

# COSC 426 F24 Lab 4

## Introduction

A basic component of all language models is a tokenizer. In this lab, you will develop an intuition for the design decisions behind a tokenizer.

By completing this lab, you will demonstrate that you:

- Understand the motivation for tokenization
- Can implement parts of tokenization
- Can reason through design decision in tokenization
- Can relate tokenization to real world problems

## Provided files

- Lab4.py
- evaluate.py
- through\_the\_looking\_glass.txt
- [A google doc template](#)

## What to submit

- Lab4.py with the added functions and tests for each grammar.
- A pdf of your google doc

## Part 0: What are words?

Suppose you wanted to calculate the probability of the following sentences:

- (1) “The unicorn joined Paul Atriedes in riding Shai-Hulud into the desert abyss.”
- (2) “unicorn The Atriedes joined in Paul Shai-Hulud riding abyss. desert the into”.

I think this is the first time you’ve ever seen these two sentences. Let’s assume that our training data also didn’t contain these sentences. We might first think to calculate the probability of each sentence by:

$$\frac{\text{Number of times you saw (1)}}{\text{Number of total sentences you saw}}$$

and

$$\frac{\text{Number of times you saw (2)}}{\text{Number of total sentences you saw}}$$

We would assign each a probability of 0. Ideally, we’d like to correctly record the fact that (1) is more likely than (2). One way to do this is to break this sentence down into **words** and calculate the probability based on this. In this lab, we will think about what we want to count as separate **words**.

## Part 1: Lexical Diversity

There are many ways to break down a sentence into words. One way is to just split on spaces, making words from (1) like “The” and “Shai-Hulud”. There are other more complicated ways to break sentences into words. It is helpful to have a measure that captures different approaches. In this lab, we will use as our measure **lexical diversity**, which is the ratio of word types to word tokens (word types are unique words and word tokens are the total number of words, including repeats). Put another way, lexical diversity is defined as:

$$\frac{\text{Number of word types}}{\text{Number of word tokens}}$$

Here are some lexical diversity values for different tokenizations of `through-the-looking-glass.txt`:

- If tokenization is just split on spaces: 0.21243
- If tokenization is from nltk's tokenizer: 0.09629

**\*\*Question\*\*:** Using these lexical diversity values reason about which tokenization approach has the least unique tokens.

## Part 2: Tokenization

In thinking about tokenization, consider the following sentences:

- (3) All the time you're eating your breakfast, I'll repeat 'The Walrus and the Carpenter' to you; and then you can make believe it's oysters, dear!
- (4) "Only you must eat them both, if you buy two," said the Sheep.
- (5) "Didn't I tell you?" the King repeated impatiently.

In `Lab4.py`, you should implement at *least 3 different* tokenization approaches, trying to get lower lexical diversity scores. To check your lexical diversity score, your code should output a plain text file with spaces separating the words. Then you should run `evaluate.py`. It takes one command line argument with the flag `--token_fname`. See the help message below:

```
>>> python evaluate.py --help
usage: evaluate.py [-h] [--token_fname TOKEN_FNAME]
```

Evaluate a tokenizer

options:

```
-h, --help            show this help message and exit
--token_fname TOKEN_FNAME
                        path to file with space seperated words
```

**\*\*Question\*\*:** Give brief descriptions of your three approaches and their lexical diversity on the google doc.

## Part 3: Application

In automatic speech recognition, one combines recorded speech together to make new utterances. Suppose your company uses a tokenizer to break down sentences into units that are associated with recorded speech. You'd like units bigger than one single sound (like just `f` for example), so that the generated speech sounds more natural. You have to pay someone for each recorded speech unit.

**\*\*Question\*\*:** Reason about what you'd like from the performance of your tokenizer in the context of automatic speech recognition. Consider lexical diversity and give some examples to illustrate what would make for a good tokenizer.