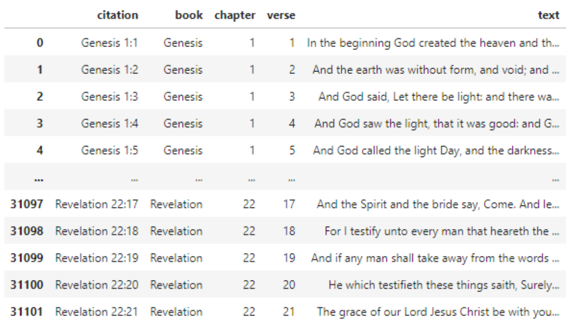
SI650 Final Project: Worship Songs Search Engine

Fangzhe Li
University of Michigan
School of Information / Ann Arbor, MI
`fangzhel@umich.edu`

1 Introduction

Worshiping is an important part in every christian's life and singing worship songs together is more and more becoming a routine in the church life. However, there are a lot of questions the worship leaders need to think of when finding worship songs: if the theme of the song fits in the vibration of that day, if the song has scripture reference, if the song is aligned with the sermon of that day, etc. In this project, a search engine based on scripture references is built, targeting on the church or worship leaders who focus on having worship songs with proper scripture references. The common practice in the church in North America to picking worship songs is still based on the expert knowledge of the worship leaders, which requires them to have a good knowledge on the well-known worship songs and to keep update with new-released albums by different worship ministries. There are some churches using some databases to maintain and record these songs so they can have a plenty of choices. There are some worships search engines that providing theme searching and popular song searching but they usually don't provide the option that people input a verse and output songs related to that verse. In this project, a search engine is built to provide queries based on King James Version of the Bible to give back related worship songs to that verse and similarities generated by a sentence transformer model is used to improve the performance of BM25. The main contribution of this project includes: 1) created a cleaned worship song dataset with the scripture references by web scraping. 2) explored how a pre-trained embedding without fine-tuned with bi-encoder model will help improve the performance of learning to rank model.



	citation	book	chapter	verse	text
0	Genesis 1:1	Genesis	1	1	In the beginning God created the heaven and th...
1	Genesis 1:2	Genesis	1	2	And the earth was without form, and void; and ...
2	Genesis 1:3	Genesis	1	3	And God said, Let there be light: and there wa...
3	Genesis 1:4	Genesis	1	4	And God saw the light, that it was good: and G...
4	Genesis 1:5	Genesis	1	5	And God called the light Day, and the darkness...
...	...	...	...	...	...
31097	Revelation 22:17	Revelation	22	17	And the Spirit and the bride say, Come. And le...
31098	Revelation 22:18	Revelation	22	18	For I testify unto every man that heareth the ...
31099	Revelation 22:19	Revelation	22	19	And if any man shall take away from the words ...
31100	Revelation 22:20	Revelation	22	20	He which testifieth these things saith, Surely...
31101	Revelation 22:21	Revelation	22	21	The grace of our Lord Jesus Christ be with you...

Figure 1: Sample annotated bible data

2 Data

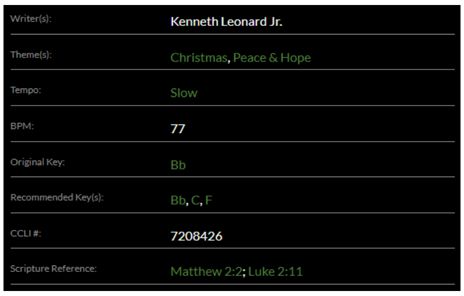
2.1 Bible Dataset

The data source for this project is King James verison bible and some annotated cross reference from openbible.info. The King James version bible dataset contains four columns: Book, Chapter, Verse and Text.

1. Book: The book name of the Bible verse
2. Chapter: The chapter number of the Bible verse
3. Verse: The verse number of the Bible verse
4. Text: The text content of the Bible verse

2.2 Worship Songs Data Source

A large portion of efforts on this project is spent on how to scrape data, clean it up and design a way to annotate the data automatically. The data source is https://www.worshiptogether.com/songs. They provide following information: the themes of worship songs, the tempos and recommended keys, and the verses it refers to in these songs, which are used to generate relevance score later. And after cleaning up all the data, a dataset with 1982 songs are collected. Also, the ground truth of 1711



Writer(s):	Kenneth Leonard Jr.
Theme(s):	Christmas, Peace & Hope
Tempo:	Slow
BPM:	77
Original Key:	Bb
Recommended Key(s):	Bb, C, F
CCLI #:	7208426
Scripture Reference:	Matthew 2:2; Luke 2:11

Figure 2: Raw data from worshiptogether

3	0.001
2	0.132
1	0.439
0	0.428

Table 1: Distribution of Relevance Scores

queries to this dataset is also built according to the annotation method which will be illustrated later. The scale of 0 - 3 is used and the distribution of relevance scores is shown in the table 1:

3 Related Work

According to my research, there are still a lot of churches that don't have a systematical regulation on worship songs to use but their use of worship songs highly according to individuals like elders, pastors, etc. Pastors and elders in the churches who organize worship events like Sunday celebrations, Worship gatherings will be the main targets to benefit from this project. The Paper for BERT (Devlin et al., 2018) is quite helpful since it gives me an idea on the task BERT is trained for to help me get to know how to fine-tune it properly. This paper (Wang et al., 2020) discusses how this 12 layer model is trained and how it performs, a simplified version of it is used in this project. This paper (Ingalls, 2017) discusses the different between hymns and Contemporary Worship Music and gives me hint on how to use correct version of the Bible. This paper (J"arvelin and Kek"al"ainen, 2002) discussed how the ndcg is used and thought as a good indicator for our task of worship song search engine. This paper (Papineni et al., 2002) other criteria that can be used to evaluate the quality of two aligned text.

4 Methodology

4.1 Data Manipulation

4.1.1 Web Scraping

Since all the worship songs information are loaded by JavaScript, so I used Python Selenium with a headless Chrome browser to get all data. The list of links to songs are first scraped and stored in song_urls_list.json. Then I get the information from of each song using Python library Beautiful Soup to parse the page and have all the information stored in a file called song.csv.

4.1.2 Data preprocessing

The missing values and duplicate rows are dropped and the references of the Bible is unified by using regular expression. All the special cases are dealt separately and all the punctuation in the query are removed to make sure next step will be good to go.

4.1.3 Construct relevance score

To build the relevance score for 3391202 rows, it is not practical to annotate it one by one by hand. So a method of annotation generation comes out. Given the query is a verse, if the verse is listed in the scripture references for a song (document), it means direct related, so the relevance score will be 3. As for partial related queries and documents, we build the inverted index between queries and themes, and use voting to decide the themes of a verse (query). Then if the set of themes of query has large than or equal to 2 intersections with the set of themes of document, they are more related to each other and the relevance score will be 2. And if the set of themes of query has 1 intersection with the set of themes of document, it means that they are somewhat related so the relevance score will be 1. Otherwise it will be 0, which means totally irrelevant. Due to the limitation of computing capacity, especially because of the deep learning part will be used later, only 20 queries are picked to be use as the qrels. And the 20 queries selected are from different genres of the Bible to make sure we can evaluate it in different cases. Query includes: Psalms like Psalm 89:11, Epistles like Colossians 2:13 and Gospels like Matthew 16:24, which varies among different genres of the Bible.

4.2 Model

The models used here include: TF-IDF, BM25 and three learning to rank models: random forest, fastrank and lightgbm. Also, based on these methods, a sentence transformer is used to generate cosine similarity between embeddings of queries and documents. The Python library used

1. sklearn.ensemble.RandomForestRegressor
2. fastrank
3. lightgbm

The sentence transformer sentence-transformers/all-MiniLM-L6-v2 is pre-trained on nreimers/MiniLM-L6-H384-uncased model,which is a model by keeping only every second layer of (Wang et al., 2020) and fine-tuned by 1 billion sentence pairs dataset. This embedding maps sentences and paragraphs to a 384 dimensional vector space and can be used in downstream tasks. The sentence vectors are trained especially for our task of getting sentence similarity.

Since not very good sentence pair are available or can be constructed on this dataset, fine-tuning is not done on this sentence transformer. But the pre-trained one should have enough general English knowledge to deal with our case.

4.3 Features

In learning to rank, I used the following as features:

1. TF-IDF
2. BM25
3. cosine similarity between query and document through sentence transformer

The first two are very common practice in learning to rank and cosine similarity will add information to our reranking through the attention mechanism used in the deep learning model.

5 Evaluation and Results

Since we have 1982 documents and 20 queries, our qrels has 39640 rows. The training, valid and test queries are split as 14, 2, 4 separately.

6 Baseline

The baseline is TF-IDF and BM25, the result is shown in Table 2:

The performance of TF-IDF is better than BM25 in both map and ndcg.

name	map	ndcg
bm25	0.297717	0.452061
tf_idf	0.304127	0.456555

Table 2: Baseline Performance

name	map	ndcg	ndcg_cut_10	mrt
bm25	0.297717	0.452061	0.340589	26.538131
tf_idf	0.304127	0.456555	0.339237	27.655761
random_forest	0.322511	0.464732	0.419328	267.318306
fast_rank	0.304970	0.457099	0.360435	127.533824
lightgbm	0.305806	0.455761	0.293088	126.613475

Table 3: Learning to Rank Performance

7 Learning to Rank

Then the technique of learning to rank is used, and BM25, TF-IDF are used as the features and the result is in Table 3: We can see that the random forest model does a good job here in both map and ndcg and it outperforms a lot than our two baseline in ndcg@10, which is very important indicator in our worships songs retrieval system. It seems that the ensemble of BM25 and TF-IDF does improve our performance.

8 Learning to Rank with Similarity

The technique of learning to rank is still used, and besides BM25, TF-IDF, the similarity between songs' lyric and verses are added and the result is in Table 4:

It is not hard to see that in most case the ndcg is very close to the version without using similarity and there is a large drop in ndcg@10 for our best model till now random forest, which means adding the similarity doesn't help our search engine but hurt its performance in many way. Also the response time almost double, which is bad for user experience of our search engine.

name	map	ndcg	ndcg_cut_10	mrt
bm25	0.297717	0.452061	0.340589	25.382276
tf_idf	0.304127	0.456555	0.339237	29.706594
random_forest	0.313845	0.459670	0.340272	452.797800
fast_rank	0.306321	0.457816	0.333805	322.522716
lightgbm	0.305806	0.455761	0.293088	129.802805

Table 4: Learning to Rank with Similarity Performance

9 Discussion

From the above evaluation, it is seen that the random forest is the best model in our context by ensembling the features of bm25 and tf-idf. Random Forest really excels itself in his performance on ndcg@10. The reason ndcg@10 is important is as follows: Since the dataset gets 1982 contemporary worship songs, which is a large portion of them, even though we also take hymns into consideration, it can't reach more than 5000 songs. This number is too few for 31, 102 verses in the bible. This means for a lot of verses you can't find a corresponding song to them. And even we take the fact that there are some more well-known verses and they have more songs about them. The direct related songs will be very few, that's why ndcg@10 is very important. Also we notice that using the similarity doesn't help our performance that much or even hurts our model performance. It is not surprising that the response time is largely increased since we use the deep learning model. And the main reason for decreasing in ndcg@10 may be that the training data is not big enough for the similarity to function. Since the more queries are ready, once computing capacity is available, it is easy to explore how similarity performs. Another reason may be that the sentence transformer is not fine-tuned on Bible context, a better way may be using the bi-encoder structure to fine-tune it before put it into the search engine. Also, for the auto-annotated relevance score, I checked it by myself and think it is quite legit and I tried to fine-tune the number of intersection/ add bar for themes and it doesn't change the result that much. The version of the Bible can also be a potential problem because King James version's English is more proper to search Hymns but for Contemporary Worship Music, probably it will be better to use American Standard Version of the Bible.

10 Conclusion

Building a worship song search engine is a challenging and interesting topic. In this report, we constructed the largest worship song reference dataset available to public and proposed a feasible method on annotating the relevance score on lyrics and references through annotated theme information. We also built our search engine based on the random forest using the features of TF-IDF and BM25, which outperforms TF-IDF and BM25, our baseline. Even though the new feature similarity generated by the embeddings of lyrics and verses struggled with improving our performance, there are still a lot of room for us to explore and improve.

11 Other Things We Tried

A lot of efforts are spent on trying finding different sources for worships songs with proper sculpture references but it is a pleasure to finally clean it out and make it largest dataset available to public on worship songs and their references. Also some time has been spent on trying to make the code from assignment 4 works but it still not perform properly so the model is switched to a not fine-tuned sentence transformer.

12 What You Would Have Done Differently or Next

As discussed in the Discussion part, I think it will be possible to build some sentence pair from the annotated data we have between lyrics and references to fine-tune the sentence transformer or tried other bi-encoder models to improve the performance of the similarity part. Also, with more computing capacity available, we can fully make use of all the 1711 queries we have. Lastly, we can also create more queries that don't have a song related with them and manually annotate them to test the performance of model. Lastly, we can change the version of Bible from King James version to American Standard version to increase the closeness of the language style.

13 Acknowledgments

Here I want to thank for pastors and friends from Harvest Mission Community Church and Ann Arbor Chinese Christian Church's support on gathering information on how the worship is organized.

References

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding.

Monique M. Ingalls. 2017. Style matters: Contemporary worship music and the meaning of popular musical borrowings. *Liturgy*, 32(1):7–15.

Kalervo J"arvelin and Jaana Kek"al"ainen. 2002. Cumulated gain-based evaluation of ir techniques. *ACM Trans. Inf. Syst.*, 20(4):422–446.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: A method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*, ACL '02, page 311–318, USA. Association for Computational Linguistics.

Wenhui Wang, Furu Wei, Li Dong, Hangbo Bao, Nan Yang, and Ming Zhou. 2020. Minilm: Deep self-attention distillation for task-agnostic compression of pre-trained transformers.