

Assignment 1

CMSC 473/673 — Introduction to Natural Language Processing

Item	Summary
Assigned	Wednesday September 1st
Due	Tuesday September 14th
Topic	Warmup with Counting and Basic Probabilities
Points	85

In this assignment you will step through some introductory NLP techniques.

You are to *complete* this assignment on your own: that is, the code and writeup you submit must be entirely your own. However, you may discuss the assignment at a high level with other students or on the discussion board. Note at the top of your assignment who you discussed this with or what resources you used (beyond course staff, any course materials, or public Discord discussions).

The following table gives the overall point breakdown for this assignment.

Question	1	2	3	4
Points	15	15	25	30

What To Turn In Turn in a writeup in PDF format that answer the questions; turn in all requested code, environment files (if applicable) and execution instructions necessary to replicate your results. Please do **not** turn in the raw UD data (used in Q4).

Be sure to include specific instructions on how to build (compile) your code. Answers to the following questions should be long-form. Provide any necessary analyses and discussion of your results.

How To Submit Submit the assignment on the submission site:

<https://www.csee.umbc.edu/courses/undergraduate/473/f21/submit>.

Be sure to select “Assignment 1.”

Questions

1. (15 points) Hal Daumé III has a very nice “refresher” tutorial called “Math for Machine Learning:” http://www.umiacs.umd.edu/~hal/courses/2013S_ML/math4ml.pdf. Where indicated, some of the following questions are taken from that primer. I encourage everyone to read the primer, but especially if you have difficulty answering the following questions.
 - (A) [based on Exercise 1.2] Compute the derivative of the function $f(x) = \exp(-\frac{1}{2}(x - \mu)^2)$ with respect to x (assume μ is a constant).
 - (B) [Exercise 1.5] Compute the derivative of the function $f(x) = \log(x^2 + x - 1)$.
 - (C) Compute the derivative with respect to x of the function $f(x) = \log(\sum_{k=1}^K \exp(kx^k))$, for finite, positive, integral K .
 - (D) Compute the derivative with respect to x of the function $f(x) = \log(\prod_{k=1}^K \exp(kx^k))$, for finite, positive, integral K .
 - (E) Given two K -dimensional vectors u and v , compute the dot product $u^\top v$ (write out a formula for the dot product between arbitrary real-valued vectors u and v of size K).
 - (F) Given the matrix $A = \begin{pmatrix} 4 & 2 & 0 \\ -1 & 0 & -1 \end{pmatrix}$, compute the values $A^\top A$ and $AA^\top$.
2. (15 points) Let X_i be the random variable representing the i th roll of a six-sided die and let x_i be the rolled (observed) value of the i th roll. Assume that each roll is independent from one another and the die doesn’t change between rolls.¹ Let $x_1, \dots, x_4$ be the results of rolling that die 4 different times. Each x_i can have a value of 1, 2, 3, 4, 5, or 6.
 - (A) If the die is fair (each of the six sides is equally likely), compute $p(x_1, x_2, x_3, x_4)$ (the probability of observing those four rolls in that particular order).
 - (B) The **expected value** of a random variable X_i is the average the outcomes of X_i weighted by how likely each outcome is. Mathematically, we can represent the expected value of X_i as $\mathbb{E}[X_i] = \sum_k k * p(X_i = k)$, where the sum iterates over all possible values of X_i .
 - i. Compute $\mathbb{E}[X_i]$ assuming that each outcome is equally likely.
 - ii. Compute $\mathbb{E}[X_i]$ under the distribution $p(X_i = 1) = p(X_i = 3) = p(X_i = 5) = 0.2$, $p(X_i = 2) = p(X_i = 4) = p(X_i = 6) = \frac{2}{15}$.
3. (25 points) Read the syllabus, watch the video about academic integrity (<https://drive.google.com/file/d/lnx-k5kXwlxOTj2rrhBhU69uwehamlj7/view>), and then answer the following questions. You must be logged in via UMBC to watch the video.²
 - (a) This course will require you to write written English in addition to code. So let’s say you’re writing a report and you want to use information from some previous source. Provide, with an example, of how:
 - i. you would directly quote from that source.
 - ii. you would paraphrase from that source.
 - iii. you would *improperly* use information from that source in a way that would be academically dishonest. (For this particular sub-question 3((a))iii, you will obviously not get in trouble for academic integrity violations, provided your answer is your own.)

¹We would say that each X_i is an i.i.d. sample, where i.i.d. means “independently and identically distributed.”

²While this video was prepared for graduate student orientation, it is relevant for all 473 and 673 students.

- (b) Explain, in a paragraph, how you can work together while avoiding issues of academic integrity.
 - (c) When it comes to group work, is each individual responsible for the entire group's work?
 - (d) Provide three potential instances of academic integrity relating to coding.
 - (e) Discuss how you can utilize online resources and forums, such as Stack Overflow, effectively and with integrity in this class.
 - (f) Discuss how online resources like Stack Exchange or Chegg can be misused and result in academic dishonesty.
4. **(30 points)** In this question, you'll be doing some basic counting of words—the “Hello, World” of NLP.

In the GL directory

`/afs/umbc.edu/users/f/e/ferraro/pub/473-f21/data/ud-treebanks-v2.2`
you will find the entire Universal Dependencies dataset. If you'd prefer to download it yourself, you can from <http://universaldependencies.org/>. Just make sure you get version 2.2.

For this question, look at the **UD English-EWT** folder. In that folder, you'll see a license file, a README file, and three pairs of files. The files we'll want to be concerned with are the `.conllu` files. One corresponds to the “training” data, another corresponds to the “development” data, and the last corresponds to the “test” data. For this assignment, do not run your code on the test data. **If you use the test data or if you use the `.txt` files, you will not receive full credit! If you write code that tries to interpret lines in the `.conllu` file that begin with a `#`, you will not receive full credit!**

The original text of each word is in the second column of each row. For example, the first sentence has 29 tokenized words. We can see this by getting and space-joining all of the words for the first sentence of `en_ewt-ud-train.conllu`:

```
Al - Zaman : American forces killed Shaikh Abdullah al - Ani , the
preacher at the mosque in the town of Qaim , near the Syrian border .
```

The other columns will be useful later in the semester: they contain information like the lemma for each word (third column), linguistic/morphological information, including part-of-speech tags (columns 4–6), and syntactic information (columns 7–9).

- (a) Read through the document discussing the format of the `.conllu` files:

<https://universaldependencies.org/docs/format.html>.

With that understanding, compute the number of sentences in both the training and development splits? On average, how many words are there per sentence?

You can use any method you want to compute the number and averages (e.g., Linux commands, or a small Python or Java program). Turn in what you wrote, or provide the command in your writeup with a brief explanation of the steps involved.

While it is a good idea to question your data, especially if it looks strange/not what you expected, for this question you can take these tokens as they are: assume that, for *some* application, they are useful. The text you see is called *tokenized text*. In particular, it is text that has been tokenized, or split into individual “words,” according to a particular specification. You may be surprised that we consider punctuation as different tokens.

But let's dive into this some more. The individual instances you observe are **tokens**, where each token is drawn from a set of **types**. Using a programming analogy, we can say that word types are like classes while word tokens are like instances of that class. For example, in the following sentence there are six types and eight tokens:

the gray cat chased the tabby cat .

Notice that this computation includes punctuation.

- (b) Identify the fields (columns) considered to be morphological annotations. Choose one of these fields and, in 2-3 sentences, speculate on the benefit that such an annotation may provide.
- (c) Using the "FORM" field from the training file, how many different word types and tokens are there? Do not perform any processing that modifies the words. Turn in the code for this, or if done on the command line, describe how.
- (d) Examine some of the most common words, like the twenty, thirty, and fifty most common ones. Using examples, discuss what you notice about those common words. Each word should be alphanumerically (lexically) distinct.

- (e) In the above question, should all of these items actually be considered distinct? What are some ways that we could group together words? Hypothesize some effects your collapsing would have. You can argue for or against your collapsing method.

Hint: There is not a right or wrong answer here. You may want to examine the "LEMMA" field when coming up with your answer. Some collapsing methods may be more appropriate than others, but the question is to think about these methods and what effect they may have.³

- (f) Using the FORM field, examine some of the least common words. Do words that appear only once look like "standard" words? What about words that appear ten times? Fifty times? What about 100 times?

Regardless of whether these words were *standard*, are they "reasonable?" That is, are they items that you would want to be able to talk about as distinct items? From a computational point of view, do you want to spend the computational resources to deal with them?

Argue for or against these items' reasonableness. If you find them unreasonable, propose a solution. You do not need to implement it.

- (g) The FEATS fields is a "list" of attribute-value pairs $k = v$, separated by a vertical pipe ("|"). How many unique attributes k are there in the training set?
- (h) Now it's time to look at the development split: this is in the file `en_ewt-ud-dev.conllu`. How many word tokens are there? How many word types were not seen in the training data? We call these *out of vocabulary* (OOV) words.

Again, do not perform any processing that modifies the words. Turn in the code for this, or if done on the command line, describe how.

³Now, there are *simpler* answers. In particular, some collapsing methods can be accomplished with simple calls to standard string processing functions. Others could be accomplished with some more advanced processing (e.g., see column 3).