## 1121 Web Search and Mining\_Project2 Report

Abstract

In Project 2, I utilized three different retrieval methods: BM25 and Language Model with various smoothing techniques. For each of these methods, I conducted tests both with and without stemming the data. This was done to observe how much of a difference stemming makes in the effectiveness of the search results. Finally, I evaluated the scores of the retrieved results to assess their performance.

Introduction

I divided the experiment into three stages:

1. First Stage: Data Indexing - I achieved control over whether the data was stemmed or not by running different files.
2. Second Stage: Model Selection - I used parameters to control the choice of retrieval model I wanted to use.
3. Third Stage: Evaluation of Retrieved Results - The results obtained from the search were sent to TREC\_EVAL for evaluation, and conclusions were drawn based on these results.

Method

For testing purposes, I employed the following methods:

1. OKAPI BM25
2. Language Modeling with Laplace Smoothing
3. Language Modeling with Jelinek-Mercer Smoothing

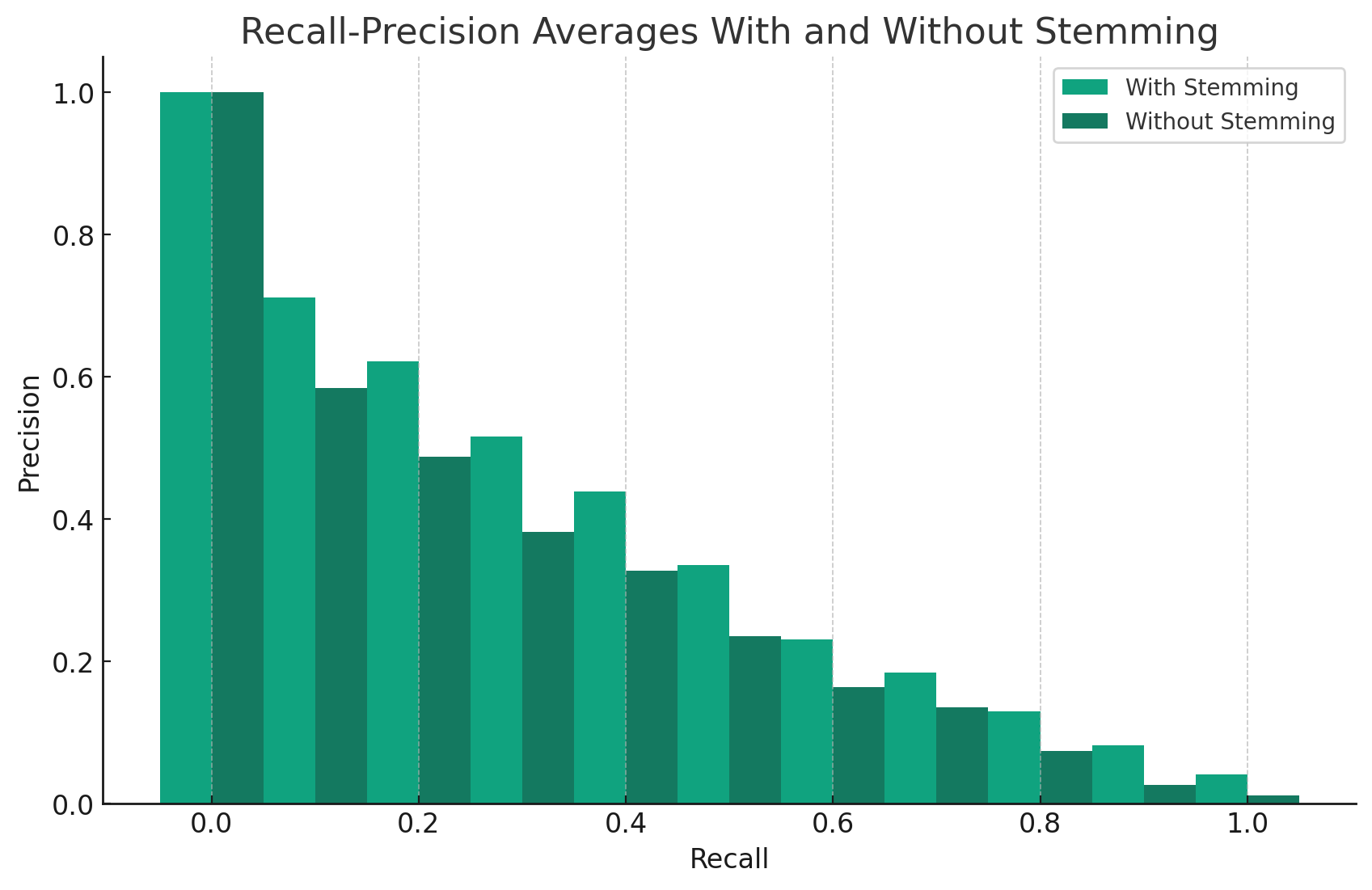
Data

I use the WT2G data collection as a database of searching engine.

And use the 40 queries for the corpus as the keywords.

BM25 result

|  |  |
| --- | --- |
| With stemming | Without stemming |
|  |  |



We can draw the following summarize：

With Stemming：

1. A greater number of documents and relevant documents were retrieved.
2. There was a higher average interpolated precision-recall at most recall levels.
3. For all retrieved documents, the average precision (non-interpolated) was higher.

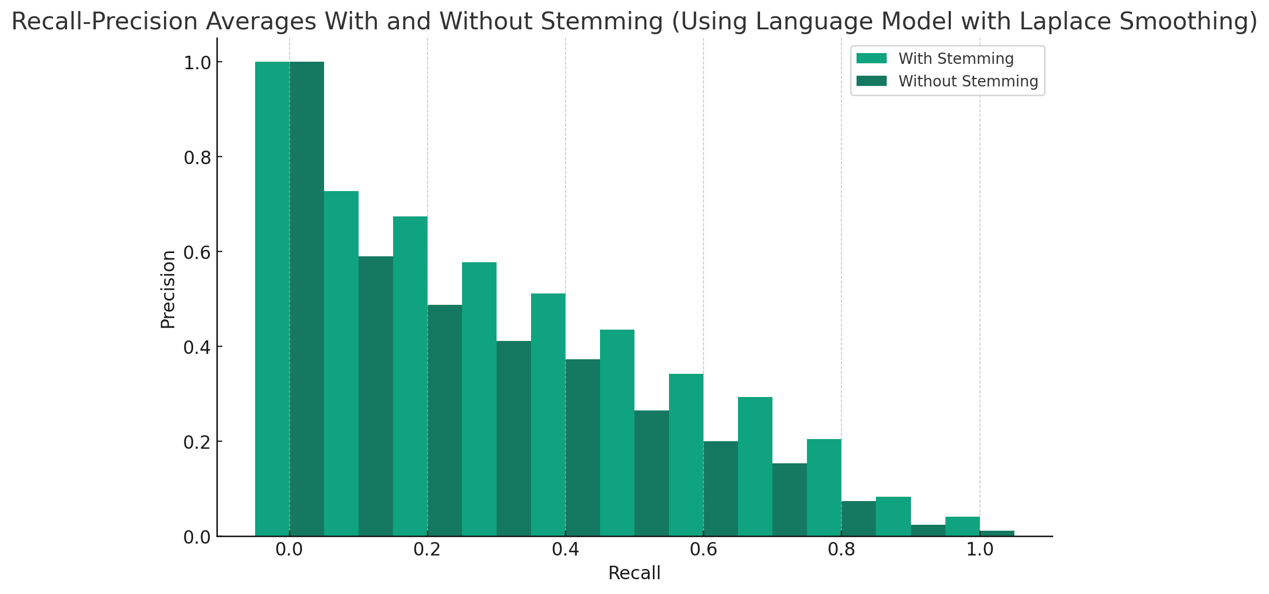
Without Stemming：

1. Fewer documents and relevant documents were retrieved.
2. There was a lower average interpolated precision-recall at most recall levels.
3. For all retrieved documents, the average precision (non-interpolated) was lower.

From these results, we can conclude that stemming improves the performance of the BM25 retrieval algorithm. This is evident from the higher precision at various recall levels, indicating that more relevant documents were retrieved earlier in the search process.

Language model with Laplace smoothing

|  |  |
| --- | --- |
| With stemming | Without stemming |
|  |  |

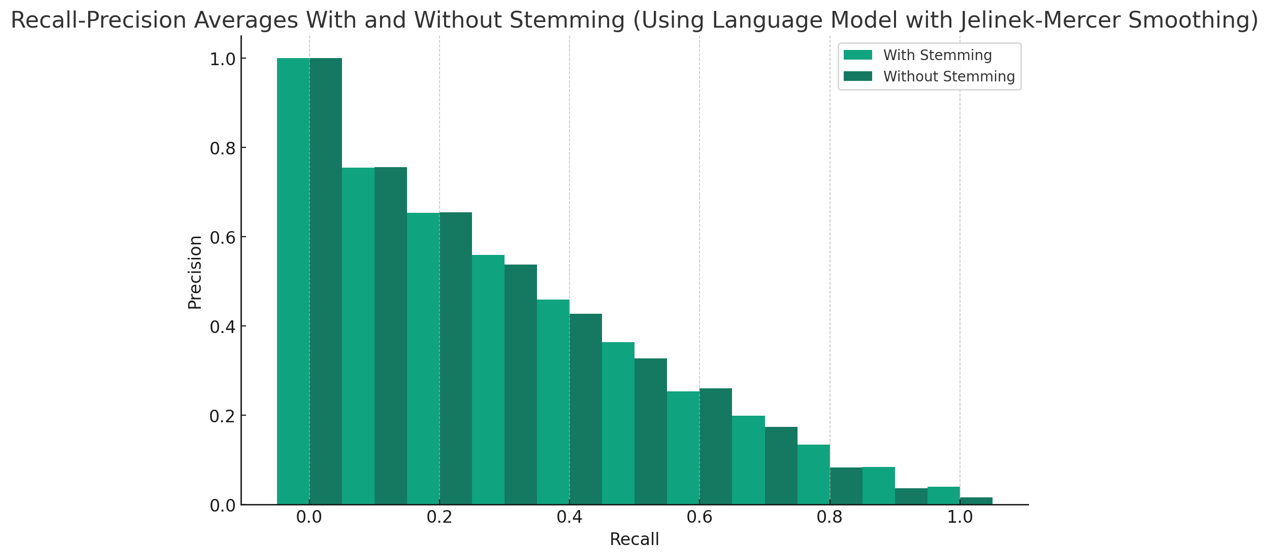


We can draw the following summarize：

From the graph, it's evident that using stemming resulted in higher precision at most recall levels. This indicates that stemming can enhance retrieval performance, making it easier to retrieve relevant documents. This finding aligns with the general understanding in information retrieval that stemming can improve the precision of a system.

Language model with Jelinek-Mercer smoothing (0.8 of the weight)

|  |  |
| --- | --- |
| With stemming | Without stemming |
|  |  |



We can draw the following summarize：

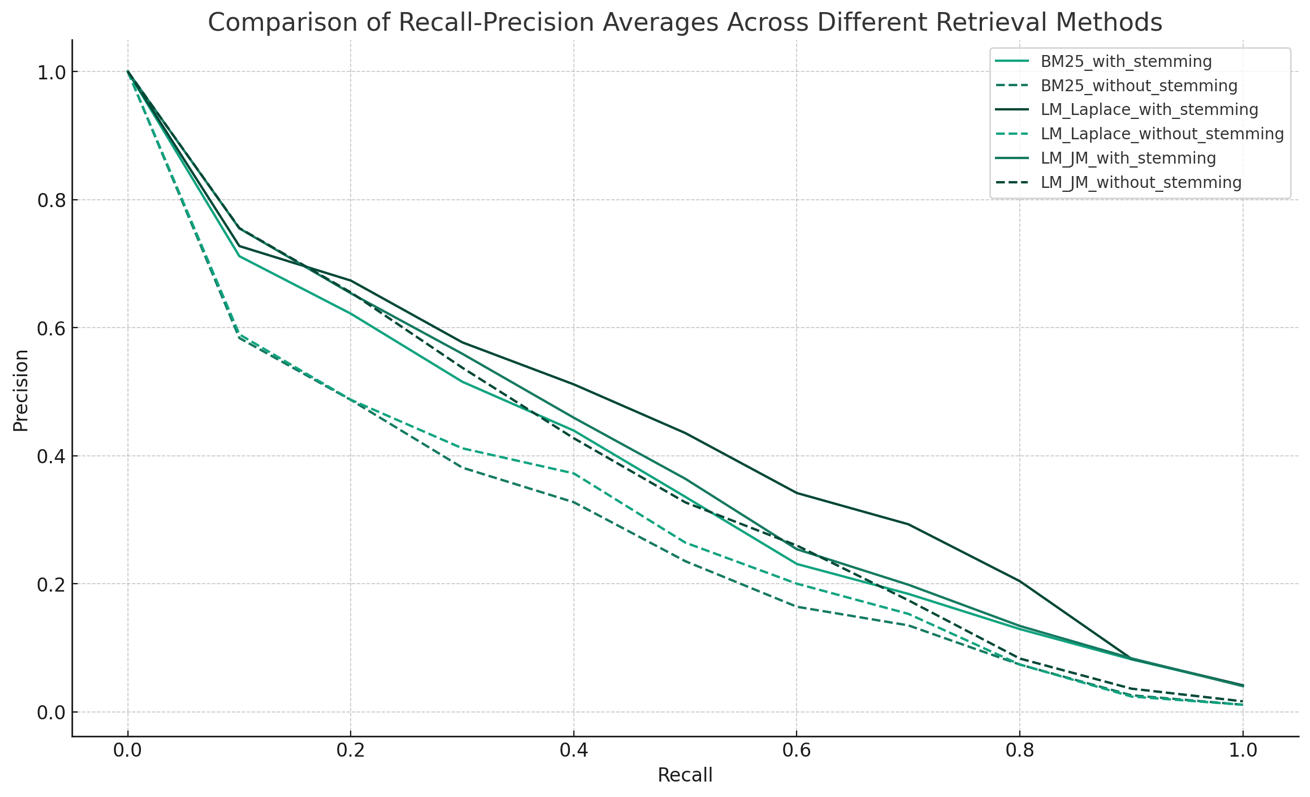
The histogram demonstrates the average precision-recall values for text retrieval using a language model with Jelinek-Mercer smoothing (with a weight of 0.8), comparing results

with and without the use of stemming. The graph reveals：

The precision with stemming is slightly higher than without stemming at most recall levels.At higher recall rates, the precision difference between the two methods is smaller. However, at medium to low recall rates, the precision with stemming is notably higher.

This further confirms the positive impact of stemming on enhancing retrieval effectiveness, particularly in the early stages of retrieval where higher precision is crucial.

Conclusions



Based on the experimental data provided for the three retrieval methods — BM25 retrieval, Language Model with Laplace Smoothing, and Language Model with Jelinek-Mercer Smoothing — we can summarize as follows：

BM25 Retrieval：

Stemming significantly improves retrieval precision, especially at lower recall rates.

Language Model with Laplace Smoothing：

The use of stemming also enhances retrieval effectiveness, leading to more relevant documents being retrieved.

Language Model with Jelinek-Mercer Smoothing：

Stemming likewise increases precision, though the difference is less pronounced at higher recall rates.

From the results of these three retrieval techniques, it's evident that stemming is an effective text preprocessing step. It consistently enhances retrieval performance across different search algorithms.

Beside, My friends and I observation is quite intriguing. It seems that encapsulating the IndexSearcher and setSimilarity for the Jelinek-Mercer smoothing within a function allows the program to run smoothly. However, when these elements are not used within a function, errors occur during the execution of search.py.

This issue might be related to the scope of variables or the initialization of certain objects in Python. When these elements are within a function, they are isolated and have a local scope, which could prevent conflicts or reinitialization issues that might occur in a global scope.

To better understand the issue and possibly resolve it, a detailed analysis of the error messages and the specific code structure would be required. Understanding how the IndexSearcher and setSimilarity are implemented and interact with other parts of your code outside the function could provide insights into why the error occurs.

|  |  |
| --- | --- |
| Used function(correct) | Unused function(error) |
|  |  |