus: biomedical texts annotated for uncertainty, negation and their scopes. *BMC Bioinformatics*, 9(S11).

Klaus-Peter Wegera, Hans-Joachim Solms, Ulrike Demske, and Stefanie Dipper. 2021. Reference Corpus of Early New High German (1350–1650) (Version 1.0). Ruhr-Universität Bochum, Martin-Luther-Universität Halle-Wittenberg, Universität Potsdam.

Xiaokun Wu, Min Chen, Wanyi Li, Rui Wang, Limeng Lu, Jia Liu, Kai Hwang, Yixue Hao, Yanru Pan, Qingguo Meng, Kaibin Huang, Long Hu, Mohsen Guizani, Naipeng Chao, Giancarlo Fortino, Fei Lin, Yonglin Tian, Dusit Niyato, and Fei-Yue Wang. 2025. LLM fine-tuning: Concepts, opportunities, and challenges. *Big Data Cogn. Comput.*, 9(4):87.

Yang Xiao, Yunke Li, Shaoyujie Chen, Hayden Barker, and Ryan Rad. 2025. Do you actually need an llm? rethinking language models for customer reviews analysis. *Artificial Intelligence Review*, 58(10).

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, Chujie Zheng, Dayiheng Liu, Fan Zhou, Fei Huang, Feng Hu, Hao Ge, Haoran Wei, Huan Lin, Jialong Tang, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jing Zhou, Jingren Zhou, Junyang Lin, Kai Dang, Keqin Bao, Kexin Yang, Le Yu, Lianghao Deng, Mei Li, Mingfeng Xue, Mingze Li, Pei Zhang, Peng Wang, Qin Zhu, Rui Men, Ruize Gao, Shixuan Liu, Shuang Luo, Tianhao Li, Tianyi Tang, Wenbiao Yin, Xingzhang Ren, Xinyu Wang, Xinyu Zhang, Xuancheng Ren, Yang Fan, Yang Su, Yichang Zhang, Yinger Zhang, Yu Wan, Yuqiong Liu, Zekun Wang, Zeyu Cui, Zhenru Zhang, Zhipeng Zhou, and Zihan Qiu. 2025. Qwen3 technical report. *Preprint*, arXiv:2505.09388.

Lars Erik Zeige, Gohar Schnelle, Martin Klotz, Karin Donhauser, Jost Gippert, and Rosemarie Lühr. 2025. Deutsch Diachron Digital - Referenzkorpus Altdeutsch (Version 1.2). Humboldt-Universität zu Berlin.

Amir Zeldes, Florian Zipser, and Arne Neumann. 2013. Paula xml documentation.

Florian Zipser and Laurent Romary. 2010. A model oriented approach to the mapping of annotation formats using standards. In *Workshop on Language Resource and Language Technology Standards, LREC 2010*, La Valette, Malta.

A Conversion Capabilites

Integrated Third-Party Conversion Tools

COMPONENT	Module	File Format
Pepper		
	RSD	.rsd
	CoraXML	.xml
	PAULA	.xml
	Gate	.xml
	TigerXML	.xml
	Treetagger	.txt
	Spreadsheet[†]	.xls | .xlsx
	WebannoTSV	.tsv
	GeTa	.json
	Toolbox	.txt
	CoNLL[†]	.conll | .txt
	TEI[†]	.tei | .xml
	GenericXML	.xml
	TCF	.xml
	RST	.rst
	Elan	.eaf
	PTB	.mrg
	EXMARaLDA[†]	.exb | .exs | .exf
	CoNLLCoref	.conll
	SaltXML	.xml
	Text	.txt
	MMAX2[†]	.mmax | .xml
	GrAF[†]	.xml | .graf
	GraphAnno	.json
	UAM[†]	.txt | .xml | .uam
Annatto		
	CoNLL-U	.conllu
	EXMARaLDA	.exb
	GraphML	.xml
	meta	.csv
	opus	.xml
	PTB	.mrg
	relANNIS-3.3[†]	.annis | .version ...
	SaltXML	.xml
	table	.csv
	textgrid	.txt
	toolbox	.txt
	treetagger	.txt
	whisper	.json
	Spreadsheet	.xlsx
	GenericXml	.xml
openConv		
	text	.txt
	TEI[†]	.tei | .xml
	alto	.alto
	doc	.doc
	docx	.docx
	HTML	.html

Dedicated READER COMPONENTS

COMPONENT	Module	File Format
ANNIS		
	relANNIS-3.3[†]	.annis | .version ...
Sketch Engine		
	SE	.csv
SOCC		
	SOCC	.tsv
SFU		
	SFU	.xml
Conan Doyle		
	CD-SCO	.txt[*]
PB-FOC		
	PB-FOC	.txt[*]
Bioscope		
	BS	.csv
Deeptutor Neg		
	DT-Neg	.txt
Leipzig Glossing		
	LGR	.tex

[*]CoNLL dialect
[†]Some formats span multiple file types; the most relevant are listed.

Table 4: Overview of all the specific corpus formats that each component is capable of converting. On the left side, a list of integrated third-party conversion tools is shown. Each COMPONENT supports multiple corpus formats, with some tools offering overlapping functionality. On the right side, a list of dedicated READER COMPONENTS and the specific file formats they support is displayed. Some COMPONENTS may share the same file format extension, but often require distinct internal structures. For instance, the CSV format used by Sketch Engine differs significantly from that of the Bioscope corpus.

B Corpus IDs

ID	Dataset Name
A	DDD-AD-Z-Notker_Kleinere-De_Partibus_logice
B	DDD-AD-Physiologus
C	DDD-AD-Z-Notker-Psalmen-Glossen
D	DDD-AD-Z-Notker_Kleinere-De_Musica
E	DDD-AD-Murbacher_Hymnen
F	DDD-AD-Murbacher_Hymnen_Latein
G	DDD-AD-Genesis
H	DDD-AD-Isidor_Latein
I	DDD-AD-Isidor
J	DDD-AD-Z-Notker_Kleinere-Syllogismus
K	DDD-AD-Z-Notker_Kleinere-Ars_Rhetorica
L	DDD-AD-Monsee
M	DDD-AD-Z-Notker_Cantica
N	DDD-AD-Benediktiner_Regel
O	DDD-AD-Benediktiner_Regel_Latein
P	DDD-AD-Kleinere_Altsächsische_Denkmäler
Q	DDD-AD-Z-Notker_Boethius-De_Interpretatione
R	DDD-AD-Kleinere_Althochdeutsche_Denkmäler
S	DDD-AD-Z-Notker_Boethius-Categoriae
T	DDD-AD-Tatian_Latein
U	DDD-AD-Z-Notker-Martianus_Capella
V	DDD-AD-Tatian
W	DDD-AD-Heliand
X	DDD-AD-Otfrid
Y	DDD-AD-Z-Notker_Boethius-De_Consolatione_philosophiae

Table 5: A mapping of document IDs to dataset names for datasets contained in the reference corpus of Old High German (Zeige et al., 2025).

C Prompts

C.1 Qwen3 Few-Shot Prompt

~~~
System Prompt:
------------------------------------------------------------
**Task:**
Identify negation cues in the input sentence.
**Instructions:**
* Input is a JSON object with `"sent"` containing a tokenized sentence (list of strings).
* Output must be a JSON object with `"cue_mask"`, an array of the same length as `"sent"`.
* Each element in `"cue_mask"` should be:
  * `1` if the corresponding token is a negation cue.
  * `0` otherwise.
* A sentence may contain multiple negation cues, or none.
**Negation cues include (but are not limited to):**
* **Explicit negation words:** *not, n't, never, no, without, nothing, none, nobody, nowhere, neither, nor*
* **Weak/limiting negators:** *hardly, barely, scarcely*
* **Negative adjectives/adverbs:** *improper, inadmissable, impossible, invalid, unacceptable, incorrect*
* **Negative verbs:** *deny, lack, fail, forbid, prohibit, exclude*
* **Other context-dependent words or phrases** that function as negation cues should also be marked.
**Examples:**
Input:
```json
{"sent": ["But", ",", "in", "the", "second", "place", ",", "why", "did", "you", "not", "come", "at", "once", "?", "''"]}
```
Output:
```json
{"cue_mask": [0,0,0,0,0,0,0,0,0,0,1,0,0,0,0,0]}
```
Input:
```json
{"sent": ["It", "is", "most", "improper", "--", "most", "outrageous", "."]}
```
Output:
```json
{"cue_mask": [0,0,0,1,0,0,0,0]}
```
Input:
```json
{"sent": ["But", "no", "one", "can", "glance", "at", "your", "toilet", "and", "attire", "without", "seeing", "that", "your", "disturbance", "dates",
↪  "from", "the", "moment", "of", "your", "waking", ".", "''"]}
```
Output:
```json
{"cue_mask": [0,1,0,0,0,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0]}
```
Input:
```json
{"sent": ["You", "do", "n't", "mean", "--", "you", "do", "n't", "mean", "that", "I", "am", "suspected", "?", "''"]}
```
Output:
```json
{"cue_mask": [0,0,1,0,0,0,0,1,0,0,0,0,0,0,0]}
```
Input:
```json
{"sent": ["``", "Good", ",", "Watson", ",", "very", "good", "--", "but", "quite", "inadmissable", "."]}
```
Output:
```json
{"cue_mask": [0,0,0,0,0,0,0,0,0,0,0,1,0]}
```

... etc.

User Prompt:
------------------------------------------------------------
Input:
```json
{"sent": $sent}
```
Output:
~~~

Figure 5: Few-shot prompt template used for the Qwen3-0.6B experiment.

C.2 GPT-5 ICL Prompt

~~~
Input:
```json
{"sent": $sent_1}
```
Output:
```json
{"cue_mask": $cue_mask_1}
```
Input:
```json
{"sent": $sent_2}
```
Output:
```json
{"cue_mask": $cue_mask_2}
```

...

Input:
```json
{"sent": $sent_n}
```
Output:
```json
{"cue_mask": $cue_mask_n}
```

Given the examples above, please label the following test cases and return the output in a json format with "test_id" as the
↪  key and the predicted cue_mask as the value:

```json
{"sent": $sent_1, "test_id": $test_id_1}
```

```json
{"sent": $sent_2, "test_id": $test_id_2}
```
...

```json
{"sent": $sent_m, "test_id": $test_id_m}
```
~~~

Figure 6: Pure ICL prompt template used for the GPT-5 experiment.