RADemics

Text Preprocessing and Feature Extraction Techniques Enhancing the Accuracy of NLP Models in Industry Applications

Prajna K B, Ashwani Gupta
NITTE MEENAKSHI INSTITUTE OF TECHNOLOGY, MJP ROHILKHAND UNIVERSITY

2. Text Preprocessing and Feature Extraction Techniques Enhancing the Accuracy of NLP Models in Industry Applications

1Prajna K B, Assistant professor, Department of ECE, Nitte Meenakshi Institute of technology, Bengaluru, Karnataka, India, prajna.kb@nmit.ac.in

2Ashwani Gupta, Assistant Professor, CSIT Department, MJP Rohilkhand University, Bareilly Uttar Pradesh 243006, ashwanicta@gmail.com

Abstract

This book chapter provides an in-depth exploration of text preprocessing and feature extraction techniques pivotal for enhancing the accuracy of Natural Language Processing (NLP) models in real-world industry applications. Emphasizing key preprocessing steps, including tokenization, normalization, stopword removal, and stemming, the chapter examines their critical role in reducing noise, improving model generalization, and ensuring high-quality input data. Special attention is given to addressing challenges such as ambiguity in stemming and lemmatization, particularly in domain-specific contexts like legal and medical texts. The chapter also investigates innovative strategies in feature extraction, including dimensionality reduction and embedding techniques, to further optimize model performance. By discussing case studies from various industries, including e-commerce, healthcare, and social media, the chapter demonstrates how robust preprocessing pipelines directly contribute to the success of NLP applications. The integration of these techniques leads to the creation of more efficient, scalable, and accurate NLP models for diverse industrial domains.

Keywords:

Text Preprocessing, Feature Extraction, NLP, Tokenization, Normalization, Stemming.

Introduction

In the field of Natural Language Processing (NLP), the preprocessing phase is crucial for transforming raw textual data into a format that is more suitable for modeling [1]. Raw text is often unstructured and filled with noise, which can lead to unreliable model outputs if not handled properly [2,3]. Text preprocessing aims to clean, normalize, and standardize the data to reduce complexities and enhance the accuracy of downstream NLP models [4]. Key preprocessing steps such as tokenization, stopword removal, stemming, and lemmatization play a pivotal role in reducing dimensionality, mitigating noise, and ensuring that the model focuses on the most relevant information [5,6]. In addition to improving model accuracy, preprocessing also speeds up the learning process by ensuring that irrelevant data points do not introduce unnecessary overhead [7]. Furthermore, efficient preprocessing ensures that the model can generalize better, making it capable of performing across diverse domains, from sentiment analysis to machine translation

[8,9]. Thus, the role of preprocessing is not just foundational but also critical to building effective NLP models that perform consistently in real-world applications [10].

Tokenization is one of the earliest steps in text preprocessing and serves as the foundational building block for all subsequent operations in NLP [11]. By breaking text into smaller units, such as words or subwords, tokenization simplifies complex sentences and transforms them into manageable pieces for the model [12,13]. While tokenization is often straightforward for structured texts, it becomes more challenging in the case of multi-lingual data or noisy texts, such as those found in social media or medical records [14,15]. In multilingual NLP tasks, tokenization must account for various linguistic structures and grammatical nuances, requiring the use of sophisticated algorithms that can handle a wide array of languages and alphabets [16]. Similarly, tokenization in noisy data often involves dealing with informal language, abbreviations, and misspellings, all of which can distort the process if not handled effectively[17,18]. Thus, advanced tokenization strategies must be employed to address these complexities and ensure that models can accurately interpret the text, regardless of its linguistic origin or quality [19].

Text normalization is a key preprocessing step aimed at standardizing textual data, ensuring that variations of words are reduced to a common representation [20]. Normalization involves processes such as lowercasing, removing punctuation, expanding contractions, and handling different spelling conventions [21]. In the case of domain-specific applications, normalization becomes even more crucial, as specialized terms must be treated consistently to preserve their meaning [22]. For example, in the healthcare industry, the terms “heart attack” and “myocardial infarction” may appear in different forms within medical records but refer to the same condition. Standardizing such terms through normalization ensures that the model can correctly associate different expressions of the same concept [23]. Furthermore, normalization helps reduce the vocabulary size, making the model more efficient by focusing on core content rather than minor variations in language [24]. Despite its importance, normalization must be applied carefully, as over-normalization can lead to the loss of meaningful distinctions, especially in domains where precise terminology is critical for accurate analysis [25].