CS 613: NLP

Assignment 2: Language Modeling

Total marks: **100 Pts.** **(+10 BONUS)**	**Submission deadline:** ~~**23:59:59 Hrs, September 18, 2023 (Monday)**~~ **23:59:59 Hrs, September 20, 2023 (Wednesday)**

Assignment Instructions

A walkthrough of the assignment will be presented in the upcoming Monday's class on 11 September 2023:

1. Regarding the late submission, we will be following the penalty as per the table:

Late Submission	**Penalty (Out of 100)**
Till 1-hour past the deadline	5 points
1 to 12 hours past the deadline	10 points
12 to 24 hours past the deadline	20 points
24 to 36 hours past the deadline	40 points
36+ hours past the deadline	100 points

2. We will follow the zero plagiarism policy, and any act of plagiarism will result in a zero score for the assignment.
3. Please cite and mention others' work and give credit wherever possible.
4. If you seek help and discuss it with the stakeholders or individuals, please ask their permission to mention it in the report/submission.

Problem Statement

Training the n-gram language models on the subreddits data.

Tasks (100 Pts. +10 Pts. [BONUS])

1. Take the data from **here**.
2. Use the NLTK sentence tokenizer.
3. Split the corpus randomly into 80% and 20% ratio (sentences, aka documents).
4. Train the LM on the 80% split and validate the remaining 20% split.

5. Train the following LMs and report the respective perplexity scores. Perplexity will be computed as an average over all the sentences.
    a. Unigram **[10 Pts.]**
    b. Bigram **[10 Pts.]**
    c. Trigram **[10 Pts.]**
    d. Quadgram **[10 Pts.]**

    Note that No existing NLP library is allowed to implement the above language models. All these language models have to be implemented by hand, and they should strictly follow the definitions given in the class. In case you have reused any existing code or ChatGPT-like conversational agents for code generation, it might **lead to ZERO marks**.
6. Use the Laplace smoothing on the above LMs and compare the change in the perplexity with and without smoothing.

    Note that No existing NLP library is allowed to implement smoothing techniques. All these language models have to be implemented by hand, and they should strictly follow the definitions given in the class. In case you have reused any existing code or ChatGPT-like conversational agents for code generation, it might lead to ZERO marks. **[10 Pts. on each LM; Total 40 Pts.].**
7. Write a justification with your observations: When smoothing was not considered and where the smoothing was considered. **[20 Pts.]**
8. Choose any **2** other smoothing of your choice (*Additive, Good Turing, or Kneser-Ney*; and Train the same n-gram LMs) and write your understanding of using different smoothing techniques.

    Note that No existing NLP library is allowed to implement smoothing techniques. All these language models have to be implemented by hand, and they should strictly follow the definitions given in the class. In case you have reused any existing code or ChatGPT-like conversational agents for code generation, it might lead to ZERO marks. **10 Pts. [BONUS].**

Points Split: 10+10+10+10+40+20 = 100 + 10 (Bonus)

Submission

1. Submit your code (GitHub) or colab notebook with proper comments to this link.
    a. Make sure the individual contribution is appropriately added.

Expectations from the team:

1. Properly divide the team into sub-groups and distribute your tasks equally.
2. Write the contributions or tasks completed by each team member.

References

If required, please feel free to take help from the following references:

1. [Medium] N-gram language models
2. [GFG] N-Gram Language Modelling with NLTK

FAQs

1. *Feel free to do the preprocessing and showing analysis with/without preprocessing.*
2. *The split will be random.*
3. *If the NLTK sentence tokenizer is not returning the tokenized sentences, use the sentences as they are.*
4. How much pre-processing do we need to do? Is it sufficient to do only spell correction and lowercase? Or lemmatization is necessary for this
    a. *It is up to the team. Metrics can be checked with and without processing techniques.*
5. Any hints on how to choose vocabulary?
    a. *You need to check the vocab from the given dataset.*
6. without smoothing, we can get zero probability, which leads to infinity perplexity. So, in that, how will we find the average of perplexity?
    a. *In that case, infinity + infinity will be infinity, so that will be expected.*
7. ***Since the dataset is multilingual, feel free to use different tokenizers. Make sure to add an analysis of the tokenizers and data.***
8. ***In the case of n-grams without smoothing, do we neglect the words that do not appear in train data, or else their probability would be zero, leading to infinite perplexity? So, is it fine to neglect those words and calculate the perplexity?***
    a. ***Yes, it is fine.***