University of Central Missouri

Department of Computer Science & Cybersecurity

CS5760 Natural Language Processing

Fall 2025

Homework 2.

Student name:

Submission Requirements:

- Once finished your assignment push your source code to your repo (GitHub) and explain the work through the ReadMe file properly. Make sure you add your student info in the ReadMe file.
- Submit your GitHub link on the Bright Space.
- Comment your code appropriately ***IMPORTANT.***
- Any submission after provided deadline is considered as a late submission.

Q1. Bayes Rule Applied to Text (based on slide: Bayes' Rule for documents)
The PPT shows that classification is based on:

$$c_{MAP} = \arg\max_{c \in C} P(c)\, P(d \mid c)$$

Tasks:

1. Explain in your own words what each term means: P(c), P(d|c) and P(c|d).
2. Why can the denominator P(d) be ignored when comparing classes?

Q2. Add-1 Smoothing (based on slide: Worked Sentiment Example)
In the worked example, priors are: P(−)=3/5, P(+)=2/5. Vocabulary size = 20.

Tasks:

1. For the negative class, the total token count is 14. Compute the denominator for likelihood estimation using add-1 smoothing.
2. Compute P(predictable|−) if the word "predictable" occurs 2 times in the negative documents.
3. Compute P(fun|−) if "fun" never appeared in any negative documents.

Q3. Worked Example Document Classification (based on slide: Test document "predictable no fun")
Using the smoothed likelihoods and priors from Q2, compute the probability scores for the document *"predictable no fun"* under both the positive and negative classes.

Tasks:

1. Show each step of the multiplication.
2. Which class should the system assign to this document?

Q4. Harms of Classification (based on slide: Avoiding Harms in Classification)

Tasks:

1. Define **representational harm** and explain how the Kiritchenko & Mohammad (2018) study demonstrates this type of harm.
2. What is one risk of censorship in toxicity classification systems (based on Dixon et al. 2018, Oliva et al. 2021)?
3. Give one reason why classifiers may perform worse on African American English or Indian English, even though they are varieties of English.

Q5: Evaluation Metrics from a Multi-Class Confusion Matrix

The system classified 90 animals into Cat, Dog, or Rabbit. The results are shown below:

System \ Gold	**Cat**	**Dog**	**Rabbit**
Cat	5	10	5
Dog	15	20	10
Rabbit	0	15	10

Tasks:

1. Per-Class Metrics
   - Compute precision and recall for each class (Cat, Dog, Rabbit).
2. Macro vs. Micro Averaging
   - Compute the macro-averaged precision and recall.
   - Compute the micro-averaged precision and recall.
   - Briefly explain the difference in interpretation between macro and micro averaging.
3. Programming Implementation
   Write Python code that:

   1. Accepts the confusion matrix above as input.
   2. Computes per-class precision and recall.
   3. Computes macro-averaged and micro-averaged precision and recall.
   4. Prints all results clearly.

Q6. Bigram Probabilities and the Zero-Probability Problem

You are given the following bigram counts from a small training corpus:

Previous word	Next words (with counts)
<s>	I: 2 , deep: 1
I	love: 2
love	NLP: 1 , deep: 1
deep	learning: 2
learning	</s>: 1 , is: 1
NLP	</s>: 1
is	fun: 1
fun	</s>: 1
ate	lunch: 6 , dinner: 3 , a: 2 , the: 1

Tasks:

1. Bigram Sentence Probabilities
   Using maximum likelihood estimation (MLE):

$$P(w_i \mid w_{i-1}) = \frac{C(w_{i-1}, w_i)}{C(w_{i-1})}$$

   - Compute the probability of sentence S1: <s> I love NLP </s>.
   - Compute the probability of sentence S2: <s> I love deep learning </s>.
   - Which sentence is more probable under the bigram model?

2. Zero-Probability Problem
   Using the same table, compute:
   - P(noodle|ate) with MLE.
   - Explain why this probability creates problems when computing sentence probabilities or perplexity.

- Apply Laplace smoothing (Add-1) to recompute P(noodle|ate). Assume the vocabulary size is 10 and total count after “ate” is 12.

Q7. Backoff Model (based on “Activity: <s> I like cats … You like dogs” slide)

Training corpus:

<s> I like cats </s>
<s> I like dogs </s>
<s> You like cats </s>

Counts:

- I like = 2
- You like = 1
- like cats = 2
- like dogs = 1
- cats </s> = 2
- dogs </s> = 1

Tasks:

1. Compute P(cats|I,like).
2. Compute P(dogs|You,like) using trigram → bigram backoff.
3. Explain why backoff is necessary in this example.

Q8. Programming: Bigram Language Model Implementation (based on “Activity: I love NLP corpus” slide)

Tasks:

Write a Python program to:

1. Read the training corpus:

2. <s> I love NLP </s>
3. <s> I love deep learning </s>
4. <s> deep learning is fun </s>

5. Compute unigram and bigram counts.
6. Estimate bigram probabilities using MLE.
7. Implement a function that calculates the probability of any given sentence.
8. Test your function on both sentences:
    - <s> I love NLP </s>

- <s> I love deep learning </s>

9. Print which sentence the model prefers and why.

Solutions H.W-2 NIDHIN NINAN (700772413)

Q-1

1.1 $P(c|d)$: Posterior Probability.

→ This signifies the probability that a document (d) belongs to certain class, given the specific words in the document.

$P(c)$: Prior Probability

→ This signifies the overall probability of class c, that is independant of any document

$P(d|c)$: Likelihood

→ Probability of observing a specific documents, given that the already know that the document belongs to class c.

1.2

$$C_{MAP} = \arg\max_{c \in C} P(c|d) = \arg\max_{c \in C} \frac{P(c)\,P(d|c)}{P(d)} \quad \text{- eq (i)}$$

$$= \arg\max_{c \in C} P(d|c)\,P(c)$$

⇒ We are able to drop the denominator because.

C_{map} classifier goal is to find the most likely class by maximizing $P(d|c)$ across all possible classes.

This means that when comparing class:;

$P(+ve|d)$ vs $P(-ve|d)$,

maximising $P(c|d)$ is equivalent to maximising the numerator in eq (i). Since the probability of (d) $P(d)$ is the same across all class., it can be dropped. for $P(+ve|d)$ and $P(-ve)$ ⇒

$$\max \left[\frac{P(d|+ve)\, d(+ve)}{P(d)} \right] \quad \text{OR} \quad \max \left[\frac{P(d|-ve)\, P(-ve)}{P(d)} \right]$$

$$\equiv \max \left[P(d|+ve)\, P(+ve) \right] \text{ OR } \max \left[P(d|-ve)\, P(-ve) \right]$$

(*) We are dropping the constant factor $P(d)$.

Q-2.

$P(-ve) = 3/5 \quad P(+ve) = 4/5 \quad$ Vocabulary size = 20

Laplace (Add 1) Smoothing Probability.

$$P(w_i|c) = \frac{count(w_i|c) + 1}{\sum_{w \in V} count(w|c) + |V|}$$

2.1 Denominator of likelihood estimation is;

$\sum_{w \in V} count(w_i|c) + |V|$ [total count of all tokens in a class + vocabulary size]

$\Rightarrow$ token total in -ve class = 14 (given).

$= 14 + 20 = \boxed{34}$

2.2 $P(\text{predictable}|-ve) = P(w_i|c)$; $count(w_i|c) = 2$

$$P(w_{i=\text{'predictable'}} | c = -ve) = \frac{count(w_i|c) + 1}{\sum_{w \in V} count(w|c) + |V|} = \frac{2+1}{14+20}$$

$$\boxed{P(w_i|-ve) = \frac{3}{34}}$$

2.3 $count(\text{fun} = w_i | -ve) = 0$

$$P(w_{i=\text{'fun'}} | c = -ve) = \frac{0+1}{14+20} \Rightarrow \boxed{P(w_i|-ve) = \frac{1}{34}}$$

Q-3

$$\text{Score}(c) = P(c) \times \prod_{w \in d} P(w|c)$$

$P(+ve) = 3/5$, $P(-ve) = 4/5$, Vocabulary = 20

$$P(w_i|c) = \frac{\text{count}(w_i|c) + 1}{\sum_{w \in V} \text{count}(w|c) + |V|}$$

$\Rightarrow$ Document (d) = "predictable no fun"

3.1

$$P(\text{predictable}|-ve) = \frac{2+1}{14+20} = \frac{3}{34}$$

(different from example as prior from Q-2).

$$P(\text{no}|-ve) = \frac{1+1}{14+20} = \frac{2}{34}$$

$$P(\text{fun}|-ve) = \frac{0+1}{14+20} = \frac{1}{34}$$

$$P(\text{predictable}|+ve) = \frac{0+1}{9+20} = \frac{1}{29}$$

[count(predictable|+ve) = 0, count(w|+ve) = 9] from PPT example

$$P(\text{no}|+ve) = \frac{0+1}{9+20} = \frac{1}{29}$$

[count('no'|+ve) = 0, count(w|+ve) = 9] from PPT example

$$P(\text{fun}|+ve) = \frac{1+1}{9+20} = \frac{2}{29}$$

[count('fun'|+ve) = 1, count(w|+ve) = 9] from PPT example

$$\text{Score}(-ve) = \left(\frac{3}{5}\right)\cdot\left(\frac{3}{34}\right)\cdot\left(\frac{2}{34}\right)\cdot\left(\frac{1}{34}\right) = \frac{18}{5 \times 34^3} = 9.16 \times 10^{-5}$$

$$\text{Score}(+ve) = \left(\frac{2}{5}\right)\cdot\left(\frac{1}{29}\right)\cdot\left(\frac{1}{29}\right)\cdot\left(\frac{2}{29}\right) = \frac{4}{5 \times 29^3} = 3.29 \times 10^{-5}$$

3.2. Score(-ve) 9.16×10^{-5} > Score(+ve) 3.29×10^{-5}.

∴ The document would be assigned to -ve class.

Q-4.

4.1 Representational Harm are caused by an NLP system that demeans a social group. This type of harm typical occurs by perpetuating negative stereotypes about that groups.

The Kiritchenko & Mohammad 2018 study, came to the conclusion NLP system assigned lower sentiment and more negative emotion to sentences w/ African American name due to propagation of negative stereotypes found in the training data. The study involved analyzing 200 sentiment analysis systems using pairs of sentences that were identical except for names (comparison between sentences featuring African American names against European names).

4.2. A significant risk of using toxicity classification systems for content moderation was the censorship of legitimate speech particularly from minority group. The system incorrectly flagged non-toxic sentences that merely mentioned minority identities (like the words 'blind' or 'gay').

4.3 One reason why classifier perform worse in English variants, like American English or Indian English, was due to issues in the training data. NLP systems amplify biases present in training data, due to lack of sufficient representation of varieties.

Q.5.

ACTUAL → / PREDICTION ↓	CAT	DOG	RABBIT	TOTAL
CAT	5	10	5	20
DOG	15	20	10	45
RABBIT	0	15	10	25
TOTAL	20	45	25	90

5.1 $\text{Precision} = \frac{TP}{TP+FP}$ $\quad \text{Recall} = \frac{TP}{TP+FN}$

$\text{Precision(cat)} = \frac{5}{5+15}$ $\quad \text{Precision(Dog)} = \frac{20}{20+25}$ $\quad \text{Precision (Rabbit)} = \frac{10}{10+15}$

$\text{Precision(cat)} = \frac{5}{20} = \frac{1}{4}$ $\quad \text{Precision(Dog)} = \frac{20}{45} = \frac{4}{9}$ $\quad \text{Precision (Rabbit)} = \frac{2}{5}$

$\text{Recall(cat)} = \frac{5}{5+15}$ $\quad \text{Recall(Dog)} = \frac{20}{20+25}$ $\quad \text{Recall(Rabbit)} = \frac{10}{10+15}$

$\text{Recall(cat)} = \frac{1}{4}$ $\quad \text{Recall(Dog)} = \frac{4}{9}$ $\quad \text{Recall(Rabbit)} = \frac{2}{5}$

5.2 Macro-averaged Precision & Recall calculated the avg of per class precision & recall score and it gave equal weight to each class.

$$\text{Precision}_{(macro)} = \frac{\text{Precision(Cat)} + \text{Precision(Dog)} + \text{Precision (Rabbit)}}{3}$$

$$\text{Precision}_{(macro)} = \frac{1}{4} + \frac{4}{9} + \frac{2}{5} = 0.365$$

$$\text{Recall}_{(macro)} = \frac{\text{Recall(Cat)} + \text{Recall(Dog)} + \text{Recall(Rabbit)}}{3}$$

$$\text{Recall}_{(macro)} = \frac{1}{4} + \frac{4}{9} + \frac{2}{5} = 0.365$$

Micro-averaged involved pooling the counts the TP, FP and FN for all the classes & then computed the precision & recall. It gave equal weight to each instance.

	(CAT)		(DOG)		(RABBIT)		(TOTAL)
Total TP =	5	+	20	+	10	=	35
Total FP =	15	+	25	+	15	=	55
Total FN =	15	+	25	+	15	=	55

$$\text{Precision}_{(\text{micro})} = \frac{\text{Total TP}}{\text{Total TP} + \text{Total FP}} = \frac{35}{35+55} = \frac{35}{90} = \frac{7}{18}$$

$$\text{Recall}_{(\text{micro})} = \frac{\text{Total TP}}{\text{Total TP} + \text{Total FN}} = \frac{35}{35+55} = \frac{35}{90} = \frac{7}{18}$$

$$\text{Precision}_{(\text{micro})} = \frac{7}{18}, \quad \text{Recall}_{(\text{micro})} = \frac{7}{18}$$

Macro-averaging gives equal weight to each class. It is a better measure of performance when the model needs to perform well across all classes, especially small or rare ones.
Micro-averaging pools results together and is dominated by the scores of the largest class. In single-label multi-class scenario, where every item is classified, the micro-average score is equivalent to accuracy.

Q-6.

Total count for required words based on corpus:
$C(\langle s \rangle) = 3$, $C(I) = 2$, $C(\text{love}) = 2$, $C(\text{deep}) = 2$, $C(\text{learning}) = 2$, $C(\text{NLP}) = 1$

$$\text{Max Likelihood} = P(w_i \mid w_{i-1}) = \frac{C(w_{i-1}, w_i)}{C(w_{i-1})}$$

6.1 Probability of "$\langle s \rangle$ I love NLP $\langle s \rangle$"

$$= P(s_1) = P(I \mid \langle s \rangle) \cdot P(\text{love} \mid I) \cdot P(\text{NLP} \mid \text{love}) \cdot P(\langle s \rangle \mid \text{NLP})$$

$$= \frac{2}{3} \cdot \frac{2}{2} \cdot \frac{1}{2} \cdot \frac{1}{1} = \frac{4}{12}.$$

$$P(s_1) = \frac{1}{3}.$$

Probability of "<s> I love deep learning <s>"

$= P(S_2) = P(I \mid <s>) \cdot P(love \mid I) \cdot P(deep \mid love) \cdot P(learning \mid deep) \cdot P(<s> \mid learning)$

$= \frac{2}{3} \cdot \frac{2}{2} \cdot \frac{1}{2} \cdot \frac{2}{2} \cdot \frac{1}{2} = \frac{8}{48} = \frac{1}{6}.$

$P(S_2) = \frac{1}{6}.$

$P(S_1) > P(S_2) \equiv$ acc. to the model;

$S_1 \equiv$ "I love NLP" is more probable.

6-2. $P(noodle \mid ate)_{MLE} = \frac{C(ate, noodle)}{C(ate)} = \frac{0}{6+3+2+1} = 0$

The problem with a zero probability for an N_gram is that it means the entire sentence containing the N_gram will be assigned a probability of zero.
This also affects perplexity (the inverse probability of the test set, normalised by the number of words). This would mean the perplexity would not be computable for 0 zero.

$P(noodle \mid ate)_{Laplace\ Smoothing} = \frac{C(w_{n-1}, w_n) + 1}{C(w_{n-1}) + V}$

$= \frac{C(ate, noodle) + 1}{C(ate) + V} = \frac{0+1}{12+10}$

$P(noodle \mid ate)_{Laplace\ Smoothing} = \frac{1}{22}.$

Q-7.

Max Likelihood $= P(w_i | w_{i-1}) = \frac{C(w_{i-1}, w_i)}{C(w_{i-1})}$, $C(like) = 3$

7.1 $P(cats | I, like) = \frac{C(I, like, cats)}{C(I, like)} = \boxed{\frac{1}{2}}$

7.2 $P(dogs | You, like) = \frac{C(You, like, dogs)}{C(You, like)} = \frac{0}{1}$ → Trigram Prob.

So backoff to bigram,

$P(dog | like) = \frac{C(like, dog)}{C(like)} = \frac{1}{3}$ → Bigram Prob.

Since is the probability is 0, the

$\boxed{\text{backoff estimate} = \frac{1}{3}}$.

7.3 Backoff is necessary in this example to handle zero-frequency problem caused by data sparsity, resulting in zero probability under the Maximum Likelihood Estimate. Backoff solves this by falling back to the lower order N-gram (from trigram → bigram), which has a non-zero count. This will ensure that the model can handle sparcity and generalize to sequences that do not appear in the training data.