Coursework 2 (S2311720)

1. Introduction to the overall coursework

In this coursework, we have three parts to complete, IR Evaluation, Text Analysis, and text classification and I will briefly introduce how I did in each of these three tasks.

a. In IR Evaluation, there are 10 queries results for each of 6 systems in system_result.csv file and true results in qrels.csv file, we did 6 kinds of evaluation methods in our code that are P@10, R@50, r-precision, AP, nDCG@10 and nDCG@20. By following the formulas in lecture slides, we can calculate all these 6 evaluations number, and finally we need to print these out in a text file called ir_eval.csv and use the provided script file to test its format. By completing this part, I learned how to evaluate different IR systems with different values like average precision and discount cumulative gain. There are also some challenges while doing this task, one is that we need to cast some variable types like int and list to use proper methods and avoid having error messages.

b. In text analysis, we have three separate corpus Quran, New Testament, and Old Testament from file train_and_dev.tsv, I first separate the file to corpus name and text and do the preprocessing (remove stop words, stemming, and tokenization) on the text, then we applied two formulas from the lecture slides called Mutual Information and Chi-square to calculate these two scores for each of these three corpora and have a raked list of results. Then by having these results, we could know what the main context and idea of each corpus is. Then we runed LDA for each corpus by using genism library, we first put the preprocessed text of each corpus into genism’ s Dictionary method, then we used doc2bow method to apply bag-of-words method into the corpus, then we used lda to get document topics and print it out. By doing this part of task, I learned methods from genism library which can make my code more efficiency and knowing how to get a proper document topic probability by applying LDA method. Some challenges like how to merge dictionaries are solved during the coding process.

c. In text classification task, we have a training data which contains many twitters text with their corresponding tag: positive, neutral, or negative, and we will use support vector machine to do the classification task. For the baseline, I just do the tokenize work and separate them into three list, documents, categories and vocabularies, each of these indexes are corresponding to each other, then, I split training and development set into 80% and 20% and shuffled them. Then I converted them into matrix and put into the SVM for training. After the training process is finished, I printed out the precision, recall, and f1-score for the baseline. And we get a macro f1 score 0.6 on development set, then for the improvement, I will try to change parameters in SVM, doing more preprocessing, and more classification learning methods. One of thing that I learned from it is that some preprocessing method like remove stop words does not necessary increase our f1 macro score, and the meaning of C in SVM model.

2. IR Evaluation

Here are the mean scores of each system for their P@10, R@50, r-precision, AP, nDCG@10 and nDCG@20

	P@10	R@50	r-precision	AP	nDCG@10	nDCG@20
System1	0.390	0.834	0.401	0.400	0.363	0.485
System2	0.220	0.867	0.253	0.300	0.200	0.246
System3	0.410	0.767	0.448	0.451	0.420	0.511
System4	0.080	0.189	0.049	0.075	0.069	0.076
System5	0.410	0.767	0.358	0.364	0.332	0.424
System6	0.410	0.767	0.448	0.445	0.400	0.491

Then we calculate the p-value of the two tailed t-test result:

	P@10	R@50	r-precision	AP	nDCG@10	nDCG@20
Best sys	S3, S5, S6	S2	S3, S6	S3	S3	S3
Second sys	S1	S1	S1	S6	S6	S6
p-value	0.888	0.703	0.759	0.967	0.882	0.869

Assuming the difference of score between best system and second system obey the normal distribution, we can use the p-value approach to hypothesis testing to indicate if the system is statistically significantly better than the second system. Null hypothesis is that the mean score difference for the best system and the second is equal, and the alternative hypothesis is that the mean score difference for the best and the second is not equal. Here we have p value all much larger than the significant value, which is 0.05, so we are failed to reject null hypothesis which means the best system is not statistically significantly better than the second system. Also, we could see that in the first table, the differences of scores between the best system’ s scores and the second-best system’ s score is less than 0.05.

3. Token Analysis

Here is the table for top 10 highest scoring words for MI and Chi-square

	Mutual Information									
OT	lord	king	israel	jesu	land	son	christ	peopl	offer	muhammad
NT	jesu	christ	lord	thing	israel	discipl	king	faith	land	peter
Quran	god	peopl	lord	muhammad	messeng	believ	torment	day	deed	revel
	Chi-square									
OT	lord	jesu	king	israel	son	land	christ	peopl	offer	muhammad
NT	jesu	christ	thing	discipl	lord	faith	receiv	peter	paul	spirit
Quran	god	peopl	lord	muhammed	messeng	believ	torment	day	deed	revel

From the table above, we could see that top 10 ranking words are mostly similar in two different scoring ways except their ranking position, some words like peter and paul are likely name in NT. Also, I found interesting that lord ranks very high in all corpuses, then I looked up them in the file and found that "your lord" which is a coreference appears a lot in all corpuses. From the ranking above, I could learn that OT is mainly about Jesus Christ, Israel, and Muhammad, NT is mainly about Jesus Christ, Israel, and Jesus' faith and disciplines, Quran is mainly about Muhammed, God, believe, and torment.

4. Token Analysis

Here is the table for top 10 tokens and their probability scores for each of the 3 topics.

OT	topic_id: 16, score: 0.08146145921744077 0.061*"midst" + 0.060*"sin" + 0.032*"lord" + 0.029*"transgress" + 0.029*"bread" + 0.028*"break" + 0.028*"commit" + 0.027*"prophesi" + 0.025*"sight" + 0.023*"burnt"
NT	topic_id: 2, score: 0.08956440765293915 0.093*"earth" + 0.058*"heaven" + 0.055*"lord" + 0.053*"good" + 0.040*"god" + 0.032*"kingdom" + 0.031*"east" + 0.026*"full" + 0.026*"serv" + 0.024*"trust"
Quran	topic_id: 15, score: 0.07992562292933598 0.055*"holi" + 0.051*"speak" + 0.051*"burn" + 0.045*"spirit" + 0.044*"lord" + 0.039*"god" + 0.032*"wisdom" + 0.026*"shame" + 0.022*"wise" + 0.022*"fire"

By following the table above, we know that OT talks about transgress commit and sin, NT talks about earth and heaven, Quran talks about Holy spirit. After I printed out the top five topic for each corpus, I found that topic 15 appears in all three corpuses. Topic 7 has present in OT and NT but not appears in Quran, some words like face (292 times in OT, 90 times in NT, 60 times in Quran), high (269 times in OT, 76 times in NT, 30 times in Quran), end (187 times in OT, 85 times in NT, 36 times in Quran) are more frequently showed in NT and OT but not Quran. LDA model tells us a set of topics that are likely to have generated that collection of words. By comparing to MI and Chi-square, a word can have a very high score in MI and chi-square, but it may not appear in the LDA topic tokens because they appear equally likely in all the three corporas.

5. Classification

In order to train the baseline model on SVM, I split the train.txt dataset into 80% training set and 20% development set with shuffle. Then, I used convert_to_bow_matrix method to convert trainset to BOW and train the SVM with C=1000. After training is finished, the macro f1 score reaches 0.99883 in development set, macro precision score reaches 0.99897, macro recall score reaches 0.99868. Then I printed out the three instances from the development set that the baseline system labels incorrect, and are shown below:

Row ['realdonaldtrump', 'daniellemuscato', 'control', 'rapeincest', '3', 'you', 'think', 'the', 'death', 'penalty', 'for', 'a', 'murderer', 'is', 'equivalent', 'to', 'the', 'slaughter', 'of'] has been classified as negative and should be neutral
Row ['wooooo', 'josh', 'hamilton', 'just', 'made', 'tomorrow', 'nights', 'dinner', 'plans', 'half', 'off', 'papajohns', 'rangers', 'walk', 'off', 'single', 'in', '76', 'win', 'over', 'yankees'] has been classified as neutral and should be positive
Row ['tell', 'it', 'sistah', 'elizabethwarren', 'trumptransitionteam', 'draintheswamp'] has been classified as neutral and should be positive

There is a very interesting phenomena that most incorrect contexts are be labeled as neutral, then I found that in the training data, there is 8790 neutral text, 5983 positive text, and 3885 negative texts; there are more neutral text than the others, so the classifier knows more features about neutral text and tends to classify more text into neutral label. Also, I found that these three texts have two labels and when they are separated into training and development set, it will cause incorrected label. The best way to solve this phenomenon is to get more data for positive and negative texts, so I decided to 1. Removing the stop words; 2. Having a lower or higher C to train SVM. 3. Use logistic regression. Here is the result, I used macro f1 score as standard to compare each method.

	Baseline	No Stopwords	C=10	C=6000	Logistic Regression
Development Set	Macro f1=0.581934	Macro f1=0.551688	Macro f1=0.592533	Macro f1=0.590905	Macro f1=0.599115
Test Set	Macro f1=0.591888	Macro f1=0.217204	Macro f1=0.594858	Macro f1=0.583119	Macro f1=0.604991

From the table above, C=10 (improve 0.010599 on development set and 0.00297 on test set) and Logistic Regression (improve 0.017181 on development set and 0.013103 on test set) performance better than the baseline model. Lower C performance better is because the parameter C is a regularization parameter which controls the trade-off between achieving a low error on training data and minimizing the norms of the weights, if we have larger C, it will have greater punishment for wrong classification and generalization ability reduced and have lower score in test set. Notice that without stop words have a significant decrease in test set, that may because in our stop words file, it contains words like "not" which is essential in sentiment analyzing and should not be deleted. And I also tried another classification model: logistic regression and it improves the most among all these method.