**Reflection Journal: L04 – Representation Methods**

**Marvin Azuougu**

**Department of Science, Technology, Engineering &** **Math**

**Houston Community College**

[**ITAI-2373-Natural Language Processing**](https://eagleonline.hccs.edu/courses/293189) **Summer 2025**

**Professor: Patricia McManus**

**24th of June 2025.**

**The Journey from Words to Numbers**

**Text Representation**

Today's session on text representation has been a truly illuminating "adventure" into the core of how machine learning interacts with human language. The module, "From Words to Numbers," perfectly encapsulates the fundamental challenge and the ingenious solutions explored. It's clear that bridging the gap between symbolic human language and numerical machine logic is the cornerstone of Natural Language Processing (NLP).

**Part 1-2: Foundations and Sparse Representations – The Initial Bridge**

The initial segments of the lab focused on the foundational "why" and "how" of converting text into a numerical format that computers can understand. The primary takeaway from this section is that machine learning algorithms, by their very nature, operate on mathematical principles. They excel at processing numbers, identifying patterns, and making predictions from numerical data. Human language, however, is rich with context, nuance, and an inherently non-numerical structure. This makes direct interpretation by an algorithm impossible. As the lab effectively demonstrated, it's like trying to teach a calculator to directly understand a story; it needs the information translated into a numerical input for any meaningful operation to occur.

To achieve this translation, **text preprocessing** emerged as an indispensable first step. The methodical process of lowercasing, removing punctuation, tokenization (breaking text into individual words), removing common "stop words" like "the" and "is," and stemming (reducing words to their root form) is crucial. My implementation of the preprocess\_text function solidified my understanding of how each step contributes to cleaning and standardizing the text, making it ready for numerical conversion. For instance, stemming "running," "runs," and "ran" to "run" ensures that these semantically related words are treated as the same feature by the algorithm, preventing redundant representations and improving model efficiency.

Following preprocessing, we dove into the **Bag of Words (BOW)** representation, which serves as a foundational method for turning processed text into numerical vectors. The concept is remarkably simple yet powerful: treat each document as a collection of words, disregarding their order, and count the frequency of each unique word. Building my own build\_bow\_representation function from scratch really cemented the mechanics: constructing a vocabulary of all unique words and then creating a vector for each document where each position corresponds to a word in the vocabulary, and its value is the count of that word in the document. The subsequent comparison with Scikit-learn's CountVectorizer provided a satisfying confirmation of my understanding, showing that the underlying principles are identical, even if the professional library handles various optimizations.

However, the lab also quickly highlighted the **limitations of BOW**, which are quite significant. The most prominent issue is the complete loss of **word order and grammatical structure**. The examples provided, such as "The dog ate my homework" versus "The homework ate my dog," perfectly illustrate this. Despite having entirely opposite meanings, their BOW representations would be identical because they contain the same words, just in a different sequence. This also extends to issues of negation, where "This movie is not bad" and "This movie is bad" could be seen as similar if the word "not" is removed or simply counted without understanding its semantic impact on "bad."

Another critical limitation brought to light by the BOW visualization was **sparsity**. My observation of the high sparsity percentage in the generated heatmap confirmed that most entries in the BOW matrix are zero. This signifies that for any given document, only a small fraction of the entire vocabulary is present. While this makes sense—a single movie review won't use every word in the English language—it translates to inefficient memory usage and potentially increased computational overhead, as the algorithm must process many zeros.

Despite these limitations, it's important not to dismiss BOW entirely. It still holds considerable utility in several scenarios. For **simple text classification tasks** like spam detection or basic sentiment analysis, where the presence of certain keywords is highly indicative, BOW can perform surprisingly well. It also serves as an excellent **baseline model** for more complex NLP tasks; if a more sophisticated method doesn't significantly outperform a simple BOW model, it suggests that the added complexity might not be justified. Its simplicity and lower computational demands also make it suitable for environments with **limited resources** or when quickly developing an initial prototype. In essence, while it lacks semantic depth, its straightforwardness makes it a valuable tool in specific contexts.

**Part 3: Moving Beyond Sparse – Weighted Representations**

The introduction to TF-IDF (Term Frequency-Inverse Document Frequency) promises to address some of BOW's shortcomings by introducing a weighting scheme. The core idea is brilliant: not all words are equally important. Common words like "the" or "is" offer little insight into a document's unique content, whereas rarer, domain-specific terms can be highly informative. TF-IDF aims to give higher importance to words that are frequent within a specific document (Term Frequency, TF) but rare across the entire collection of documents (Inverse Document Frequency, IDF). This helps to highlight words that are truly characteristic of a document, rather than just common fillers. This weighted approach is a logical next step to imbue our numerical representations with more semantic significance, moving beyond mere counts.

In conclusion, this module has provided a robust foundation in text representation. Starting from the fundamental necessity of converting human language into machine-readable numbers, through the practicalities of text preprocessing, and into the mechanics and limitations of Bag of Words, I feel much more equipped to understand how text data is prepared for machine learning tasks. The journey is just beginning, with TF-IDF and N-grams offering promising avenues to capture more meaning, hinting at the fascinating complexity and ongoing innovation in the field of NLP.