■ SCCH ■ Softwarepark 32a ■ A-4232 Hagenberg

AK-Graph – Data Quality Dimensions

Deliverable Report

Deliverable	D5.4 (part of)
Work Package	WP 5: Data Quality Analysis
Written by	David Pizhuk, Verena Geist (SCCH)

Table of Contents

Abstract / Executive Summary

In this document, we describe the developed data quality dimensions and their metrics as part of the evaluation of the concept for automated data quality assessment in WP5.

References on which the development of the metrics is based is listed in Section 4.

All described dimensions and metrics are implemented in the AK-Graph Dashboard[1].

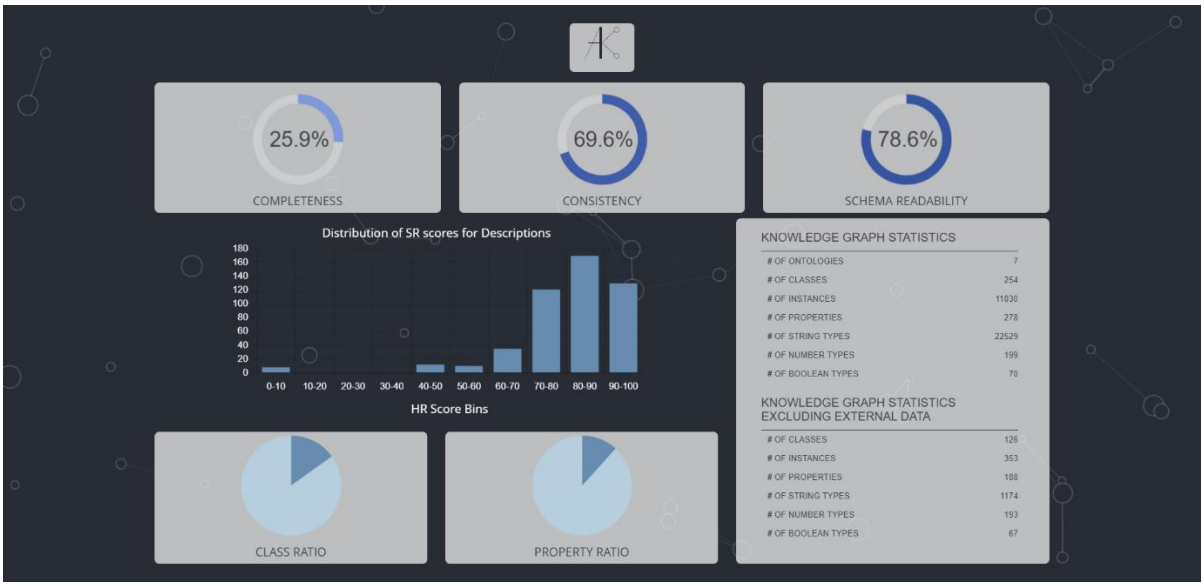

Figure 1. Landing Page of AK-Graph Data Quality Dashboard

1 Completeness

Completeness measures the extent to which the data in a knowledge graph is fully populated and covers the required aspects. It evaluates various components to ensure that the dataset is comprehensive and includes all necessary information for its intended use. Key Components of Completeness:

- **Class Completeness**: Evaluates the presence of instances for each class within the ontology.
- **Property Completeness**: Assesses the usage of properties within the dataset.
- **Relationship Completeness**: Ensures that essential relationships and properties are present in the dataset.
- **Metadata Completeness**: Ensures that key metadata elements are present and appropriately populated in the ontology, classes, and instances.
- **Interlinking Completeness**: Assesses the degree of linkage between entities within the dataset.

Completeness is crucial for ensuring that a knowledge graph provides a full and accurate representation of the domain it models. A high completeness score indicates that the dataset is well-

[1] Link to GitHub repository of dashboard's client side

populated, connected, and rich in necessary information, making it reliable and useful for various applications.

1.1 Class Completeness

Class Completeness is calculated by dividing the number of classes with instances by the total number of classes.

The **Class Ratio** pie chart shows the distribution of classes with and without instances. One section represents the proportion of classes that have instances, while the other section shows the proportion of classes without instances.

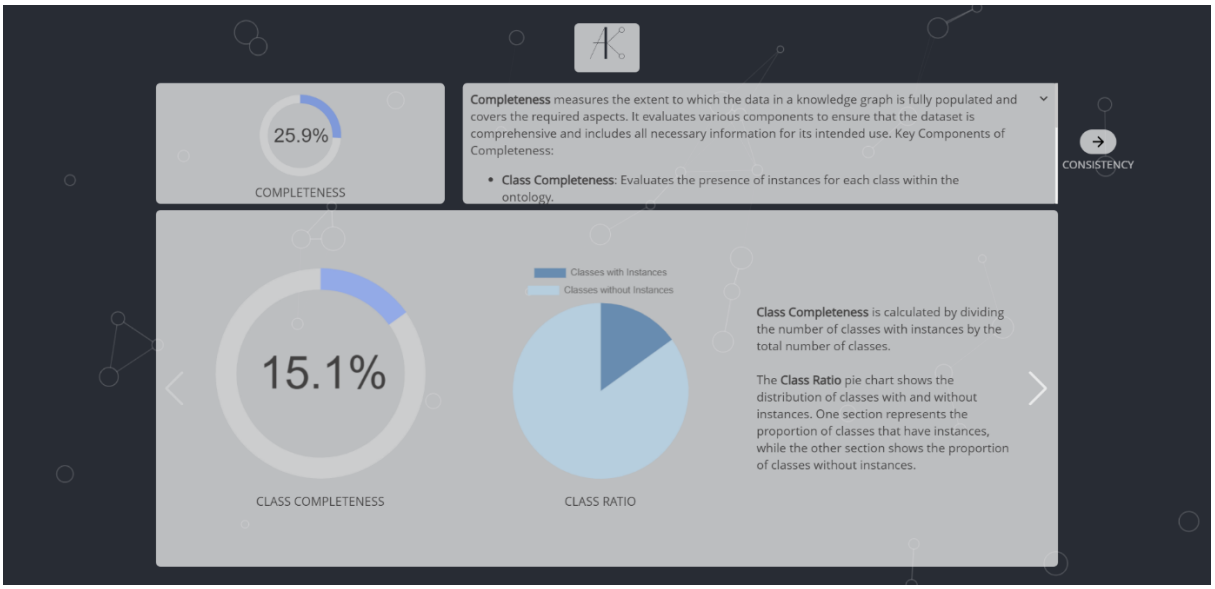

Figure 2. Class Completeness.

1.2 Property Completeness

Property Completeness is calculated by dividing the number of populated properties to the total number of properties.

The **Property Ratio** pie chart illustrates the distribution between populated and non-populated properties. One section represents the share of populated properties, while the other highlights the proportion of non-populated properties.

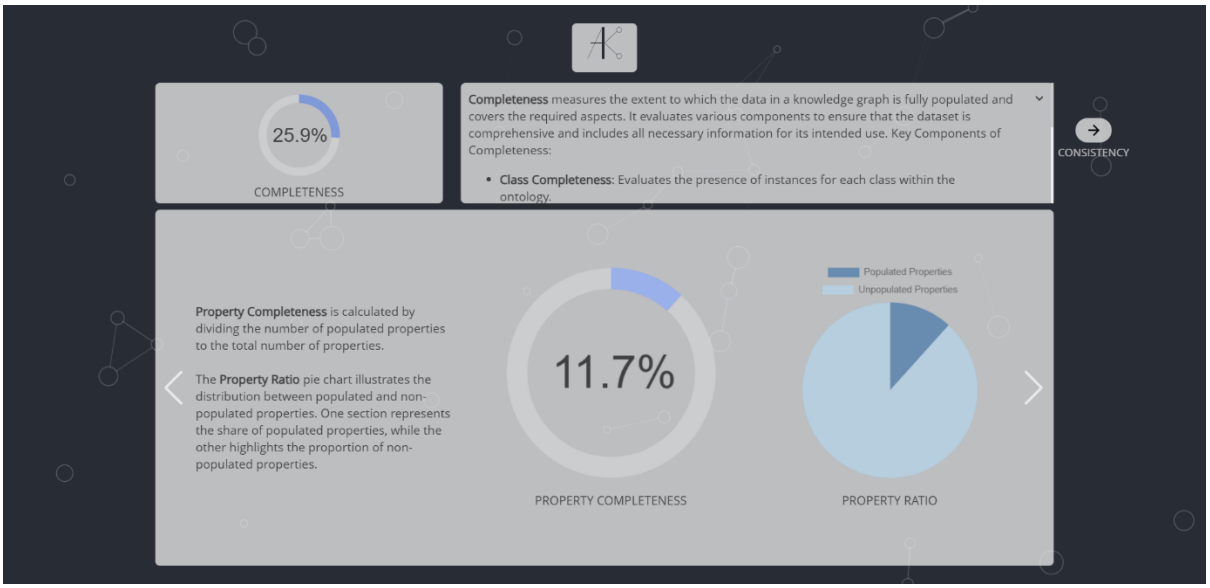

Figure 3. Property Completeness.

1.3 Relationship Completeness

Relationship Completeness measures how many instances of a subject conform to a specific relationship pattern (subject-predicate-object) within a dataset. For each statement, the process checks if an instance of the subject has a valid relationship with an object through a specific predicate.

For better understanding of this metric, the formula of its calculation is introduced:

$$\mathrm{RC} = \frac{1}{\mathrm{n}} \sum_{\mathrm{i}=1}^{\mathrm{n}} \frac{\mathrm{S_i}}{\mathrm{T_i}}$$

Where:

- RC is the overall relationship completeness.
- n is the number of statements.
- S_i is the number of subject instances that conform to the relationship in the i-th statement.
- T_i is the total number of subject instances in the i-th statement.

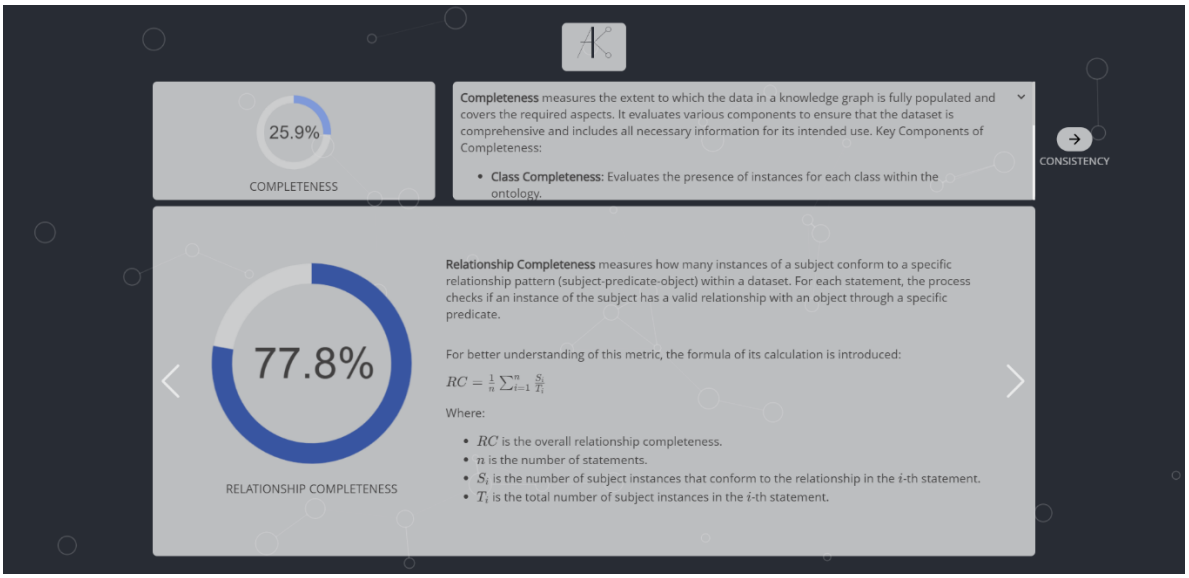

Figure 4. Relationship Completeness.

1.4 Metadata Completeness

Metadata Completeness evaluates how well key metadata properties (both essential and optional) are populated across different levels: ontology, class, and instance.

The process assigns weights(0.5, 0.3 and 0.2 respectively) to each component (ontology, class, and instance metadata) to calculate an overall score. In the next slide the more detailed information on each component can be found.

Ontology MD Completeness: This component is determined by assessing the presence of essential and optional metadata elements across all ontologies. Essential metadata elements are given more weight, and the score primarily depends on how many of these essential elements are present. If the essential metadata score falls below a certain threshold, the presence of optional metadata slightly influences the final score.

Class MD Completeness: This component is calculated based on the presence of key metadata elements for all classes, showing how well these elements are documented across all classes.

Instance MD Completeness: Similar to the class MD completeness, this component is calculated by evaluating the presence of key metadata elements for all instances.

1.4.1 Ontology Metadata Heatmap

This heatmap visualizes the presence of various metadata properties across different ontologies and is designed to help you understand which metadata are missing.

- **X-Axis**: Lists metadata properties, including both essential and optional ones.
- **Y-Axis**: Lists ontologies.
- **Cells**: Colored to show whether a metadata property is present in an ontology. Darker colors indicate presence, while lighter colors indicate absence.

- **Tooltip**: Provides details about the ontology and metadata property when hovering over a cell or y-axis label.

Figure 5. Metadata Completeness and its components.

1.5 Interlinking Completeness

To assess the interlinking of entities, we evaluate three key components:

- **Class Interlinking**: We measure how well different classes are interconnected. This involves looking at how many classes are linked to others compared to the total number of classes available.
- **Property Interlinking**: We assess the interconnections between properties. This is done by comparing the number of properties that are linked to others with the total number of properties.
- **Instance Interlinking**: We evaluate the interlinking of instances, which includes the count of linked instances versus the total number of instances.

Each of these components is analyzed to produce a score, which reflects the overall completeness of interlinking across classes, properties, and instances. The final interlinking completeness score is the average of these individual scores, giving you a comprehensive view of how well different parts of your data are connected.

Figure 6. Interlinking Completeness.

2 Consistency

The **dimension of consistency** in data quality measures how well data adheres to its expected definitions and formats. It assesses multiple components to verify that properties, data types, and formats are accurately defined and used. Key Components of Consistency:

- **Property Definition Consistency**: Ensures that the domains and ranges of properties are properly defined and that properties are not ambiguous.
- **Format Consistency**: Evaluates the uniformity of data formats, such as dates, strings, numbers, and booleans, ensuring they conform to the expected standards.
- **Scope Conformity**: Measures how well the values assigned to properties conform to their defined domains and ranges.
- **Scope Specificity**: Ensures that domains and ranges are specific enough to reduce ambiguity, avoiding the use of overly broad concepts such as rdf:Resource or owl:Thing.
- **Property Completeness**: explained in Property Completeness section.
- **Conflict Detection**: Detects potentially dangerous or conflicting triples, such as contradictory statements or property misuse.

Consistency is essential for maintaining the integrity and reliability of a knowledge graph, ensuring that it follows established rules and conventions. A high consistency score indicates that the dataset is well-structured, precise, and free from logical contradictions, making it more trustworthy for analysis and use.

2.1 Property Definition Consistency

Property Definition Consistency assesses how well the domains and ranges of properties are defined within a knowledge graph.

To calculate it, properties with only a domain, only a range, and both domain and range are identified. It is important to note that any domains and ranges defined as blank nodes are considered absent for the property. Properties with only a domain or only a range are counted as half-defined each, while properties with both domain and range definitions are counted as fully defined. The final consistency score is computed by summing the contributions of properties with both domains and ranges and those with only one, then dividing by the total number of properties to obtain a normalized score.

For better understanding of this metric, the formula of its calculation is introduced:

$$PDC = \frac{\frac{P_d + P_r}{2} + P_b}{A}$$

Where:

- PDC is the overall property definition consistency.
- P_d is the number of properties with domain only.
- P_r is the number of properties with range only.
- P_b is number of properties with both domain and range defined.
- A is number of all properties.

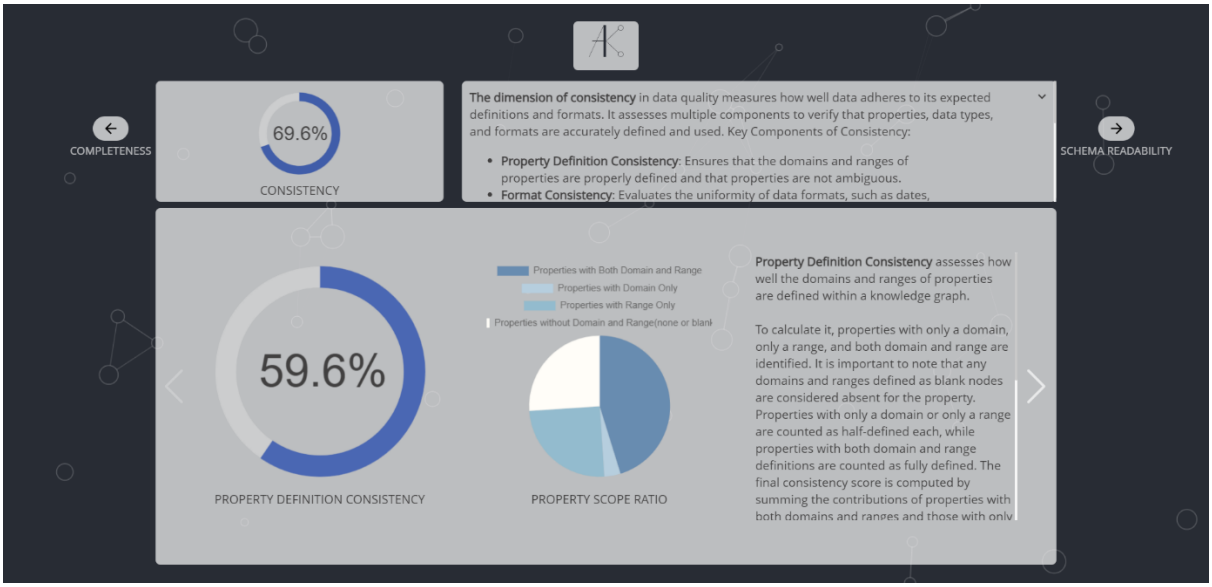

Figure 7. Property Definition Consistency.

2.2 Format Consistency

Format Consistency refers to the uniformity in the representation of various types of data within a dataset or system. It ensures that dates, strings, numbers, and boolean values adhere to a consistent format, facilitating better data integrity, accuracy, and usability.

The process of calculating format consistency involves assessing several key components: **date format consistency**, **string format consistency**, **number format consistency** and **boolean format consistency**, which are described in more detail in the next slide.

Each of these components is calculated independently and then combined to provide an overall measure of format consistency. The final score is determined by averaging the consistency scores across these categories, resulting in a unified metric that reflects the overall data format quality.

2.2.1 Date Format Consistency

Date Format Consistency ensures that all date values in a dataset are not only correctly typed but also conform to ISO 8601 format, such as YYYY-MM-DD. The dataset is examined for entries where the date properties (e.g., dcterms:created, dc:date, dcterms:issued) are represented using the xsd:date datatype and the values strictly follow the YYYY-MM-DD pattern.

- Key Calculation: It is computed as the ratio of correctly formatted and typed dates (following the YYYY-MM-DD format) to the total number of date entries in the dataset.
- Example: If some date entries are typed as strings or follow other formats like MM/DD/YYYY, they reduce the overall consistency score. Properly formatted dates like 2023-09-15 increase it.

2.2.2 String Format Consistency

String Format Consistency ensures that text data is properly formatted and conforms to expected rules regarding language encoding, valid characters, and non-emptiness.

- Key Components:
  - Language Encoding: Verifies that string entries include proper language tags or encodings (e.g., distinguishing between English and French).
  - Valid Characters: Ensures that strings contain only permitted characters, avoiding corrupt or unsupported symbols.
  - Non-emptiness: Checks that string values are not empty.

- Key Calculation: This metric of consistency is evaluated by calculating the average of the language encoding, valid character usage, and non-emptiness scores.

2.2.3 Number Format Consistency

Number Format Consistency ensures that numerical values are correctly typed and not mistakenly stored as strings.

- Key Components:
  - Correctly Typed Numbers: Numbers should be stored in their correct type (e.g., integers or floating points) rather than being stored as strings.
  - Numbers in Strings: Identifies numeric values stored as strings, which can cause issues when processing data.
- Key Calculation: The ratio of correctly typed numbers to the total number of numeric values (including those mistakenly stored as strings) determines the number format consistency.

2.2.4 Boolean Format Consistency

Boolean Format Consistency ensures that boolean values are correctly typed and not mixed with string representations of boolean values (e.g., true or false).

- Key Components:
  - Correctly Typed Booleans: Ensures that boolean values are stored using the correct data type rather than as strings.
  - Booleans in Strings: Detects boolean values that are represented as strings rather than their intended boolean type.
- Key Calculation: The ratio of correctly typed booleans to the total number of boolean values (including those stored as strings) reflects the boolean format consistency.

Figure 8. Format Consistency and its components.

2.3 Scope Conformity

Scope Conformity measures how well the actual data in a dataset adheres to the predefined domains and ranges of properties. In RDF or ontology-based systems, each property typically has an associated **domain** (the class of the subject) and **range** (the class of the object) that specifies where and how it should be used. Scope conformity evaluates the percentage of property triples that align with these definitions.

Calculation:
For each property, the system computes the ratio of property triples where the subject (for domain conformity) or object (for range conformity) conforms to the explicitly defined domain or range. This is done by comparing the number of triples that match the domain or range to the total number of triples for that property.

- The final conformity score is the mean of domain and range conformity values across all properties in the dataset.

Example:
If a property dcterms:created has a domain of Person and a range of Date, but some triples use Document as the subject or a string as the object, the scope conformity for that property would be reduced.

2.4 Scope Specificity

Scope Specificity evaluates how specific or broad the defined domains and ranges of properties are. It identifies whether the domains and ranges are too generic (overly broad) or sufficiently detailed, thus helping to improve data quality by avoiding vague classifications.

- Overly Broad URIs: Certain URIs represent overly broad categories (e.g., Thing or Resource) that are not specific enough to provide meaningful context. The system evaluates how many of the defined domains and ranges use overly broad URIs compared to more specific ones.
- Non-broad Items: The number of non-broad domain and range definitions (i.e., those that are not overly broad) is counted to determine the specificity.

Calculation:

- The system calculates the ratio of non-broad domains to the total number of defined domains and the ratio of non-broad ranges to the total number of defined ranges.
- The final specificity score is the mean of these two proportions.

Example:
If a property has a domain defined as Thing and a range as Literal, it is considered overly broad, and the scope specificity will be low. If the domain is defined as Person and the range as Date, the specificity score will increase because these are more precise definitions.

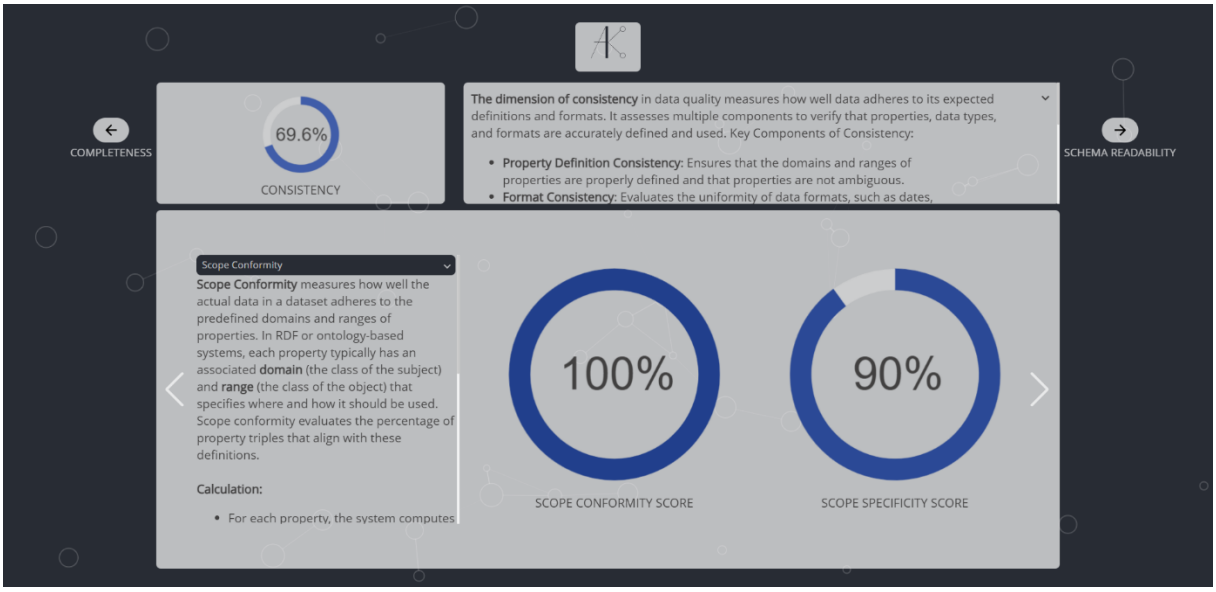

Figure 9. Scope Conformity and Scope Specificity.

2.5 Conflict Detection

In this section, user has a possibility to interact with data. User can see triples (subject, property, object), where multiple objects are assigned to the same subject-property pair, indicating a potential conflict. They are visualized in a view of tree to provide a better interaction(property -> subject -> object).

Steps to interact with a feature:

- Review the triples and identify inconsistencies.
- Select the conflicting triples you believe are incorrect.
- Check how your selection affects the consistency score.

User's input helps improve data accuracy.

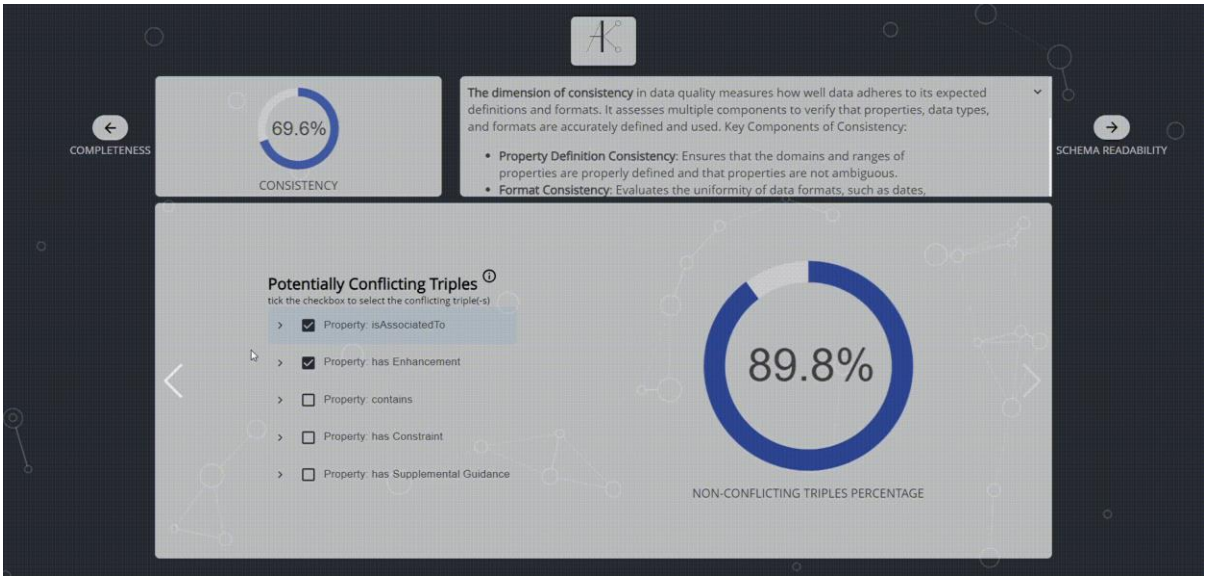

Figure 10. Interaction with Conflict Detection section.

3 Schema Readability

Schema Readability measures how easily the textual elements of a knowledge graph can be understood by human users.

It focuses on the clarity, comprehensibility, and accessibility of the schema, ensuring that users can interpret and navigate the information effectively.

Key Components of Schema Readability:

- **Word Existence**: Examines the presence of words in established lexicons, differentiating between existing and non-existing words to assess the validity of the labels and descriptions.
- **Style Consistency**: Checks for adherence to defined naming styles and conventions, ensuring that schema elements maintain a uniform structure and style, which enhances overall readability.

- **Language Confidence**: Measures the reliability of language detection in **descriptions** to ensure that the schema communicates effectively in the intended language, accounting for any penalties associated with non-English content.
- **Cognates Absence**: Assesses the lack of synonyms and hypernyms, ensuring the data is precise and aligned with expected terminologies.
- **Encoding Information Penalty**: Applies when encoding information is not present, impacting overall data quality assessment.

It is important to note the difference between the roles of labels and descriptions when assessing schema readability. In the case of **Language Confidence**, this measure is much more rational to apply only to descriptions, as they form complete sentences, allowing the Lingua language detector to perform significantly better than it would on individual words or word pairs typically found in labels(see Figure 11). On the other hand, the **Cognates Absence** score is particularly relevant for labels and titles, where avoiding ambiguities is critical, but it is not valid for descriptions. Recognizing these distinctions enhances the precision and relevance of schema readability assessment.

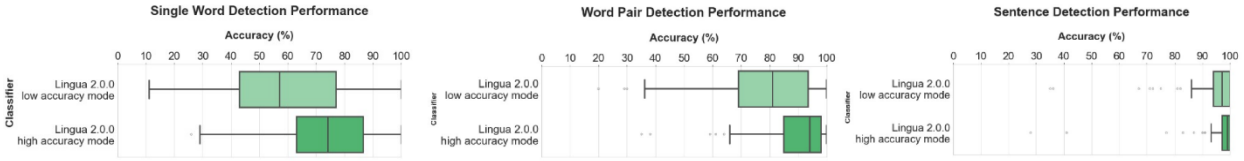

Figure 11. Lingua Language Detector Performance in different scenarios[2].

3.1 Label Readability

Label Readability calculation process involves several key steps and metrics designed to evaluate the quality and clarity of **labels** (important to mention, that **titles** in Knowledge Graph are treated as labels):

- **Word Existence**: For each label, the individual words are checked for their existence in a lexical database (WordNet). This evaluation categorizes words into three types: **existing words**(words that are found in WordNet), **non-existing words**(words that are both nonsense and have no recognized meaning) and **partial matches**(words that are not nonsense but do not have full recognition in WordNet).
  A score is then calculated for each label based on the proportion of existing words to the total number of words.
- **Style Consistency**: it is evaluated based on predefined conventions that may vary depending on whether the label represents a class, instance or property. Specific criteria include:
  - **Character Validity**: Ensures that labels contain only valid characters and do not use improper casing conventions like PascalCase or camelCase.
  - **Custom Style Evaluation**: Analyzes how the words are formatted and capitalized according to the expected style. This includes checking if the first word is in lowercase for properties or capitalized for class instances.

[2] data is taken from https://github.com/pemistahl/lingua-py

The style consistency score is derived from the adherence of the label to these conventions.

- **Cognates Absence**: The evaluation continues by assessing the presence or absence of synonyms and hypernyms for the words within each label. The calculations involve identifying whether any synonym or hypernym exists in the unique set of words derived from other labels. A score is computed based on how many words lack synonyms and hypernyms.

The final readability score for each label is calculated by taking the average of the following metrics: **word existence score**, **style consistency score**, **synonym absence score**, and **hypernym absence score**. Additionally, an **encoding penalty** is applied based on whether the label includes encoding information. Labels without such information incur a penalty, which adjusts the final readability score downwards.

Figure 12. Label Readability.

3.1.1 Histogram “Distribution of SR score for Labels”

This histogram is designed to help you better understand the calculations and the quality of text data in the Knowledge Graph. It displays the distribution of Schema Readability (SR) scores for **labels**. The SR scores are quantified in the range from 0% to 100%, which is divided into 10 intervals, each spanning a width of 10. The x-axis of the histogram is divided into 10 bins, each representing a range of SR scores. The y-axis represents the frequency of SR scores that fall within each of these bins. For each bin, the histogram shows how many scores fall into that particular range. Overall, the histogram visually represents how SR scores are distributed across the predefined score ranges.

Figure 13. Histogram “Distribution of SR score for Labels”.

3.2 Description Readability

Description Readability calculation process involves several key steps and metrics designed to evaluate the quality and clarity of **description** (important to mention, that **comments** in Knowledge Graph are treated as descriptions):

- **Word Existence**: For each description, the individual words are checked for their existence in a lexical database (WordNet). This evaluation categorizes words into three types: **existing words**(words that are found in WordNet), **non-existing words**(words that are both nonsense and have no recognized meaning) and **partial matches**(words that are not nonsense but do not have full recognition in WordNet).
  A score is then calculated for each description based on the proportion of existing words to the total number of words.
- **Style Consistency**: it is evaluated based on predefined conventions. Specific criteria include:
  - **Character Validity**: Ensures that descriptions contain only valid characters and do not use improper casing conventions like PascalCase or camelCase.
  - **Custom Style Evaluation**: Analyzes how the words are formatted and capitalized according to the expected style.

  The style consistency score is derived from the adherence of the description to these conventions.
- **Language Confidence**: We measure how confident we are in the detected language of the description. Descriptions with high language confidence receive a better score.

The final readability score for each description is calculated by taking the average of the following metrics: **word existence score**, **style consistency score**, and **language confidence**. Ad-

ditionally, an **encoding penalty** is applied based on whether the label includes encoding information. Descriptions without such information incur a penalty, which adjusts the final readability score downwards.

Figure 14. Description Readability.

3.2.1 Histogram “Distribution of SR score for Descriptions”

This histogram is designed to help you better understand the calculations and the quality of text data in the Knowledge Graph. It displays the distribution of Schema Readability (SR) scores for **descriptions**. The SR scores are quantified in the range from 0% to 100%, which is divided into 10 intervals, each spanning a width of 10. The x-axis of the histogram is divided into 10 bins, each representing a range of SR scores. The y-axis represents the frequency of SR scores that fall within each of these bins. For each bin, the histogram shows how many scores fall into that particular range. Overall, the histogram visually represents how SR scores are distributed across the predefined score ranges.

Figure 15. Histogram “Distribution of SR score for Descriptions”.

4 Bibliography

1. Lisa Ehrlinger, Gudrun Huszar, and Wolfram Wöß. “A Schema Readability Metric for Automated Data Quality Measurement”. In: June 2019.
2. Subhi Issa et al. “Knowledge Graph Completeness: A Systematic Literature Review”. In: IEEE Access 9 (2021), pp. 31322–31339.
3. Xiangyu Wang et al. “Knowledge graph quality control: A survey”. In: Fundamental Research 1.5 (2021), pp. 607–626.
4. Michael Hucka. “Nostril: A nonsense string evaluator written in Python”. In: Journal of Open Source Software 3.25 (2018), p. 596.