TTDS Coursework2

s2154448

1 Overall

1.1 Code

When I was writing the first part of the code (IR Evaluation), I spent a lot of time thinking about how to convert the data in the *qrels.csv* and *system_results.csv* files into the format I wanted, which would improve the efficiency of my code when calculating P@10, R@50 and so on.
Take P@10 as an example, I have created a list called *query_rel_doc* in advance, which can store relevant documents' number for each query. First, I split *system_results.csv* into 6 systems based on system number. Then for each system, I split it into 10 queries based on query number and compute for each query. In calculating P@10, the data in *query_rel_doc* can be used without re-traversing the data in *qrels.csv*. In addition to this, *query_rel_doc* can also be used in the calculation of R@50, r-precision, AP and so on, which saves a lot of computation time for me.
In calculating nDCG@10 and nDCG@20, to generate G lists and iG lists more efficiently, I create a dictionary within a dictionary called *doc_rel_dict* based on *qrels.csv*, which looks like {query_number: {doc_number: relevance_value, …}, …}. Therefore, I am able to get relevance value early by using *doc_rel_dict[query_number][doc_number]*.

1.2 Challenge

When I calculate Mutual Information (MI), the program keeps coming up with calculation errors. I found out that it is because the x in $\log_2(x)$ may equal zero during the calculation. Therefore, I improved my program by adding the judgment statement (when $x = 0$, make $\log_2(x) = 0$).

$$MI = \frac{N_{11}}{N}\log_2\left(\frac{NN_{11}}{N_{1.}N_{.1}}\right) + \frac{N_{01}}{N}\log_2\left(\frac{NN_{01}}{N_{0.}N_{.1}}\right) + \frac{N_{10}}{N}\log_2\left(\frac{NN_{10}}{N_{1.}N_{.0}}\right) + \frac{N_{00}}{N}\log_2\left(\frac{NN_{00}}{N_{0.}N_{.0}}\right)$$

1.3 Missing part

In the third part (Text Classification), I have tried only three improvements: applying stemming and stopping, using tf-idf features instead of just counts, trying another classifier(SVM). There are many more enhancements that I have not bothered to implement, such as duplicating hashtags words, getting more training data and so on.

2 IR Evaluation

The average scores achieved for each system is below.

System	P@10	R@50	r-precision	AP	nDCG@10	nDCG@20
1	0.390	0.834	0.401	0.400	0.363	0.485
2	0.220	0.867	0.252	0.300	0.200	0.246
3	0.410	0.767	0.448	0.451	0.420	0.511
4	0.080	0.189	0.049	0.075	0.069	0.076
5	0.410	0.767	0.358	0.364	0.332	0.424
6	0.410	0.767	0.448	0.445	0.400	0.491

Table 2.1. The average scores achieved for each system

Based on the average scores achieved for each system, we can select the best and second system for each evaluation. The best and second system with each evaluation shows below:

	P@10	R@50	r-precision	AP	nDCG@10	nDCG@20
best system	3	2	3	3	3	3
best score	0.410	0.867	0.448	0.451	0.420	0.511
second system	6	1	6	6	6	6
second score	0.410	0.834	0.448	0.445	0.400	0.491

Table 2.2. The best and second system with each evaluation

To find whether the best system is statistically significantly better than the second system, we can calculate p-value in the t-test. The p-value is a measure of the probability that an observed difference could have occurred by chance. If the p-value is higher than a chosen threshold (e.g., 0.05), the observation is likely to have occurred by chance. Firstly, we can establish test hypotheses and determine the test level.

$$\begin{cases} H_0: \mu = \mu_0 \\ H_1: \mu \neq \mu_0 \\ \alpha = 0.05 \end{cases}$$

The t-test is a test for the degree of difference between two means of a small sample. It uses the theory of t-distribution to infer the probability of a difference occurring and thus determine whether the difference between the two means is significant. The calculation formula is below.

$$t = \frac{\bar{x} - \mu}{s/\sqrt{n}}$$

$\bar{x}$ is sample mean, μ is another overall sample mean, s is sample variance and n is number of samples.
We can use *ttest_ind* in *scipy.stat* to calculate the calculated t-statistic and the two-tailed p-value. The result for each evaluation is below.

	P@10	R@50	r-precision	AP	nDCG@10	nDCG@20
t-statistic	0	0.388	0	0.043	0.151	0.168
p-value	1	0.703	1	0.967	0.882	0.869

Table 2.3. The p-value for each evaluation

All the p-values for each evaluation are much higher than 0.05, which means we cannot conclude that the best system is significantly better than the second system for each evaluation.

3 Token Analysis

We compute the Mutual Information and X^2 scores for all tokens for each the three corpora, and the top 10 highest scoring words for each method for each corpus is below.

Corpus		1	2	3	4	5	6	7	8	9	10
OT	MI	jesus	israel	king	lord	christ	believ	god	muhammad	son	torment
	X^2	jesus	lord	israel	king	god	christ	believ	muhammad	faith	son
NT	MI	jesus	christ	lord	discipl	israel	king	peopl	peter	paul	land
	X^2	jesus	christ	lord	discipl	peter	paul	thing	israel	spirit	john
Quran	MI	god	muhammad	believ	torment	messeng	revel	king	israel	unbeliev	guidanc
	X^2	muhammad	god	believ	torment	messeng	revel	unbeliev	guidanc	disbeliev	unjust

Table 3.1. Top 10 words for each method for each corpus

3.1 Differences between the ranking produced by the two methods

When I observed the rankings produced by the two methods (MI and X^2), I found that there is no significant difference between the 10 highest scoring words measured by these two methods for each corpus. The order of some words is different. For example, in OT corpus, the word ‘lord’, which is ranked 4th in MI, is ranked 2nd in X^2. Some words only appear in MI, while others only appear in X^2, such as ‘faith’ in X^2 for OT corpus and ‘unbeliev’ in MI for Quran corpus.

3.2 What I learn about the three corpora from these ranking

When I compared the ranking words between the three corpora, I found that the OT and NT corpus may have similar content. This is because there are many similar words in the top 10 ranking words of OT and NT corpus, such as ‘jesus’, ‘lord’ and ‘christ’. As far as I am concerned, MI and X^2 calculate the degree of association (positive or negative correlation) between the token and the corpus, the content of the corpus can be reflected by the token with high ranking words. For example, the word ‘jesus’ does not exist in OT corpus, but it ranks the first in both MI and X^2 for OT corpus, which means it has the highest degree of negative correlation.

4 Topic Analysis

The top 10 tokens and their probability scores for each of the 3 topics which is most associated with each corpus show below.

corpus	topic	1	2	3	4	5	6	7	8	9	10
OT	topic18	god	lord	judgment	peopl	evil	children	day	abraham	glori	afraid
		0.253	0.062	0.052	0.037	0.029	0.023	0.021	0.020	0.019	0.018
	topic11	god	lord	peopl	truth	deed	book	forgiv	turn	great	merci
		0.101	0.074	0.061	0.047	0.041	0.036	0.035	0.033	0.032	0.028
	topic12	messeng	son	spirit	prophet	testifi	wait	wife	man	name	daughter
		0.161	0.149	0.083	0.080	0.024	0.023	0.022	0.019	0.018	0.017
NT	topic13	god	believ	receiv	lord	hear	power	live	fear	soul	day
		0.136	0.081	0.058	0.042	0.030	0.029	0.029	0.028	0.026	0.025
	topic16	word	suffer	reward	hous	lord	thing	father	speak	find	happi
		0.087	0.066	0.061	0.036	0.036	0.035	0.034	0.027	0.027	0.020
	topic18	god	lord	judgment	peopl	evil	children	day	abraham	glori	afraid
		0.253	0.062	0.052	0.037	0.029	0.023	0.021	0.020	0.019	0.018
Quran	topic11	god	lord	peopl	truth	deed	book	forgiv	turn	great	merci
		0.101	0.074	0.061	0.047	0.041	0.036	0.035	0.033	0.032	0.028
	topic18	god	lord	judgment	peopl	evil	children	day	abraham	glori	afraid
		0.253	0.062	0.052	0.037	0.029	0.023	0.021	0.020	0.019	0.018
	topic13	god	believ	receiv	lord	hear	power	live	fear	soul	day
		0.136	0.081	0.058	0.042	0.030	0.029	0.029	0.028	0.026	0.025

Table 4.1. Top 10 tokens of top 3 topics for each corpus

4.1 My own labels for the 3 topics

	topic	my own label
OT	topic18	judge people
	topic11	forgive people
	topic12	family spirit
NT	topic13	power in belief
	topic16	suffering and happy
	topic18	judge people
Quran	topic11	forgive people
	topic18	judge people
	topic13	power in belief

Table 4.2. Labels for the 3 topics

4.2 What the LDA model tell me about the corpus

For topic18, it is presented in OT, NT and Quran, which rank the first, the third and the second respectively.

As for topic11, it appears to be common in OT and Quran but not NT. The high probability words from topic11 are [god, lord, peopl, truth, deed, book, forgiv, turn, great, merci]. In terms of topic13, it appears to be common in NT and Quran but not OT. The high probability words from topic13 are

[god, believ, receiv, lord, hear, power, live, fear, soul, day].
In comparison to MI and X^2, the LDA model generated 20 topics and 10 terms are linked together within each topic. In MI and X^2, the 10 terms are not linked and the terms in the same document are independent of each other.

5 Classification

Three examples of misclassification are shown below.
1st: ['god', 'is', 'not', 'a', 'man', 'that', 'he', 'should', 'lie', 'nor', 'a', 'son', 'of', 'man', 'that', 'he', 'should', 'repent', 'has', 'he', 'said', 'and', 'will', 'he', 'not', 'do', 'or', 'has', 'he', 'spoken', 'and', 'will', 'he', 'not', 'make', 'it', 'good']
2nd: ['then', 'they', 'arose', 'early', 'in', 'the', 'morning', 'and', 'swore', 'an', 'oath', 'with', 'one', 'another', 'and', 'isaac', 'sent', 'them', 'away', 'and', 'they', 'departed', 'from', 'him', 'in', 'peace']
3rd: ['my', 'hand', 'has', 'found', 'like', 'a', 'nest', 'the', 'riches', 'of', 'the', 'people', 'and', 'as', 'one', 'gathers', 'eggs', 'that', 'are', 'left', 'i', 'have', 'gathered', 'all', 'the', 'earth', 'and', 'there', 'was', 'no', 'one', 'who', 'moved', 'his', 'wing', 'nor', 'opened', 'his', 'mouth', 'with', 'even', 'a', 'peep']
As far as I am concerned, in these misclassified documents, the words in these documents do not have a clear topic, so they can be classified as OT, NT or Quran.

In the baseline system, I use *LinearSVC* model with *C*=1000 and *max_iter*=5000. To improve the performance compared to the baseline system, I have tried 3 approaches. The first is applying stemming and stopping, and the second is using TF-IDF features instead of just counts. The last one is trying another classifier, such as *SVC* model.

For the baseline system (*LinearSVC*), the result is below.

split	p-quran	r-quran	f-quran	p-ot	r-ot	f-ot	p-nt	r-nt	f-nt	p-macro	r-macro	f-macro
train	0.998	0.994	0.996	0.997	0.991	0.994	0.972	0.993	0.982	0.989	0.992	0.991
dev	0.826	0.852	0.838	0.917	0.874	0.895	0.714	0.792	0.751	0.819	0.839	0.828
test	0.843	0.831	0.837	0.921	0.888	0.904	0.744	0.827	0.783	0.836	0.848	0.841

Table 5.1. Result for baseline

For the improved system (applying stemming and stopping), the result is below.

split	p-quran	r-quran	f-quran	p-ot	r-ot	f-ot	p-nt	r-nt	f-nt	p-macro	r-macro	f-macro
train	0.954	0.965	0.960	0.963	0.968	0.965	0.908	0.889	0.889	0.942	0.940	0.941
dev	0.804	0.876	0.838	0.923	0.891	0.907	0.758	0.780	0.769	0.828	0.849	0.838
test	0.780	0.853	0.815	0.915	0.894	0.904	0.771	0.768	0.770	0.822	0.838	0.830

Table 5.2. Result for system which applies stemming and stopping

After applying stemming and stopping, the Macro-F1 score increased by 0.010 on the dev set, while decreased by 0.011 on the test set.

For the improved system (instead of just counts), the result is below.

split	p-quran	r-quran	f-quran	p-ot	r-ot	f-ot	p-nt	r-nt	f-nt	p-macro	r-macro	f-macro
train	1.000	0.999	1.000	0.997	0.998	0.997	0.993	0.991	0.992	0.997	0.996	0.996
dev	0.858	0.865	0.861	0.920	0.902	0.911	0.757	0.793	0.775	0.845	0.853	0.849
test	0.889	0.839	0.863	0.920	0.916	0.918	0.781	0.822	0.801	0.863	0.859	0.861

Table 5.3. Result for system which uses TF-IDF features

After using TF-IDF features instead of just counts, the Macro-F1 score increased by 0.021 on the dev set and increased by 0.020 on the test set.

For the improved system (using SVC model), the result is below.

split	p-quran	r-quran	f-quran	p-ot	r-ot	f-ot	p-nt	r-nt	f-nt	p-macro	r-macro	f-macro
train	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000
dev	0.933	0.903	0.918	0.878	0.825	0.851	0.934	0.960	0.947	0.915	0.896	0.905
test	0.924	0.900	0.912	0.881	0.832	0.856	0.928	0.953	0.940	0.911	0.895	0.902

Table 5.4. Result for system which uses SVC model

After using, the Macro-F1 score increased by 0.077 on the dev set and increased by 0.061 on the test set.

In conclusion, we are not able to improve the performance on the test set while applying stemming and stopping. However, when using TF-IDF features and using SVC model, the Macro-F1 score increased by 0.020 and 0.061 respectively. Of the three approaches I have tried, the one using the SVC model has resulted in the most significant performance improvement.