Building N-Gram Language Models from Scratch

Iron Code Benders

Csci 5541 March 2023

Introduction

Author classification in natural language processing involves identifying the author of a given text based on their writing style or other textual features [1]. It has applications in forensic linguistics, content analysis, plagiarism detection, literary studies, and computational social science [2]. N-gram based language models are commonly used for authorship attribution, as they capture unique stylistic features and patterns of different authors. Language models estimate the probability of observing a word given the preceding words, allowing them to identify distinct language patterns of different authors. This assignment aims to (1) develop an author classification system using n-gram based language models and (2) evaluate different smoothing, backoff, and interpolation techniques for improved accuracy. The final classifier should be able to identify the author of a text based on language models trained on a given training set.

Data and Methodology

The dataset used in this assignment consists of text data written by four well-known authors: Jane Austen, Charles Dickens, Leo Tolstoy, and Oscar Wilde. Each author's text is stored in a separate file, with the file names indicating the author's last name followed by "_utf8.txt". It's important to note that each file is encoded using UTF-8, a widely used character encoding that ensures our program can handle a diverse range of input data. There are four files in total, and the number of sentences in each file varies. The file with the smallest dataset is austen_utf8.txt, which has 6037 sentences, and the file with the largest dataset is tolstoy_utf8.txt, which has 9667 sentences. These text files will be used to train and evaluate language models for author classification.

To build language models, we'll start by cleaning individual sentences from each text file containing author data. Using the NLTK LM package, we'll create 2-gram language models by passing cleaned sentences to the "padded_everygram_pipeline" function, generating two iterators named "train" and "vocab." After fitting a Maximum Likelihood Estimator (MLE) to these iterators, we'll apply smoothing techniques such as KneserNey Interpolation, Laplace

(add-one), and Stupid Backoff to improve the model's accuracy. Finally, we'll determine the best language model for classification by computing perplexity scores.

KneserNey Interpolated smoothing, Laplace smoothing (add one), and Stupid Backoff Estimators are three techniques used in natural language processing to estimate the probabilities of words and phrases. KneserNey Interpolated smoothing is an interpolation-based method that estimates the probability of a word based on the context in which it appears, assigning more weight to lower-order probabilities based on word frequency [3]. Laplace smoothing (add one) involves adding one count to every possible event in the training data to ensure there are no zero probabilities [3]. Stupid Backoff Estimators use a simple backoff approach to estimate probabilities of words given their context and are computationally efficient and effective in practice [3]. We'll use each of these techniques on the n-gram language model and analyze and compare their performance.

Language Model Analysis

The following table summarizes the accuracy rates obtained for each smoothing, backoff, and interpolation technique on bigram and trigram models trained on text data from four different authors: Austen, Dickens, Tolstoy, and Wilde.

Technique	Model	Austen	Dickens	Tolstoy	Wild
Kneser-Ney Interpolated Smoothing	Bigram	43.6%	39.4%	46.5%	41.7%
Kneser-Ney Interpolated Smoothing	Trigram	45.7%	39.6%	46.1%	42.5%
Laplace Smoothing	Bigram	93.5%	30.5%	68.9%	51.6%
Laplace Smoothing	Trigram	45.7%	39.6%	46.1%	42.5%
Stupid Backoff	Bigram	86.4%	40.5%	45.9%	42.5%

Absolute Discounting	Bigram	N/A	50.1%	N/A	49.8%
Laplace Smoothing	Bigram	67.4%	N/A	59.4%	N/A

From the results, we can see that Laplace smoothing on a bigram model achieves the highest accuracy rates for all texts, but it suffers from overfitting and may not generalize well to new texts. Kneser-Ney interpolated smoothing on trigram models achieves similar results to bigram models, indicating that higher-order n-grams can improve performance. Stupid backoff provides a simple and fast way to address unknown words, but it is not as effective as smoothing techniques.

Text Generation Analysis

Text generation analysis was performed on four different language models trained on literary works from four authors. Each model generated sentences based on the text prompt "i". The sentences generated are as follows:

- *austen_utf8.txt*
    - *Believe in the other delay*
    - *Do something to be observable*
    - *Suppose that you cannot be connected*
    - *Do all that she went through at finding out of mr*
    - *Will know a cheerful an affability that you*
- *dickens_utf8.txt*
    - *Remember we i was*
    - *Wait since i have nothing mrs*
    - *Suppose the moment he is quite good enough the customers and he replied steerforth retired*
    - *Knew that religious edifice immortalized by gentlemen generally remained there*
    - *Stand it gives it conveyed the gauntlet to shine in some bacon was sober oxford i was a start too*
- *tolstoy_utf8.txt*

    - *Wont change your gloomy moment proves nothing but changed at borís eyes waiting for a family her own benefit to me*
    - *Will hear it was even a while levin and delight that had known nothing fresh and bitterest sort in a*
    - *Told that narrow augesd dam raised his eyes met in her as i expect you sit on the same with*
    - *Believe in the zaraisky province then i value them he found vronsky why do what she called ive brought and*
    - *Have wished that he might kill a tone of the strange look then he first and talking and yet gone*
- *wilde_utf8.txt*
    - *Said dorian gray drove down piccadilly i felt keenly the people to seven*
    - *Am in silver dragon that effect of yours*
    - *Have committed suicide of your theories about*
    - *Am not even believed in gold might serve to forget us with the wonders happen*
    - *Believe immortal principles of the mouth with a tragedy a dreadful state that might be got up and how fearful*

Generated texts for each author vary in quality. Jane Austen's generated sentences are somewhat coherent but lack discernible meaning. Charles Dickens' generated sentences contain recognizable phrases but often lack coherence. Leo Tolstoy's generated sentences are the most coherent and meaningful, containing recognizable themes and ideas. Oscar Wilde's generated sentences contain some recognizable phrases and ideas but often lack coherence and context. Overall, Tolstoy's generated sentences are the most coherent and meaningful. Austen's and Dickens' lack coherence and context, while Wilde's contain recognizable themes and ideas but are difficult to decipher.

Discussion and Conclusion

Based on the results, it is clear that using Laplace Smoothing on a bigram model provided the highest accuracy rates for all authors except for Dickens. For Dickens, StupidBackoff on a bigram model provided the highest accuracy rate.

In terms of the generated sentences, it is difficult to make a direct comparison since the sentences generated are different for each model. However, it seems that the quality of the sentences generated for Austen and Wilde are relatively better compared to those generated for Dickens and Tolstoy. This could be attributed to the fact that Austen and Wilde have relatively smaller corpora compared to Dickens and Tolstoy, which may have affected the quality of the generated sentences.

Overall, the results suggest that using smoothing and backoff techniques can significantly improve the accuracy of n-gram models for the task of author classification. However, the choice of technique and model order may vary depending on the specific author and corpus being analyzed.

References

[1] ChatGpt. Prompt: "Briefly describe the task of author classification."

[2] Roy, N. (2019, September 13). Authorship Analysis as a Text Classification or Clustering Problem. Towards Data Science. URL: https://towardsdatascience.com/authorship-analysis-as-a-text-classification-or-clustering-problem-312549d4a4c0

[3] Speech and Language Processing. Daniel Jurafsky & James H. Martin. Copyright © 2023. All rights reserved. Draft of January 7, 2023.