Amharic DBpedia Chapter: A Knowledge Graph for a Low-Resource Language

Hizkiel Mitiku Alemayehu[1], **Tilahun Abedissa Taffa**[2,3], **Meti Bayisa**[4], **Andargachew Asfaw**[4], **Hamada M. Zahera**[1], **Ricardo Usbeck**[2], **and Axel-Cyrille Ngonga Ngomo**[1]

[1]Data Science Group, Heinz Nixdorf Institute, Paderborn University, Paderborn, Germany
[2]Artificial Intelligence and Explainability, Leuphana University of Lüneburg, Germany
[3]Department of Informatics, University of Hamburg, Germany
[4]Addis Ababa University, Addis Ababa, Ethiopia
{hizkiel, hamada.zahera, axel.ngonga}@uni-paderborn.de
tilahun.taffa@uni-hamburg.de, andargachew.asfaw@aau.edu.et, ricardo.usbeck@leuphana.de

Abstract

DBpedia is a community-driven project that extracts structured knowledge from Wikipedia via language-specific chapters. We present the first steps toward the Amharic DBpedia chapter by extending the DBpedia Extraction Framework (DEF) to support Amharic Wikipedia, including language-specific components such as Ethiopian date parsers, an Ethiopian–Gregorian calendar converter, an Arabic–Ge'ez number converter, and Amharic template mappings, together with automated extraction pipelines and the publication of the resulting knowledge graph through a live website, DBpedia Databus collection, and query endpoints. For mapping, we evaluate the zero-shot NLLB-200 translation model on Amharic infobox property names, achieving a BLEU score of 45.31. For ontology alignment, we link mapped properties to DBpedia ontology properties across 58 DBpedia classes and benchmark multilingual encoders with Amharic support, including Afro-XLM-R Base, XLM-R Base, and Amharic fine-tuned mBERT. The fine-tuned Afro-XLM-R model achieves 92.1% Top-10 accuracy and strong ranking performance, as measured by Mean Reciprocal Rank (MRR). We release all resources developed for the Amharic DBpedia chapter, including the Ethiopian date parser, Ethiopian–Gregorian calendar converter, Arabic–Ge ez numeral converter, Amharic template mappings, automated extraction workflows, and the resulting Amharic DBpedia knowledge graph with public access via the DBpedia Databus collection, Tentris query endpoint, and the live website at am.dbpedia.org.

Keywords: DBpedia, Knowledge Graph, Semantic Web

1. Introduction

Amharic is one of the national working languages of Ethiopia and is also used as a working language in several regional states (Tesfagerish et al., 2022). It belongs to the Afro-Asiatic language family, specifically the Ethio-Semitic branch, and is closely related to Tigrinya and Tigre (Hudson, 2013). Like other Semitic languages, Amharic exhibits characteristic linguistic features, such as root-and-pattern morphology, in which words are formed by inserting vowels into consonantal roots to convey different grammatical meanings (Hudson, 2013). The language is written in the Ge ez script (the characters known as Fidäl), which follows the Abugida (አቡጊዳ) writing system (Daniels and Bright, 1996).

The script contains 33 basic consonant characters, each appearing in seven vocalic forms (orders), resulting in more than 230 distinct syllabic symbols (Daniels and Bright, 1996). The writing system is read from left to right and is also used for other Ethiopian Semitic languages such as Tigrinya (Daniels and Bright, 1996). Due to its long historical use in governance, literature, and cultural life, Amharic plays a central role in Ethiopia's linguistic and cultural landscape (Encyclopaedia Britannica, 2024).

A community-driven project, DBpedia extracts structured knowledge from Wikipedia and publishes it as an open Knowledge Graph (KG) (Auer et al., 2007). The goal of DBpedia is to transform the semi-structured information in Wikipedia into machine-readable, structured data that can be queried and integrated with other datasets on the Semantic Web. DBpedia has become one of the central hubs of the Linked Open Data cloud, interlinking numerous datasets and enabling KG applications such as semantic search, question answering, and data integration (Auer et al., 2007).

The DBpedia KG is generated using the *DBpedia Extraction Framework (DEF)*, a software framework that processes Wikipedia content dumps and converts them into structured RDF triples. As illustrated in Figure 1, the extraction framework follows a multi-stage pipeline. First, the **input stage** obtains Wikipedia data either from periodic Wikipedia dump files or through the MediaWiki API. These dumps contain the complete textual and structural content of Wikipedia articles, including infoboxes, categories, templates, and links.

Proceedings of the Fifteenth Language Resources and Evaluation Conference (LREC 2026), pages 7894–7904
11–16 May 2026. ©ELRA Language Resources Association (ELRA), 2026

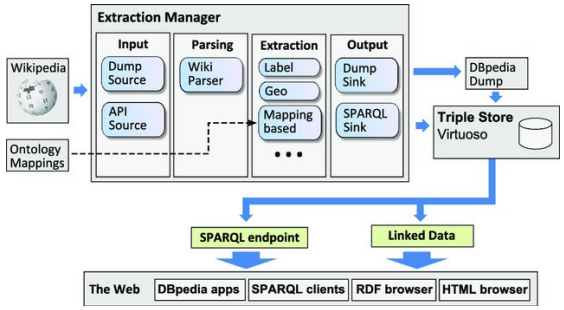

Figure 1: Overview of the DBpedia Extraction Framework. Image source (Morsey et al., 2012; Lehmann et al., 2015).

In the **parsing stage**, the DEF processes Wikipedia markup using a Wiki parser that interprets MediaWiki syntax and identifies structural elements such as infobox templates, categories, and links. This step converts the raw wiki markup into an intermediate representation suitable for structured data extraction. In the subsequent **extraction stage**, the framework applies a set of specialized extractors to generate structured information from the parsed content. DBpedia primarily employs two extraction approaches: *mapping-based extraction* and *raw extraction*. Mapping-based extraction uses community-defined ontology mappings that align Wikipedia infobox templates with the DBpedia ontology, enabling the generation of consistent RDF triples across articles (Lehmann et al., 2015). In contrast, raw extraction directly retrieves information such as labels, page links, categories, geographic coordinates, and other metadata from Wikipedia content without relying on predefined ontology mappings (Morsey et al., 2012). Finally, in the **output stage**, the generated RDF triples are written to different sinks, including dump files and SPARQL endpoints. These datasets are stored in triple stores such as Virtuoso, allowing efficient storage and querying of large-scale KGs. The resulting DBpedia dataset is then published as Linked Open Data. It can be accessed through SPARQL endpoints, RDF browsers, and web interfaces, enabling the transformation of collaboratively curated Wikipedia content into a KG that supports various semantic web applications (Auer et al., 2007; Lehmann et al., 2015).

DBpedia is organized into language-specific chapters that localize structured knowledge for different linguistic communities by extracting information from the corresponding language editions of Wikipedia. Each chapter adapts the DBpedia extraction pipeline to the linguistic and structural characteristics of a particular language, enabling the publication of KGs that reflect local knowledge. These chapters are built using the DEF[1], which combines *mapping-based extraction* from Wikipedia infoboxes with *raw extraction* from semi-structured text (Morsey et al., 2012). Mapping-based extraction relies on ontology mappings that align Wikipedia templates with DBpedia ontology properties. In contrast, raw extraction retrieves additional information such as labels, categories, links, and geographic coordinates directly from Wikipedia markup. Through these mechanisms, language chapters play an important role in expanding DBpedia's multilingual coverage and ensuring that knowledge from different Wikipedia communities becomes available as Linked Open Data.

While the DEF has had a broad impact for high-resource languages, its coverage of low-resource languages remains limited (Cojan et al., 2013; Cabrio et al., 2014; Al-Feel, 2013), restricting access to structured knowledge and hindering downstream applications. To address this gap, we extend the DEF to support Amharic. Figure 2 presents an example of triples extracted from the Amharic DBpedia, offering insight into the types of facts represented in the knowledge graph. We believe this work can serve as a practical example for extending DBpedia to other low-resource languages.

Contributions

This paper presents the first steps toward the Amharic DBpedia chapter. Amharic poses unique linguistic and technical challenges, including the Ge'ez (Ethiopic) script and the Ethiopian calendar, as well as limited tooling and community resources (Chali and Parapatics, 2023; Al-Feel, 2015; Adnew and Liang, 2024; Nigatu et al., 2024). The main contributions of this work are:

1. **Extension of the DBpedia Extraction Framework to Amharic.** We extend the DBpedia Extraction Framework (DEF) with Amharic-specific processing components, including an Ethiopian date parser, an Ethiopian–Gregorian calendar converter, and an Arabic–Ge ez numeral converter, enabling structured extraction from Amharic Wikipedia.

2. **Amharic mapping resources.** We create Amharic-specific mapping files within the DBpedia mapping framework to align Amharic Wikipedia template properties with DBpedia ontology properties and enable mapping-based extraction.

3. **Automation for mapping and ontology alignment.** We investigate automation for template-to-ontology mapping by evaluating zero-shot NLLB-200 translation for Amharic infobox properties and benchmarking multilingual encoders for ontology alignment.

[1] https://github.com/dbpedia/extraction-framework

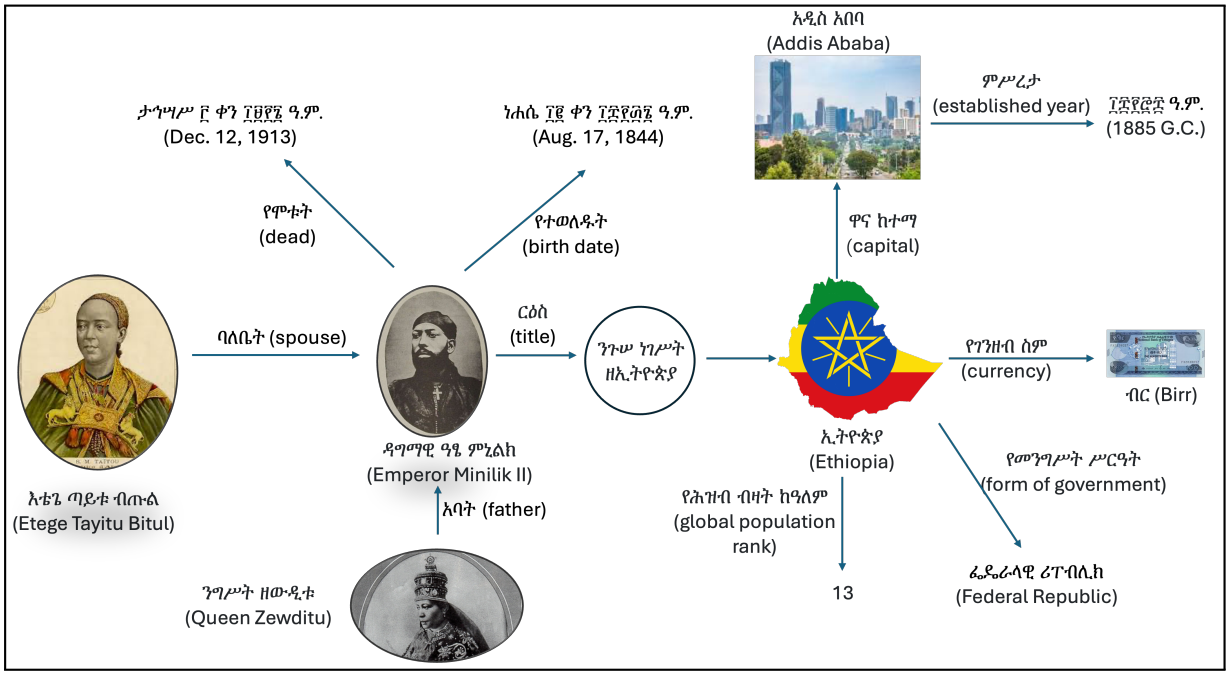

Figure 2: This image presents an excerpt of triples extracted from the Amharic DBpedia, offering insight into the facts in the KG. Highlighted are facts about Ethiopia, such as its currency (Birr), government form (federal republic), global population rank (13th), and capital city (Addis Ababa, established in 1885). As well as historical information about Emperor Menelik II (one of the kings of kings in Ethiopia history), including his spouse Etege Tayitu Bitul, the daughter of Queen Zewditu, and his birth and death dates.

4. **Open resources and infrastructure for Amharic DBpedia.** We release the extracted Amharic DBpedia knowledge graph and supporting resources through public infrastructure, including a live website, a DBpedia Databus collection, a Tentris query endpoint, automation workflows, documentation, and an archived release on Zenodo. Table 1 summarizes resources and infrastructure for the Amharic DBpedia Chapter,

Resource	Link
Live Website	am.dbpedia.org
DBpedia Databus Collection	Databus Collection
Tentris Query Endpoint	Query Endpoint
Zenodo Archive	Zenodo DOI
Automation Repository	GitHub Repository
Documentation Wiki	Project Wiki
Ethiopian Date Parser	Date Parser
Ethiopian–Gregorian Converter	Calendar Converter
Amharic Mapping	DBpedia Mapping
Arabic–Ge ez Number Converter	Number Converter
NLLB Translation Resources	Translation Resources

Table 1: Public resources and infrastructure for the Amharic DBpedia chapter.

2. Related works

Over the past two decades, DBpedia's internationalization has expanded through the creation of language-specific chapters, including Arabic, Korean, and Greek, to address the unique linguistic and structural challenges of each language.

Al-Feel (Al-Feel, 2015) describes the creation of a structured, linked-data version of Arabic Wikipedia. They highlight the importance of community involvement, localization efforts, and a long-term vision to support Arabic-language in Artificial Intelligence and knowledge-based applications. Kim et al. (Kim et al., 2010) discuss their efforts to enhance the Korean edition of DBpedia and propose methods to enrich the Korean Wikipedia using data from DBpedia. Their work primarily focuses on improving the extraction framework to better handle non-Latin characters, specifically Korean, by introducing a plug-in system tailored for language-specific extraction. This enhancement led to a significant increase in the amount of structured data extracted from the Korean Wikipedia. Additionally, they explore the automatic translation of infobox data from the English Wikipedia to Korean, aiming to address information imbalances and enrich the Korean Wikipedia's content.

The paper also introduces OntoCloud, an on-

tology constructed to efficiently organize and map infobox templates and their attributes, facilitating better synchronization and integration of information across different language editions of Wikipedia. For Greek, Kontokostas et al. (Kontokostas et al., 2012) adapted the DBpedia framework to support the Greek language. They focus on enabling International Resource Identifiers (IRIs) that allow Greek characters to be used directly in resource URIs. They developed new extractors tailored to the Greek Wikipedia to address issues with template mapping and content normalization.

3. Challenges in Building the Amharic DBpedia Chapter

The extraction framework builds language chapters via two complementary pipelines: mapping-based extraction, which converts infobox parameters into RDF triples using community-authored mappings, and raw extraction, which mines semi-structured content from article text. In practice, most high-precision triples come from the mapping-based path; therefore, any drift between page infobox usage and the corresponding template/mapping specification results in failures. Likewise, language-specific formatting (e.g., calendars, numerals) that the parsers cannot normalize will either drop facts or introduce invalid values. The following challenges describe how these failure modes manifest for Amharic.

3.1. Inconsistencies in Wikipedia Infoboxes

A significant challenge is the inconsistency between the properties of the Wikipedia infobox template and their use on Amharic Wikipedia pages. This mismatch directly affects the quality and *coverage* of triples we can extract via mappings. Inconsistencies between template design and page usage are a known issue in Wikipedia-based extraction pipelines (Morsey et al., 2012).

- **Case 1 (outdated property on page).** After a template update, the parameter and HTML tag are updated, but the page instance still references a previous property name; the parameter is therefore not rendered in the infobox, and the mapping-based extractor returns no triple.
- **Case 2 (mismatch inside include tags).** The template parameter name and page usage are updated correctly, but an internal related tag within the template still uses the old label. Even though the template and page appear correct, the outdated include tag prevents rendering, so the extractor never sees the value, and no triple is generated. In contrast, parameters already aligned with the template definition render and extract, usually making this problem difficult to detect automatically.

In a preliminary analysis across multiple Amharic Wikipedia pages, recurring instances of such inconsistencies were identified. Because each mismatch suppresses all triples for the affected field, even a small number of cases yields a disproportionate loss of extracted knowledge. Moreover, locating and repairing such inconsistencies requires manual page-level or template-level edits, a tedious and time-consuming maintenance task for low-resource language communities (Nigatu et al., 2024).

We addressed the inconsistencies by (i) directly correcting the affected pages and (ii) adding redirection/aliasing in template definitions to preserve legacy parameter names. While effective, both approaches are time-consuming and do not scale.

3.2. Language-Specific Challenges

Beyond structural issues, Amharic introduces language-specific normalization problems for dates and numerals.

A major challenge in extracting structured data from Amharic Wikipedia stems from differences between the Ethiopian and Gregorian calendars. The Ethiopian calendar consists of 13 months (12 months of 30 days and a short 13th month of 5 or 6 days), whereas the Gregorian calendar follows a 12-month structure with variable month lengths. As a result, dates appearing in infoboxes may not conform to Gregorian expectations; for example, a value such as `1.13.2017` is syntactically valid in the Ethiopian calendar but invalid under Gregorian conventions. Consequently, parsers designed for Gregorian date formats fail when encountering a 13th month. Additionally, the Ethiopian New Year (Meskerem 1) occurs on September 11 (or September 12 in Gregorian leap years), shifting the temporal alignment of events across the two systems (Tamrat, 2008). These structural differences complicate automatic parsing and normalization of date values within the DBpedia Extraction Framework, which assumes Gregorian date formats.

Another challenge concerns the representation of numbers using Ge ez numerals. Unlike the positional decimal system used in Arabic numerals, the Ge ez numeral system combines additive and multiplicative symbols, where markers such as ፻ (100) and ፼ (10,000) function as hierarchical multipliers within a number expression. Consequently, a numeral such as ፺፱፻፺፼፸፲፻፳፭ cannot be interpreted through simple token mapping or positional parsing. Instead, the numeral's structure encodes nested multiplicative relationships that must be resolved to obtain the corresponding numeric value.

This non-positional representation poses difficulties for information extraction systems designed for languages that use Arabic numerals, requiring specialized parsing strategies to interpret numeric expressions in Amharic Wikipedia correctly. Both calendar and numeral differences illustrate how language-specific conventions introduce additional complexity when adapting the DBpedia Extraction Framework to Amharic.

4. The Amharic DBpedia Chapter

To create the Amharic DBpedia chapter, we first set up custom extractors and language tools. We adapted the `mapping-based` extractor to process the Ge'ez (Ethiopic) script and to handle Ethiopian calendar formats (e.g., `2016` ዓ.ም).

Next, we updated the mappings server to include Amharic–English predicate links for core ontologies such as `dbo:Person`, `dbo:Place`, and `dbo:Event`. For example, we map the Amharic infobox attribute **የተወለዱበት ቀን** (birth date) to `dbo:birthDate`. This ensures that high-precision infobox values are consistently converted into RDF triples.

4.1. Creating new mappings

In Wikipedia, infoboxes present key facts in a tabular summary defined by templates (Auer et al., 2007). Figure **??** shows an Amharic infobox for *Africa*. Our mapping layer aligns such infobox attributes with DBpedia ontology properties, enabling reliable, schema-aware extraction.

To extract data from the Wikipedia infobox, we manually created and added new Amharic mapping files to the DBpedia mappings wiki. The current mapping statistics for Amharic are summarized in Table 2.

Statistic	%	Count
Templates mapped	100.00	84 / 84
Properties mapped	77.29	2392 / 3095
Template occurrences mapped	100.00	3958 / 3958
Property occurrences mapped	99.15	19799 / 19968

Table 2: Mapping statistics for Amharic (am): percentage and count of (1) *templates mapped* — distinct infobox templates linked to DBpedia mappings, (2) *properties mapped* — unique template parameters assigned to ontology properties, (3) *template occurrences mapped* — total uses of mapped templates across Wikipedia pages, and (4) *property occurrences mapped* — total occurrences of mapped properties across Amharic Wikipedia.

We continued by extending the framework for Amharic by configuring extractors and adding language-aware parsing components to extract structured knowledge from the Amharic Wikipedia.

4.2. Extending the DBpedia Extraction Framework

We extended the DBpedia Extraction Framework (DEF) with Amharic-aware parsers and introduced additional extractors by adapting components from the English configuration. These extractors and parsers are included in the default Amharic DEF setup.

- **GeoCoordinate Parser** — extracts geographic coordinates (Auer et al., 2007).
- **Date Time Parser** — standardizes dates/times for KG ingestion (Auer et al., 2007).
- **Date Interval Mapping** — captures time ranges for events (Auer et al., 2007).
- **Disambiguation Extractor** — processes disambiguation pages (Auer et al., 2007).
- **Gender Extractor** — adds gender information where available (Auer et al., 2007).
- **Homepage Extractor** — links entities to official websites (Auer et al., 2007).
- **Image Extractor** — retrieves representative images (Auer et al., 2007).
- **Infobox Extractor** — converts infobox fields into triples (Auer et al., 2007).
- **Topical Concepts Extractor** — identifies salient concepts (Auer et al., 2007).

The table 3 shows the number of unique RDF triples, the extractor used, and the extracted data type.

Table 4 provides a general summary of extracted data. The data was extracted from the August 8th, 2025, Wikipedia dump.

4.3. Handling Amharic-Specific Nuances

Calendar conversion. Algorithm 1 shows the date conversion procedure. It converts a date from the Ethiopic (Ethiopian) calendar into the Gregorian calendar using a two-stage procedure. First, the Ethiopic date is converted into a Julian Day Number (JDN), which represents a continuous count of days from a fixed reference point. Second, the JDN is transformed into the corresponding Gregorian date.

The input consists of an Ethiopic year Y_e, month M_e, and day D_e, where $1 \leq M_e \leq 13$. In the Ethiopic calendar, the first twelve months each contain 30 days, while the thirteenth month (Pagume)

Extractor	# Unique Triples
Category Extractor	19,250
Template Extractor (article-template)	975
Template Extractor (not-nested)	19,976
Category Labels Extractor	1,103
Commons Links Extractor	105
Disambiguation Extractor	241
External Links Extractor	4,405
Geo Extractor (mapping-based)	192
Geo Extractor (no-mapping)	1,008
Page Extractor (home page)	66
Infobox Extractor	39,892
Infobox Definition Extractor	1,508
Instance Types Extractor (transitive)	21,608
Instance Types Extractor	5,063
Language Links Extractor	340
Labels Extractor	22,917
Mapping Extractor	192
Mapping Extractor (uncleaned)	5,658
Link Graph Extractor (out-degree)	22,918
Page Extractor (page IDs)	26,648
Page Metrics Extractor (page length)	22,917
Page Links Extractor	147,062
Redirect Extractor	6,936
Revision Extractor (IDs)	26,648
Revision Extractor (URIs)	26,648
Category Extractor (SKOS)	3,780
Property Extractor	561
Template Extractor (parameters)	8,032
Topic Extractor	53
Wikipedia Links Extractor	91,668
Total	**528,370**

Table 3: Summary of extractors and number of unique triples.

Statistic	Value
Triples	596,351
Distinct subjects	68,937
Distinct predicates	1,016
Distinct objects	230,955
Avg. out-degree (IRIs only)	5.163
Redirects (`dbo:wikiPageRedirects`)	6,936

Table 4: Graph summary (Amharic DBpedia snapshot; dump-based extraction). "Avg out-degree (IRIs only)" excludes literal objects.

contains five days in regular years and six days in leap years.

The algorithm begins by defining the Ethiopic epoch E_0, which corresponds to the Julian Day Number of the first day of the Ethiopic calendar (1 Meskerem 1). It also defines a leap-year function IsLeap(Y_e), where a year is considered a leap year if $(Y_e + 1) \bmod 4 = 0$. This determines whether Pagume contains five or six days.

Before performing the conversion, the algorithm verifies that the input date is valid. If the month is Pagume ($M_e = 13$) and the day exceeds the maximum number of days allowed in that month (five or six depending on the leap year), the algorithm returns an invalid result.

The Ethiopic date is then converted into a Julian Day Number by computing the total number of days that have elapsed since the Ethiopic epoch. This calculation includes the days contributed by previous years, leap days, completed months in the current year, and the current day within the month.

Once the Julian Day Number has been obtained, the algorithm converts it into a Gregorian date using the Fliegel–Van Flandern (Fliegel and Van Flandern, 1968) method. This arithmetic procedure derives the Gregorian year, month, and day through a sequence of integer operations based on the JDN.

Finally, the algorithm returns the corresponding Gregorian date (Y_g, M_g, D_g).

Algorithm 1 Ethiopic (Ethiopian) Date Conversion to Gregorian

Require: Ethiopic date (Y_e, M_e, D_e) where $1 \le M_e \le 13$
Ensure: Gregorian date (Y_g, M_g, D_g)
Define Ethiopic epoch E_0 (JDN of 1 Meskerem 1)
Define IsLeap$(Y_e) = ((Y_e + 1) \bmod 4 = 0)$
if $M_e = 13$ **and** $D_e > 5 +$ IsLeap(Y_e) **then return** None ▷ invalid Pagume day
end if
$J \leftarrow E_0 + 365 \cdot (Y_e - 1) + \lfloor \frac{Y_e - 1}{4} \rfloor + 30 \cdot (M_e - 1) + (D_e - 1)$ ▷ convert Ethiopic → JDN
▷ Fliegel–Van Flandern: JDN → Gregorian
$l \leftarrow J + 68569$
$n \leftarrow \lfloor \frac{4l}{146097} \rfloor$; $l \leftarrow l - \lfloor \frac{146097n+3}{4} \rfloor$
$i \leftarrow \lfloor \frac{4000(l+1)}{1461001} \rfloor$; $l \leftarrow l - \lfloor \frac{1461i}{4} \rfloor + 31$
$j \leftarrow \lfloor \frac{80l}{2447} \rfloor$; $D_g \leftarrow l - \lfloor \frac{2447j}{80} \rfloor$
$l \leftarrow \lfloor \frac{j}{11} \rfloor$; $M_g \leftarrow j + 2 - 12l$
$Y_g \leftarrow 100(n - 49) + i + l$
return (Y_g, M_g, D_g)

Numeral conversion. The algorithm 2 converts a sequence of Ge ez numeral tokens into an Arabic numeral value by interpreting the multiplicative structure of the Ge ez numeral system. The input is an array of tokens A, where each token has already been mapped to its corresponding integer value. The algorithm also defines a set of multiplicative markers $M = \{100, 10000\}$, which represent the special Ge ez numerals used to denote hundreds and ten-thousands.

The conversion is performed by the recursive function Calc$(start, end)$, which evaluates the numeric value represented by the subsequence of tokens between the indices $start$ and end. If the subsequence is empty, the function returns None. If the subsequence contains only one token, the value of that token is returned directly.

For longer sequences, the algorithm first scans the token range from right to left to identify indices

corresponding to multiplier symbols (i.e., tokens whose value is in M). These multipliers act as structural separators in the Ge ez numeral system.

For each multiplier position i, the sequence is divided into a left part and a right part. The left part represents the coefficient that multiplies the multiplier value, while the right part represents the remainder of the number. The algorithm recursively evaluates both parts using CALC. If the left part is absent, it defaults to 1, reflecting the implicit multiplier in expressions such as "$\times 100$". If the right part is absent, it defaults to 0.

To ensure valid numeral structure, the algorithm verifies that the coefficient preceding a multiplier does not exceed the multiplier itself. Specifically, if the multiplier is 100, the left value must be less than 100, and if the multiplier is 10000, the left value must be less than 10000. If this condition is violated, the algorithm returns NONE, indicating an invalid numeral form.

If the multiplier structure is valid, the numeric value is computed as the multiplier value multiplied by the left coefficient, plus the value of the right subsequence. If no multiplier tokens are found in the range, the sequence is interpreted as a purely additive numeral, and the result is obtained by summing all token values.

This recursive procedure allows the algorithm to correctly evaluate both simple additive forms and compound multiplicative forms in the Ge ez numeral system.

Algorithm 2 Ge'ez Numeral to Arabic Conversion

Require: Token array A (each token mapped to $\mathbb{N}$); multipliers $M = \{10000, 100\}$

function CALC($start, end$)
  if $start > end$ **then**
    return NONE
  end if
  if $start = end$ **then**
    return $A[start]$
  end if
  $S \leftarrow [\,]$ ▷ indices of multipliers, scanned right→left
  for $i \leftarrow end$ **downto** $start$ **do**
    if $A[i] \in M$ **then**
      append i to S
    end if
  end for
  for all $i \in S$ **do** ▷ divide at multiplier node
    $L \leftarrow$ CALC($start, i-1$)
    if $L =$ NONE **then**
      $L \leftarrow 1$
    end if
    $R \leftarrow$ CALC($i+1, end$)
    if $R =$ NONE **then**
      $R \leftarrow 0$
    end if
    if $A[i] = 100$ **and** $L \geq 100$ **then**
      return NONE
    end if
    if $A[i] = 10000$ **and** $L \geq 10000$ **then**
      return NONE
    end if
    return $A[i] \cdot L + R$
  end for
  return $\sum_{k=start}^{end} A[k]$ ▷ no multipliers: additive
end function

5. Automation

The goal of this section is to investigate whether state of the art language models can assist in creating new DBpedia chapters for low-resource languages. Rather than fully automating the process, our objective is to evaluate how reliably these models can support key tasks in the DBpedia mapping workflow, particularly template property translation and ontology property alignment. We explore translation based mapping using a multilingual machine translation model and Ontology Property Alignment using encoders. Through this investigation, we aim to understand the strengths and limitations of current language models.

5.1. Template Property Translation

Creating a new DBpedia chapter requires building template mappings that connect infobox attributes to DBpedia ontology properties. However, writing these mappings manually is time-consuming and error-prone, especially for low-resource languages such as Amharic, where community participation is limited and expert contributors are scarce. To reduce manual effort, we explored translation-based automation for generating Amharic template property candidates.

This work used NLLB-200 (No Language Left Behind), a multilingual neural machine translation model (Team et al., 2022). NLLB-200 was selected for its explicit Amharic support and open availability, enabling a practical, reproducible DBpedia pipeline. Relative to other publicly available Amharic MT baselines, it has demonstrated stronger performance in prior evaluations (Alemayehu et al., 2024; Nigatu et al., 2025; Belay et al., 2022).

Experimental Setup. We compiled a test set of infobox template properties by selecting 10 commonly used Wikipedia templates (e.g., Person, Continents, Kings), each with 25–96 parameters. Each property was independently translated into Amharic by three annotators. Inter-annotator agreement, measured with Fleiss' kappa (Fleiss, 1971), was

$\kappa = 0.73$. For the final reference translation, we applied majority voting: if two or three annotators produced the same translation, that version was selected; if all three differed, a fourth annotator adjudicated. English property names were also translated into Amharic using NLLB-200 in a zero-shot setting.

Results. We assess translation quality using standard metrics BLEU (Papineni et al., 2002), METEOR (Banerjee and Lavie, 2005), ChrF++ (Popović, 2015), TER (Snover et al., 2009). The zero-shot NLLB-200 model achieved the following scores: **BLEU 45.31**, **chrF++ 73.13**, **TER 45.58**, and **METEOR 39.55**. These results indicate that the model captures substantial lexical and semantic fidelity for Amharic in a low-resource scenario.

Error Analysis. Despite promising scores, post-editing is required before integrating translations into mapping files. Typical issues include:

- **Context loss**: relational terms translated too literally.
- **Term inconsistency**: variant phrasings for the same concept across templates.
- **Technical terminology**: domain terms that should remain unchanged.
- **Formatting drift**: structural variations that break mapping syntax.

5.2. Automated Ontology Property Alignment for Amharics

In the previous section, we explored automation for the *template mapping* step by using a translation model to align Amharic template fields with their English counterparts. The present subsection addresses the subsequent step: *ontology alignment*. Concretely, after mapping an Amharic template field such as "**የትውልድ ቀን**" (yetiwilidi k'eni, "date of birth") to the English template field `dateOfBirth`, the next step is to assign the correct DBpedia ontology property (e.g., `dbo:birthDate`). Because Amharic is a low-resource language, performing ontology alignment manually is time-consuming and prone to inconsistencies. Therefore, we investigate whether multilingual pre-trained encoders can automatically align Amharic template properties with DBpedia ontology properties.

Formally, given a template property such as "**የመንግስት ዓይነት**" (yemenigisiti 'ayineti), meaning "type of government" within the class `Country`, the task is to predict the correct DBpedia ontology property (e.g., `dbo:governmentType`) from a set of candidate ontology properties. We formulate this problem as a ranking task over the ontology property set.

To address this, we model ontology alignment as a retrieval problem. Each Amharic template property is represented as a class-aware textual premise, while each candidate DBpedia ontology property is represented by its property text. Sentence-embedding models are trained using a contrastive objective (`MultipleNegativesRankingLoss`) so that correct premise–property pairs are mapped close together in the embedding space, while other properties in the batch act as negatives. During inference, we encode each premise and all candidate ontology properties, compute cosine similarity scores, and rank the ontology properties for each test instance.

We construct an Amharic property-mapping dataset derived from the Amharic Mapping resource and store it in JSONL format containing `premise` and `property_text` pairs. Duplicate pairs are removed prior to training and evaluation. The experiments use predefined training, development, and test splits; summary statistics are reported in Table 5.

Statistic	Value
Total extracted property mappings	2,797
Unique ontology properties	143
Number of DBpedia classes	61
Training examples	2,237
Development examples	280
Test examples	280
Avg. tokens per premise (Amharic)	6.8
Avg. ontology candidates per query	143

Table 5: Dataset statistics.

We selected three multilingual pretrained encoders with confirmed Amharic support. Namely, Afro-XLM-R Base, (Alabi et al., 2022), XLM-R Base (Conneau et al., 2019) and mBERT(Amharic fine-tuned) Adelani (2021) . Unlike generic multilingual benchmarks, these models either include Amharic pretraining or Amharic adaptation

Training and evaluation were performed on a single NVIDIA GeForce RTX 3090 GPU (24GB) with CUDA 12.6 support. The system had 64GB RAM and ran Ubuntu 22.04 LTS. Hyperparameters are summarized in Table 6.

We report both zero-shot and fine-tuned ranking performance using Top-k accuracy, Mean Reciprocal Rank (MRR), and Mean Rank; here, MRR is the average reciprocal of the rank of the correct property (higher is better), and Mean Rank is the average rank position of the correct property (lower is better). The results are summarized in 7.

Zero-shot performance is very low, demonstrating that property alignment cannot be solved with-

Hyperparameter	Value
Training epochs	3
Learning rate	2e–5
Optimizer	AdamW
Batch size	16
Max seq length	256
Negatives per positive	3
Warmup ratio	0.06
Gradient accumulation	1
Random seed	42

Table 6: Training configuration.

Zero-shot ontology mapping performance					
Model	**Top-1**	**Top-5**	**Top-10**	**MRR**	**Mean Rank**
Afro-XLM-R Base	3	20	**11.5**	**3.9**	**53.87**
XLM-R Base	**7**	**4.7**	6.5	3.6	74.84
mBERT (Amharic)	**7**	3.2	6.5	3.6	88
Fine-tuned ontology mapping performance					
Model	**Top-1**	**Top-5**	**Top-10**	**MRR**	**Mean Rank**
Afro-XLM-R Base	**66.9**	**88.9**	**92.1**	**78.8**	**4.51**
mBERT (Amharic)	0.441	55.9	60.2	51.4	23.79
XLM-R Base	0.484	68.5	72.0	58.2	17.25

Table 7: Ontology mapping performance: zero-shot vs. fine-tuned. Zero-shot performance is extremely low, indicating that fine-tuning is necessary for effective results. (Lower is better for Mean Rank.)

out fine-tuning, even with large multilingual encoders.

Fine-tuning dramatically improves performance, with Afro-XLM-R learning class-conditioned property semantics effectively and achieving over 89% Top-5 accuracy, making it a strong candidate for Amharic DBpedia bootstrapping. Overall, fine-tuning pretrained encoders enables accurate and scalable DBpedia ontology alignment for Amharic. This is also a testament to the fact that integrating LLMs can help enrich low-resource language datasets. The goal of this section is to investigate whether state of the art language models can assist in creating new DBpedia chapters for low-resource languages. Rather than fully automating the process, our objective is to evaluate how reliably these models can support key tasks in the DBpedia mapping workflow, particularly template property translation and ontology property alignment. We explore translation-based mapping using a multilingual machine translation model and Ontology Property Alignment using encoders. Through this investigation, we aim to understand the strengths and limitations of current language models.

6. Conclusion and Recommendations

This work presented the first Amharic DBpedia chapter and introduced tools and methods for structured knowledge extraction in a low-resource language. We extended the DBpedia Extraction Framework with Amharic-aware components, including Ethiopian calendar conversion and Ge'ez numeral normalization. In addition, we created Amharic–English template mappings and explored automation for template translation and ontology alignment using machine translation models and multilingual pretrained encoders. Our findings show that while zero-shot language models are insufficient for accurate ontology mapping, fine-tuning significantly improves alignment quality, demonstrating that selective automation can effectively support KG construction in low-resource settings. We extracted 528,370 unique triples from Amharic Wikipedia, providing the foundation for the Amharic DBpedia. All resources developed in this work, including mappings, parsers, extractors, and model configurations, are released under an open license to encourage reuse and community contribution.

While this work establishes the technical basis for the Amharic chapter, its long-term sustainability requires active community involvement. A collaborative workflow in which domain experts, language contributors, and researchers participate in mapping reviews and ontology refinement is essential. Future work should focus on improving automation tools to reduce manual effort, especially for mapping validation and incremental extraction. Since Amharic shares linguistic structure with other Ethio-Semitic languages such as Tigrinya and Tigre, the approaches introduced here can be extended to support additional DBpedia chapters in the region. Ultimately, this work represents a crucial step toward enhancing access to and representation of knowledge for low-resource languages on the Semantic Web.

7. Acknowledgment

This work has been supported by the German Federal Ministry of Research, Technology and Space (BMFTR) within the project KI-Akademie OWL under the grant no 16IS24057B, and by the Ministry of Culture and Science of North Rhine-Westphalia (MKW NRW) within the project SAIL under the grant no NW21-059D. This work is also supported by the German Research Foundation (DFG) consortium NFDI4DataScience (NFDI4DS) under Grant No. 460234259.

We thank the DBpedia community for their support and guidance, and the Google Summer of Code (GSoC) program for enabling contributions to the Amharic DBpedia project.

8. Bibliographical References

Samuael Adnew and Paul Pu Liang. 2024. Semantically corrected amharic automatic speech recognition.

H. Al-Feel. 2015. The roadmap for the arabic chapter of dbpedia. In *Proceedings*. Available at: https://api.semanticscholar.org/CorpusID:207984981.

Haytham Al-Feel. 2013. A step towards the arabic dbpedia. *International Journal of Computer Applications*, 80(3).

Jesujoba O. Alabi, David Ifeoluwa Adelani, Marius Mosbach, and Dietrich Klakow. 2022. Adapting pre-trained language models to African languages via multilingual adaptive fine-tuning. In *Proceedings of the 29th International Conference on Computational Linguistics*, pages 4336–4349, Gyeongju, Republic of Korea. International Committee on Computational Linguistics.

Hizkiel Mitiku Alemayehu, Hamada M Zahera, and Axel-Cyrille Ngonga Ngomo. 2024. Error analysis of multilingual language models in machine translation: A case study of English-Amharic translation. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 19758–19768, Miami, Florida, USA. Association for Computational Linguistics.

S. Auer, C. Bizer, G. Kobilarov, J. Lehmann, R. Cyganiak, and Z. Ives. 2007. Dbpedia: A nucleus for a web of open data. In *The Semantic Web*, volume 4825 of *Lecture Notes in Computer Science*, pages 722–735, Berlin/Heidelberg. Springer.

Satanjeev Banerjee and Alon Lavie. 2005. METEOR: An automatic metric for MT evaluation with improved correlation with human judgments. In *Proceedings of the ACL Workshop on Intrinsic and Extrinsic Evaluation Measures for Machine Translation and/or Summarization*, pages 65–72, Ann Arbor, Michigan. Association for Computational Linguistics.

Tadesse Destaw Belay, Atnafu Lambebo Tonja, Olga Kolesnikova, Seid Muhie Yimam, Abinew Ali Ayele, Silesh Bogale Haile, Grigori Sidorov, and Alexander Gelbukh. 2022. The effect of normalization for bi-directional amharic-english neural machine translation.

Elena Cabrio, Serena Villata, and Fabien Gandon. 2014. Classifying inconsistencies in dbpedia language specific chapters. In *LREC*, pages 1443–1450. Citeseer.

Dawit Chali and Peter Parapatics. 2023. Language policy and practices in an ethiopian university towards multilingualism. *Languages*, 9(6):198.

Julien Cojan, Elena Cabrio, and Fabien Gandon. 2013. Filling the gaps among dbpedia multilingual chapters for question answering. In *Proceedings of the 5th Annual ACM Web Science Conference*, pages 33–42.

Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Unsupervised cross-lingual representation learning at scale. *CoRR*, abs/1911.02116.

Peter T. Daniels and William Bright, editors. 1996. *The World's Writing Systems*. Oxford University Press, Oxford.

Encyclopaedia Britannica. 2024. Amharic language.

Joseph L Fleiss. 1971. Measuring nominal scale agreement among many raters. *Psychological Bulletin*, 76(5):378–382.

Henry F. Fliegel and Thomas C. Van Flandern. 1968. A machine algorithm for processing calendar dates. *Communications of the ACM*, 11(10):657.

Grover Hudson. 2013. *Northeast African Semitic: Lexical Comparisons and Analysis*, volume 26 of *Porta Linguarum Orientalium*. Harrassowitz Verlag, Wiesbaden.

E. K. Kim, M. Weidl, K. S. Choi, and S. Auer. 2010. Towards a korean dbpedia and an approach for complementing the korean wikipedia based on dbpedia. In *Proceedings of the 5th Open Knowledge Conference (OKCon 2010)*, CEUR Workshop Proceedings, pages 12–21.

D. Kontokostas, C. Bratsas, S. Auer, S. Hellmann, I. Antoniou, and G. Metakides. 2012. Internationalization of linked data: The case of the greek dbpedia edition. *Journal of Web Semantics*, 15:51–61.

Jens Lehmann, Robert Isele, Max Jakob, Anja Jentzsch, Dimitris Kontokostas, Pablo N Mendes, Sebastian Hellmann, Mohamed Morsey, Patrick Van Kleef, Sören Auer, et al. 2015. Dbpedia–a large-scale, multilingual knowledge base extracted from wikipedia. *Semantic web*, 6(2):167–195.

Mohamed Morsey, Jens Lehmann, Sören Auer, Claus Stadler, and Sebastian Hellmann. 2012. Dbpedia and the live extraction of structured data from wikipedia. *Program*, 46(2):157–181.

Hellina Hailu Nigatu, John Canny, and Sarah Chasins. 2024. Low-resourced languages and

online knowledge repositories: A need-finding study. pages 1–21.

Hellina Hailu Nigatu, Atnafu Lambebo Tonja, Henok Biadglign Ademtew, Hizkel Mitiku Alemayehu, Negasi Haile Abadi, Tadesse Destaw Belay, and Seid Muhie Yimam. 2025. A case against implicit standards: Homophone normalization in machine translation for languages that use the ge'ez script.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting of the Association for Computational Linguistics*, pages 311–318.

Maja Popović. 2015. chrF: character n-gram F-score for automatic MT evaluation. In *Proceedings of the Tenth Workshop on Statistical Machine Translation*, pages 392–395, Lisbon, Portugal. Association for Computational Linguistics.

Matthew G Snover, Nitin Madnani, Bonnie Dorr, and Richard Schwartz. 2009. Ter-plus: paraphrase, semantic, and alignment enhancements to translation edit rate. *Machine Translation*, 23(2):117–127.

T. Tamrat. 2008. Ethiopian calendar & millennia highlights. *International Journal of Ethiopian Studies*, 3(2):177–188. Available at: http://www.jstor.org/stable/27828897.

NLLB Team, Marta R. Costa-jussà, James Cross, Onur Çelebi, Maha Elbayad, Kenneth Heafield, Kevin Heffernan, Elahe Kalbassi, Janice Lam, Daniel Licht, Jean Maillard, Anna Sun, Skyler Wang, Guillaume Wenzek, Al Youngblood, Bapi Akula, Loic Barrault, Gabriel Mejia Gonzalez, Prangthip Hansanti, John Hoffman, Semarley Jarrett, Kaushik Ram Sadagopan, Dirk Rowe, Shannon Spruit, Chau Tran, Pierre Andrews, Necip Fazil Ayan, Shruti Bhosale, Sergey Edunov, Angela Fan, Cynthia Gao, Vedanuj Goswami, Francisco Guzmán, Philipp Koehn, Alexandre Mourachko, Christophe Ropers, Safiyyah Saleem, Holger Schwenk, and Jeff Wang. 2022. No language left behind: Scaling human-centered machine translation.

Gebremichael Tesfagerish et al. 2022. Amharic-english neural machine translation. *Journal of Artificial Intelligence Research*.

9. Language Resource References

David Ifeoluwa Adelani. 2021. *Davlan/bert-base-multilingual-cased-finetuned-amharic*. Accessed: 2025-10-24.