Text Normalization

Dimitrios Tselentis

February 11, 2025

1 Introduction

The primary objective of this task is to design and implement an automated text normalization solution to process raw composer metadata and extract clean, structured names. The goal is to eliminate redundant, incorrect, and extraneous information while ensuring that the solution is scalable and generalizable for unseen datasets.

2 Dataset Overview

The dataset comprises raw textual information related to composition writers, presenting several inconsistencies and challenges in data standardization. One notable issue is the presence of multiple names within a single entry, often separated by inconsistent delimiters such as slashes `(/)`, commas `(,)`, or vertical bars `(|)`. Additionally, the dataset contains redundant metadata, including publishing details like `"SONY/ATV MUSIC PUBLISHING"` and `"Copyright Control"` which do not contribute to the identification of individual composers. Placeholder entries, such as `"UNKNOWN WRITER (999990)"` further complicate the dataset by introducing non-informative values. Another inconsistency arises from variations in capitalization, where names appear in mixed formats (`"JERRY CHESNUT"` vs. `"Jerry Chesnut"`), requiring normalization for uniformity. Furthermore, the dataset includes names with non-Latin characters, such as `"José Hernández"`, `"Mø"` and `"Frédéric Chopin"` which necessitate proper encoding and handling to ensure accurate representation and processing of diverse linguistic elements. These challenges underscore the need for data cleaning and preprocessing to achieve a structured and standardized dataset.

3 Approach & Methodology

To systematically clean and normalize composer names, we designed a **multi-step processing pipeline** integrating **Natural Language Processing (NLP), rule-based techniques, text standardization methods and advanced validation using Large Language Models (LLMs).**.

3.1 Step 1: Data Preprocessing

The primary objective of this step is to meticulously cleanse and standardize the raw composer metadata to facilitate accurate and efficient extraction of composer names in subsequent processes.

- **Removing Noise & Placeholders:** The dataset may contain entries that do not correspond to valid composer names, such as placeholders or purely numeric identifiers, which can introduce noise into the analysis. Entries containing terms like `<Unknown>` or solely numeric values (e.g., `'999990'`) are identified and excluded from the dataset. This ensures that only entries with potential valid composer names are retained for further processing.

- **Unicode Normalization:** Composer names may include accented characters or other Unicode representations that can lead to inconsistencies in text processing. Text is converted to the Normalization Form KC (NFKC) to standardize characters with diacritical marks to their canonical forms. For instance, characters like `É` are transformed to `E`. This step ensures uniformity in text representation, facilitating accurate comparison and matching of names.

- **Standardizing Delimiters:** Composer names within entries may be separated by various delimiters such as `/`, `,` or `|`, leading to inconsistencies in parsing. Therefore, all identified delimiters are standardized to a single separator, specifically `/`. This uniformity simplifies the subsequent splitting and processing of individual composer names.

- **Case Normalization:** Inconsistent capitalization can hinder the accurate identification and comparison of composer names. All text is transformed to title case, where the first letter of each word is capitalized, and the remaining letters are in lowercase (e.g., `JERRY CHESNUT` becomes `Jerry Chesnut`). This standardization ensures consistency across the dataset, aiding in the reliable extraction and analysis of names.

3.2 Step 2: Name Extraction Utilizing Named Entity Recognition

The primary aim of this phase is to accurately identify and extract valid composer names from the metadata by leveraging advanced Natural Language Processing (NLP) techniques, specifically Named Entity Recognition (NER).

Rationale for Employing NER:

- **Complexity of Composer Names:** Composer names can vary significantly in structure, including multiple components, titles, and suffixes, which makes simple pattern-based recognition insufficient. NER models are trained to understand and identify such complexities within text.
- **Multilingual Considerations:** Given the global nature of music, composer names may appear in various languages and scripts. Pretrained NER models are equipped to handle multilingual data, enhancing the robustness of name extraction across diverse datasets.

Implementation Details:

a) Selection of NER Model: We utilize the SpaCy library's `en_core_web_sm` model, a widely adopted tool in NLP tasks known for its balance between performance and computational efficiency. This model is trained on a comprehensive corpus, enabling it to recognize a broad spectrum of named entities.

b) Processing Pipeline: The raw composer metadata is input into the NER pipeline after initial preprocessing steps, such as delimiter standardization and noise removal. The NER model processes the text to identify entities labeled as `"PERSON"` which correspond to individual names. This step involves tokenizing the text and applying the model's learned parameters to detect and classify entities.

c) Post-Processing: Post recognition, entities classified under labels other than `"PERSON"` such as `"ORG"` (organizations), are filtered out to eliminate non-relevant information like publisher names. Extracted names are normalized to ensure consistency in formatting, including converting to title case and removing extraneous whitespace.

Challenges and Mitigation Strategies:

- **False Positives/Negatives:** The NER model may occasionally misclassify entities, leading to false positives (incorrectly identified names) or false negatives (missed names). To mitigate this, a rule-based backup system is implemented to capture names that the NER model may overlook.
- **Performance on Unseen Data:** The model's performance may vary with unseen or domain-specific data. Continuous evaluation and potential fine-tuning of the NER model are planned to maintain high accuracy in name extraction.

3.3 Step 3: Rule-Based Backup Extraction

To accurately identify composer names in instances where Named Entity Recognition (NER) models may not perform optimally, particularly with unconventional name structures or formatting anomalies.

- **Pattern Matching for Capitalized Words:** NER models can sometimes fail to recognize names that deviate from common patterns or are presented in unexpected formats. Implementing rule-based pattern matching serves as a supplementary mechanism to capture such instances. By developing regular expressions to detect sequences of capitalized words that align with typical name structures. For example, the pattern `[A-Z][a-z]+ [A-Z][a-z]+` can identify names like `"John Doe."` This method ensures that names overlooked by NER due to formatting peculiarities are still extracted.
- **Handling Shortened Names & Initials:** Names presented with initials or abbreviations (e.g., `"J. Doe"`) may not be recognized by standard NER models, necessitating additional rule-based handling. Enhancing the pattern matching to account for initials and abbreviated names. For instance, the pattern `[A-Z]‡[A-Z][a-z]+` can capture names like `"J. Doe."` This ensures that such variations are identified and processed appropriately.

3.4 Step 4: Post-Processing & Standardization

To refine the extracted composer names by ensuring uniform formatting, eliminating redundancies, and establishing a consistent representation across the dataset.

- **Removing Redundant Names:** Duplicate names within entries can lead to inaccuracies in data analysis and reporting. Therefore, by implementing algorithms to identify and remove repeated names within a single entry this could be solved. For example, in the entry `"Adam Levine/Adam Levine"` the duplicate instance of `"Adam Levine"` is removed to retain a single occurrence.
- **Sorting for Consistency:** Sort composer names within each entry alphabetically. For instance, an entry like `"John Doe/Alice Smith"` is reordered to `"Alice Smith/John Doe"`. This standardization ensures uniformity and aids in systematic data analysis.

3.5 Step 5: Advanced Validation Using LLMs

To further enhance the accuracy of composer name normalization by leveraging Large Language Models (LLMs) for advanced validation and correction.

While previous steps have addressed common inconsistencies and errors in composer metadata, certain complex cases may still persist. LLMs, with their advanced language understanding capabilities, can serve as an additional layer of validation to identify and correct subtle inaccuracies that rule-based methods and traditional NLP techniques might miss.

Implementation Details:

a) Data Structuring: Prepare the extracted composer names in a structured JSON format to facilitate processing by the LLM. This structured approach ensures clarity and consistency in data representation.

b) LLM Integration: Utilize a state-of-the-art LLM to process the JSON-formatted data. The model will interpret the input and provide corrected full names for each entry.

Suggested Prompt:

```
You are an expert in music metadata normalization.
Given a list of composer names in JSON format, please provide the corrected
full names for each entry. If a name is already correct, confirm it as is.
Respond in JSON format with the original and corrected names.

Input:
{
  "composer_names": [
    {"original_name": "J. Doe"},
    {"original_name": "M. Smith"}
  ]
}

Expected Output:
{
  "composer_names": [
    {"original_name": "J. Doe", "corrected_name": "John Doe"},
    {"original_name": "M. Smith", "corrected_name": "Mary Smith"}
  ]
}
```

c) Validation with Pydantic: To ensure the integrity and correctness of the data exchanged with the LLM, employ the Pydantic library for data validation. Pydantic allows for the definition of data models and validation of JSON data against these schemas, ensuring adherence to the expected format.

Overall, integrating an LLM as a supplementary validation step, combined with JSON formatting and Pydantic validation, provides a robust framework for enhancing the accuracy and reliability of composer name normalization. This approach leverages advanced language understanding and strict data validation to ensure high-quality outcomes in music metadata processing.
However, it's important to note that implementing LLMs can incur significant costs, including computational resources, licensing fees, and potential infrastructure investments. These expenses should be carefully evaluated against the anticipated benefits to determine the overall feasibility and value of this approach

4 Conclusion

This report presents a comprehensive approach to extracting and normalizing composer names by integrating Natural Language Processing (NLP) techniques, specifically Named Entity Recognition (NER), with rule-based methods. This combined strategy effectively eliminates redundant data and standardizes composer metadata, resulting in a robust and scalable solution for the music industry.

The implementation of this methodology not only streamlines data processing but also enhances the accuracy of composer information, which is crucial for cataloging, royalty distribution, and musicological research. The adaptability of this approach ensures its applicability to diverse datasets and evolving industry standards.