

Chloe Tu
ITAI 2373 - Natural Language Processing
Professor: Anna Devarakonda
Date: September 16, 2025

Lab 04 Journal: A Reflection on Text Representation

Introduction

My journey into text representation began with a simple but important idea that words are not numbers. Computers only understand numbers, so we have to find a way to turn human language into a numerical format. This lab showed me different ways to do this, starting with easy ones and moving to more complex ones. The first step was always preprocessing the text. This means cleaning it up by making all letters lowercase taking out punctuation and getting rid of common words. After doing that the real work of turning words into numbers could begin. This foundational step of preparing the data taught me the importance of a clean slate before any advanced analysis. I realized that without proper cleaning the resulting numerical data would be noisy and useless for a machine learning model. This simple task of cleaning the text laid the groundwork for everything else I would do in the lab.

Setup → BOW Implementation

The first method began with the fundamental concept of Bag of Words or BOW. I first had to prepare the text data a crucial step known as preprocessing. This involved making all the text lowercase and removing unnecessary punctuation. Once the text was clean, I implemented BOW from scratch. The method is very simple it just counts how many times each word appears in a document. The order of the words does not matter at all. The main takeaway was understanding its simplicity and also its biggest flaw it completely loses all information about word order and grammar. For a computer "the dog bit the man" and "the man bit the dog" would be the same. The notebook's examples showed me this limitation very clearly. I learned that while BOW is a good starting point and very simple to use it is not suitable for tasks where the meaning of a sentence depends on the order of its words like sentiment analysis.

Building the BOW model from scratch was an eye-opening experience. I first had to create a complete list of every unique word across all my documents. This list became my vocabulary. Then for each document I created a vector. The length of this vector was the same as the size of my vocabulary. I would go through the document word by word and mark the presence of each word by putting a 1 in its corresponding spot in the vector. If a word appeared multiple times I would increase the count. This process made me appreciate how simple and yet limited the approach is. It made the data very sparse with most of the vector being zeros. This can be a problem for larger datasets because it uses a lot of memory and can slow down a model.

TF-IDF/N-grams

To improve upon BOW the lab introduced TF-IDF which stands for Term Frequency-Inverse Document Frequency. This method is much smarter because it weighs words based on their importance. A word gets a higher score if it appears often in one document but rarely in others. This helped the model focus on unique keywords that truly define a document's topic. For example, the word "car" might be a common word but "tire pressure" might be a much more important phrase that shows what the text is about. This method helped solve the problem of words like "the" or "a" being too common and not useful. I saw that TF-IDF gave more meaningful numbers to words that truly stood out.

The lab also showed me N-grams which are groups of words like "New York" or "ice cream." This method helped fix the word order problem of BOW by treating phrases as single units. By using 2-grams or 3-grams the model can capture context that is lost with single words. For example, "not good" is very different from just "good." N-grams help to capture this negation. However, I noticed that using N-grams made the vocabulary much bigger and created very large and empty matrices. This can be a significant challenge when working with large datasets as it requires a lot of memory and can make a model very slow to train. The trade-off between capturing context and handling a large sparse matrix was a key lesson.

Word Embeddings

The most interesting part of the lab was learning about word embeddings. This method creates a dense vector for each word. What makes them so powerful is that these vectors are not just random numbers. They capture the meaning and context of the words. Words with similar meanings are placed close to each other in the vector space. For example, the lab demonstrated how the vector for "king" minus "man" plus "woman" results in a vector very close to the vector for "queen." This showed that the model had learned abstract relationships. This kind of semantic understanding is something that BOW and TF-IDF simply cannot achieve. This part of the lab was truly amazing, and it felt like I was seeing a glimpse into the future of what AI can do with language.

I explored this section in the notebook. I figure out how words like "Paris" and "France" have a similar relationship to "Berlin" and "Germany." The visual representation of these word relationships was incredibly helpful. It showed me how a computer can understand complex human concepts just by looking at the relationships between vectors. I also learned about the different types of word embeddings such as Word2Vec and GloVe and how they are trained on massive amounts of text to learn these rich representations. This method is a huge leap from sparse representations because it allows the model to generalize and find patterns that would be impossible with simple word counts.

Comparison → Reflection

Comparing all these methods showed me the clear evolution of text representation. BOW is easy and fast but lacks context. TF-IDF adds a layer of intelligence by weighing words by their importance. N-grams help capture word order, but they still suffer from being too sparse. Word embeddings are the best approach. They solve the issues of context and meaning by creating dense vectors that capture semantic relationships. They are much more powerful than the other methods. I reflected on how these tools are used in the real world for things like search engines and chatbots. The lab also made me think about the ethical issues with word embeddings. They can inherit biases from their training data which means it is crucial to use them carefully and responsibly. This is a very important consideration for anyone working in the field. This lab provided a strong foundation and made me excited to learn more about this field. The hands-on work of building these models from scratch was very helpful. It showed me how each method works step by step and highlighted the pros and cons of each approach. I now have a solid understanding of how we can turn text into numbers for machine learning, and I am ready to explore more advanced topics. The final section of the notebook also reminded me that this is a journey and that there are even more advanced methods like contextual embeddings that I can learn about in the future.