STAT 471: Introduction to R Programming Lecture

Lecture 2: Introduction to Natural Language Processing (NLP)

Cory Suzuki

Department of Mathematics & Statistics
California State University, Long Beach

3 November 2025

Overview

1. Introduction to NLP

2. The Semi-Deep End: Using Machine Learning and Deep Learning Models for NLP

What is Natural Language Processing?

Natural Language Processing (NLP) is a type of data analysis that is used to mine, transform, or measure the sentiment behind text data and strings. This branch of data science is relatively new and has many practical applications.

- NLP is the foundation for voice recognition software and apps
- Classifying text
- Measuring the emotional sentiment of text
- Tokenization of text allows large documents to be broken down into smaller characters/strings
- Special machine learning (such as Support Vector Machines/SVMs) and deep learning models such as Transformers and Generative Pre-trained Transformers (GPTs) allow for text prediction and automated data analysis from user-based queries

Elements and Tools of NLP

- Named Entity Recognition (NER): Identifies entities and nouns such as locations, names, and organizations
- Part of Speech (POS) Tagging: Dissecting text into their respective parts of speech
- Sentiment Analysis: Identifying the emotion behind text quantitatively and visually (with word clouds)
- Text Classification: Identifying words into groups by training machine learning and deep learning models

Text Tokenization and Stopwords

In NLP, we usually perform our analysis on text data using dictionaries, which are collections of words from a particular language. This allows us to differentiate unique words from *stopwords*, which are the most commonly used words that usually have little to no meaning (the, is, that, etc.)

Text tokenization is a text preprocessing task that is usually done to split sentences into separately distinct words, usually referred to as tokens. This allows computer software to process or algorithmically analyze the text data easier.

Term Frequency-Inverse Document Frequency (TF-IDF)

TF-IDF is a special statistic used to measure the importance of a word to its document or portion of text (usually we call this the corpus). TF-IDF is the product of the text frequency and inverse document frequency statistics.

- TF is the term frequency of a document or corpus
- IDF is the measure of how much information a word provides to the document or corpus

TF-IDF Statistic

The formula for this statistic is given by:

$$tfidf(t, d, D) = f_{t,d} \times ln(\frac{N}{D})$$

where t is the term, d is the document, N is the total number of documents, $f_{t,d}$ is the number of terms divided by the total words in the document, and D is the number of documents with that term.

TF-IDF Calculation Example

Suppose we wanted to compute the TF-IDF to see how important "cat" is to all documents, and we have 3 documents with the following sentences:

Document 1: "The cat sat on the mat"
Document 2: "The dog played in the park."
Document 3: "Cats and dogs are great pets."

The first step is to calculate the term frequency of the word "cat" in all 3 documents. Then we compute the inverse document frequency by taking the natural logarithm of the ratio between the total number of documents and the number of documents containing the term "cat". Lastly, we multiply the term frequencies and inverse document frequencies.

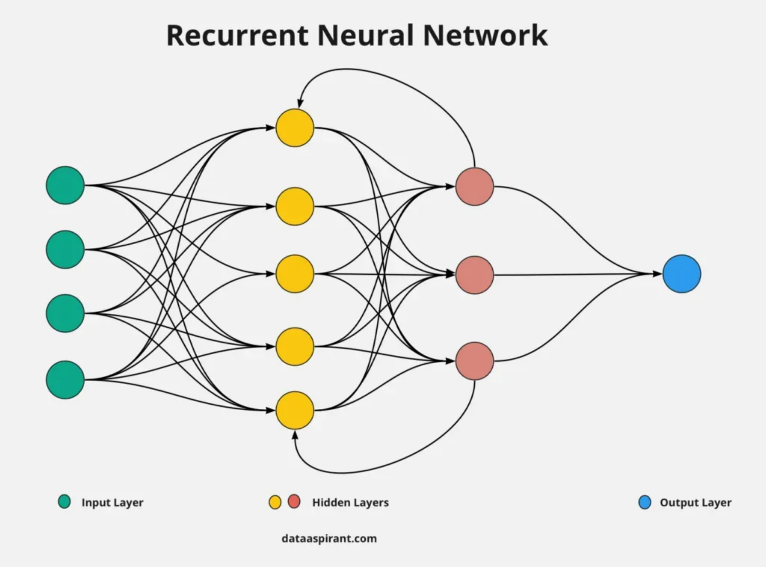
TF-IDF Calculation Example (Continued)

Support Vector Machines

Machine learning models can greatly assist in the workflows of NLP. Here, we will take a look at one model commonly used for classification tasks, also known as the Support Vector Machine (SVM). STAT 473 goes into this model in more finer detail.

- Good with small text datasets
- Performs analysis by splitting data into distinct classes with decision lines. These decision lines are generated by kernel functions and the support vectors (vectors of data points that separate the classes or groups)
- Traditionally linear, polynomial, and sigmoidal kernel functions are used, but runtimes will be long for large datasets

SVM Illustration



dataaspirant.com

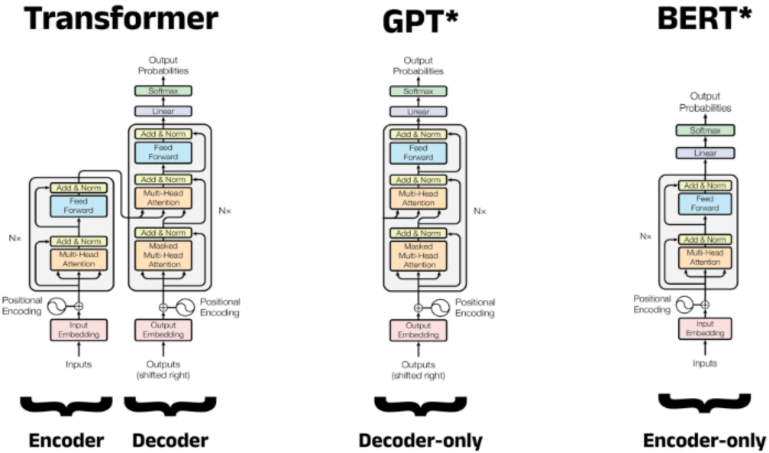
Recurrent Neural Networks (RNNs) and Bidirectional Encoder Representations from Transformers (BERT) for NLP

The RNN and BERT model architectures can be applied to NLP tasks such as text classification and sentiment analysis. These structures can handle large datasets more efficiently than SVM's, however the runtime of these algorithms on a normal CPU is a bit slow (unless you have a GPU then your runtime will be accelerated).

For simplification and efficiency, we will use the Keras package and appropriate deep learning transformer packages to handle simple NLP examples.

- RNNs are special (usually unidirectional, but can be constructed to be bidirectional) neural networks that work well with sequenced data, and in this use case, sequences of words in sentences. RNNs usually have multiple layers and nodes which "activate" in a similar way to neuronal firing in our brains.
- BERT is a special type of neural network that is bidirectional and provides high accuracy in text classification and prediction tasks. These models are usually pretrained and ready for use.

BERT, GPT, and General Transformer Architecture



*Illustrative example, exact model architecture may vary slightly

Next Time...

Bootstrap, Jacknife, and Cross Validation in Machine Learning with Applications

Announcements

Programming Assignment 3 is due this Saturday at 11:59pm!