TTDS Coursework 2

Meghna Raje

December 4, 2022

1 INTRODUCTION

1.1 General Overview

The report is divided into three parts IR Evaluation, Text Analysis and Text Classification

1. IR Evaluation - The effectiveness of a search engine or database in returning results that answer a user's query is measured by evaluation measures for an information retrieval (IR) system. Built a module to evaluate IR systems using different retrieval scores
2. Token and Text Analysis- MI and X^2 are the two methods used for token analysis. To assign token in a document to a specific topic, topic models Latent Dirichlet Allocation (LDA) is used.
3. Text Classification- Implemented a Sentiment analyser which aims to classify the text based on its polarity

1.2 A brief commentary on the system as a whole and learning from implementation

1. Implementing all the three parts from scratch gave a good understanding of on how to evaluate a search engine, how to analyse a text using different methods and implementing a sentiment analyser
2. Learning from the implementation:
   (a) Understanding of different techniques used for evaluating a system
   (b) Overview on Mi, X^2 and LDA methods for text analysis
   (c) Good understanding of machine learning algorithms for classification
   (d) Strong knowledge of feature engineering for improvising the baseline model

1.3 Brief introduction of the implemented code

For each task there a separate class defined for it. First task that is related to IR Evaluation has a class created for it named as EVAL(). This class has different methods for calculating score for each techniques like precision, recall and kDCG. Second task's class name is TextAnalysis(). It has methods defined inside it to calculate Mi score, X^2 score and LDA. Third task that is related to classification has a class named TextCLassfication() defined for it. It has different methods like baseline_model() and baseline_model_extra_preprocessing()

1.4 Challenges faced when implementation

1. Building a DCG module was bit challenging
2. Understanding the proper meaning of the terms N11, N01 for calculating Mi and Chi score was bit difficult in the beginning
3. Selecting a model to improvise the score was challenging as every classification technique was giving less score as compared to the baseline initially

1.5 Ideas on how to improve and scale implementation

1. Data given in the task 3 is imbalanced so it can be balanced using different techniques like generating synthetic samples ,under sampling and over sampling techniques

2. Using different technique for evaluating the systems like Cohen's kappa

2 IR Evaluation

Results for 10 queries across 6 systems are used in IR evaluation. The true relevant results in qrels.csv are comapred with system results.csv. P@10, R@50, r-precision, AP, nDCG@10, and nDCG@20 are six different types of IR evaluation measurements.

$$P@10 : \text{precision at cutoff 10,} P@k = \frac{\text{Total number of relevant documents retrieved @ k}}{\text{Total number of relevant documents @ k}} \tag{1}$$

$$R@10 : \text{recall at 50, with the formula, } R@k = \frac{\text{Total number of relevant documents retrieved @ k}}{\text{Total number of relevant documents @ k}} \tag{2}$$

$$r - precision : \text{precision @ rank r for a query with known r relevant documents} \tag{3}$$

$$\text{AP: average precision: } = \frac{1}{r}\sum_{k=1}^{n} P(k) \times \text{rel}(k) \tag{4}$$

where, r is the number of relevant docs for a given query, n is the number of documents retrieved, P(k) is the precision @ k, rel(k):1 retrieved doc @ k is relevant, 0 otherwise.

nDCG@10, nDCG@20: normalised discount cumulative gain at 10 and 20

$$\begin{aligned} nDCG@k =& \frac{DCG@k}{iDCG@k} \\ DCG@k =& rel_1 + \sum_{i=2}^{k} \frac{\text{rel}_i}{\text{log2(i)}} \end{aligned} \tag{5}$$

iDCG@k = ideal discount cumulative gain@k

These methods helps to evaluate IR from different perspectives.

Implemented IR evaluation metrics includes precision@10, recall@50, r-precision, average precision, nDCG@10, and nDCG@20 for six different systems. The mean scores for each metric for each system are displayed in the table below:

	precision@10	recall@50	r-precision	average precision	nDCG@10	nDCG@20
S1	0.39	0.834	0.401	0.4	0.363	0.485
S2	0.22	0.867	0.253	0.3	0.2	0.246
S3	0.41	0.767	0.449	0.451	0.42	0.511
S4	0.08	0.189	0.049	0.075	0.069	0.076
S5	0.41	0.767	0.358	0.364	0.333	0.424
S6	0.41	0.767	0.449	0.445	0.4	0.49

Table 1: System 1-6 Mean Score

Table 2 shows the best and the second best system with their p-value for each score. We perform a two-tailed t-test with the following hypotheses: Let A be the random variable of the scores of the best system, and let B be the random variable of the scores of the second-best system.

H0 : A = B (Null Hypothesis)

H1 : A != B (Alternative Hypothesis)

The test statistic is then:

$$t = \frac{\overline{A - B}}{\sigma(A - B)}.\sqrt{N} \tag{6}$$

where $\sigma(A - B)$ is the standard deviation of A - B and N is the number of samples.

	precision@10	recall@50	r-precision	average precision	nDCG@10	and nDCG@20
best system	S3,S5,S6	S2	S3,S6	S3	S3	S3
second best system	S1	S1	S1	S6	S6	S6
p-value	0.888	0.702	0.759	0.967	0.882	0.867

Table 2: Best and second best systems with p-value

	precision@10	r-precision
best system	S3	S3
other best system	S5	S6
mean score	0.41	0.449
p-value	1	1

Table 3: Two best systems with p-value

Table 2 shows the resulting p-values for best and second best system. For P@10 and r-Precision, there are multiple best systems which means that they are identical. Therefore for p-test any of the best system can be considered. Table 3 shows the p-value for two best system. As illustrated in Table 2 and Table 3, the p-values for all the system are significantly larger than $alpha = 0.05$, which means that none of the six measures can be used to reject H0 i.e we cannot determine that the best system are statistically significantly better than the second best system for that scores.

3 TOKEN ANALYSIS

Corpus	Mi Score	X^2 Score
Old Testament	[('jesus', 0.037), ('israel', 0.031), ('king', 0.026), ('lord', 0.025), ('christ', 0.019), ('believ', 0.0179), ('god', 0.016527197916528027), ('muhammad', 0.015), ('son', 0.012), ('torment', 0.012)]	[('jesus', 1464.155), ('lord', 1122.995), ('israel', 1095.934), ('king', 946.774), ('god', 788.884), ('christ', 779.406), ('believ', 750.066), ('muhammad', 608.737), ('faith', 543.937), ('son', 538.666)]
New Testament	[('jesus', 0.052), ('christ', 0.0319), ('lord', 0.022), ('discipl', 0.014), ('israel', 0.013), ('peopl', 0.009), ('king', 0.009), ('peter', 0.009), ('paul', 0.009), ('land', 0.009)]	[('jesus', 3026.683), ('christ', 1772.096), ('lord', 874.182), ('discipl', 819.833), ('peter', 528.992), ('paul', 528.992), ('thing', 503.196), ('israel', 451.221), ('spirit', 436.662), ('john', 408.176)]
Quran	[('god', 0.031), ('muhammad', 0.028), ('believ', 0.019), ('torment', 0.019), ('messeng', 0.015), ('revel', 0.013), ('king', 0.013), ('israel', 0.013), ('unbeliev', 0.012), ('guidanc', 0.012)]	[('muhammad', 1852.132), ('god', 1710.674), ('believ', 1344.232), ('torment', 1332.811), ('messeng', 1062.253), ('revel', 941.556), ('unbeliev', 848.865), ('guidanc', 810.932), ('disbeliev', 786.116), ('unjust', 765.188)]

Table 4: Top 10 highest scoring words for MI and X^2 Score

In token analysis, the two methods MI and X^2 are used on the files train_and_dev.tsv, which contain verses from the Quran, Bible New Testament (NT), and Bible Old Testament (OT). Table 4 lists the top 10 words for each corpus with the highest MI and X^2 scores. From the table we can see that the selected words are almost similar and their rankings vary to some extend. For example , for Quran corpus, the word "god" is ranked 1st in Mi but 2nd in X^2. Also word muhammad which is not in OT has scored high.
OT mentions about *lord, believe and israel*, NT talks about *jesus christ his discipline* and *two people peter and paul* and Quran frequently discusses about *muhammad, message and believe.* From the ranking we get the intuition about the base idea about each corpus.

4 TOPIC ANALYSIS

Corpus	Topic Id	Score	Top 10 token with their probabilities
Old Testament	16	0.094	0.160*"king" + 0.141*"lord" + 0.062*"word" + 0.038*"israel" + 0.034*"god" + 0.028*"peopl" + 0.027*"servant" + 0.026*"hear" + 0.021*"walk" + 0.019*"peac"
New Testament	18	0.080	0.087*"good" + 0.069*"evil" + 0.040*"fire" + 0.038*"abomin" + 0.034*"punish" + 0.033*"lord" + 0.032*"prepar" + 0.026*"life" + 0.025*"day" + 0.024*"woe"
Quran	5	0.084	0.093*"earth" + 0.057*"heaven" + 0.052*"lord" + 0.045*"god" + 0.039*"kingdom" + 0.033*"wine" + 0.030*"fall" + 0.026*"thing" + 0.025*"spoken" + 0.024*"wisdom"

Table 5: Top 10 tokens and their probability scores

From looking at the top 10 ranking tokens in Table 5, the most suitable title for OT's could be *People of Israel*, the NT's top topic title could be *Good and evil*, and the Quran's top topic title could be *Heaven on earth.*
We can draw a broad conclusion about the potential topics for each corpus using the LDA model. One word might perform well in the MI and X^2 methods, but it might not add anything to the topic analysis because it shows equal likelihood across the three corpora.
Topic Id 5 seems to be more common in Quran and OT (it is ranked 3rd in OT). The high probability word "earth" appeared 325 times in Quran, 642 times in OT and 154 times in NT. The word "lord" appeared 846 times in Quran, 5473 times in OT and 568 times in NT. We can see that the OT and Quran use these words more frequently than the NT.

5 TEXT CLASSIFICATION

In text classification, the train_and_dev.tsv dataset is firstly divided into a training set (90%) and a development set (10%) in order to train the baseline model. Then, as part of the preprocessing, punctuation ,casefold and hyperlink removal are implemented. The BOW features are extracted and the SVM classifier is trained on a training set with C = 1000. The macro-f1 score reaches **0.542** for the development set.

Below are three examples from the development set that the baseline model labels incorrectly:

We can see that the model frequently classifies the given input as neutral. The positive(5983) and negative(3385) sentiment tweets are significantly lesser in number than the neutral(8792) tweets, which may be the cause of this. As the classifier learns more about the neutral tweets, the input is more likely to be placed in this sentiment category.
By looking at the incorrectly labelled data we can infer that most of the wrong predictions are for stop words like *a, that, the, and, has, else.* So to increase the performance of the model these words need to be removed. By getting rid of these words, we can make our tweets more focused on the important

True Sentiment	Predicted Sentiment	Data labelled incorrectly
neutral	positive	[’leannynr’, ’i’, ’only’, ’go’, ’once’, ’a’, ’week’, ’or’, ’so’, ’that’, ’may’, ’be’, ’because’, ’i’, ’work’, ’at’, ’home’, ’and’, ’going’, ’to’, ’dunkin’, ’means’, ’leaving’, ’my’, ’house’]
negative	neutral	[’itsbl0ndie’, ’i’, ’still’, ’think’, ’it’, ’was’, ’fake’, ’i’, ’always’, ’thought’, ’nicki’, ’to’, ’be’, ’the’, ’type’, ’to’, ’call’, ’you’, ’out’, ’on’, ’a’, ’random’, ’tuesday’, ’not’, ’at’, ’an’, ’awards’, ’show’]
positive	neutral	[’all’, ’the’, ’talk’, ’will’, ’be’, ’about’, ’nicki’, ’and’, ’miley’, ’tomorrow’, ’nothing’, ’else’, ’has’, ’been’, ’talk’, ’worthy’, ’vmas’]

Table 6: Examples of incorrect label

information by eliminating the low-level information.
To improvise the performance on the dev set following are the techniques tried -

1. Extreme pre-processing that is removing stop words and stemming using SVC classifier
2. Random Forest Classifier
3. Extreme pre-processing + Logistic regression

	Baseline	Extreme Preprocessing + SVM	Random Forest Classifier	Extreme Preprocessing + Logistic Regression
Dev Set	0.542	0.511	0.521	**0.61**
Test Set	0.522	0.433	0.489	**0.525**

Table 7: F1 Score for different techniques

Table 7 shows that the performance is best improved for Extreme pre-processing + Logistic regression compared to the baseline system. There is gain of **7%** observed on dev set and gain of **3%** on test set for F1 macro.