Pretrained KR-SBERT-based Course Recommendation

Jungwoo Park, Chanwoo Park, San Kim, Hyeonseo Cho
Department of Computer Science, Hanyang University

Abstract

This paper introduces a user-friendly recommendation system for liberal arts courses, leveraging the SBERT model. The system, developed through HTML data collection and preprocessing, focuses on enhancing the course selection process. Utilizing the KR-SBERT pre-trained model, the system is trained with a triplet dataset and rigorously tested using cosine similarity. Evaluation metrics such as acc, hitsAt3, hitsAt5, hitsAt10, and rankingBasedMetric provide a thorough assessment of the system's performance. Findings indicate the system's potential to improve students' university experiences, with insights on training epochs, keyword effectiveness, and ranking criteria uncertainty. These findings highlight opportunities for further refinement.

1 Introduction

In many universities, students can take a variety of liberal arts courses to learn about different subjects beyond their major fields of study. However, the abundance of available options make it difficult for students to select courses that align with their interests. Traditional methods involve checking course catalogs or websites with reviews, but the overwhelming volume of information makes it challenging to make best decisions.

To address this issue, this paper introduces a user-friendly recommendation system for liberal arts courses, specifically using the SBERT(Reimers and Gurevych, 2019) model. The system trains the model with course reviews data extracted from websites and suggests courses based on user-provided reviews in a similar format. The goal is to offer personalized and effective recommendations, simplifying the course selection process.

The paper is structured as follows: Section 2 reviews relevant literature, including prior studies on SBERT and related techniques. Section 3 explores methods for extracting course reviews from websites to build a robust dataset. Section 4 details the process of training the model using the collected data. Sections 5 and 6 present experiments to evaluate the model's performance through various metrics and discuss the obtained results. Finally, Section 7 analyzes the outcomes and draws conclusions.

2 Prior Related Works

SBERT (Sentence Embeddings based on Transformers)[1] is a technique used to mathematically represent sentences. It utilizes Transformer models to convert sentences into numerical vectors, preserving their meaning and structure.

This technology relies on the Transformer architecture and a specific fine-tuning method known as Sentence-BERT. Through this approach, SBERT can effectively capture the similarity between sentences and proves valuable in various natural language processing tasks. For instance, SBERT can be used to find similar sentences or compute the similarity between them.

SBERT holds significant promise in the field of natural language understanding, enabling efficient processing and analysis of text data. Due to its capabilities, SBERT finds applications across a wide range of natural language processing tasks and applications.

3 Data

The process of preparing the dataset for model training is as follows. First, HTML files containing course reviews for each lecture are collected from websites. Next, the desired content is extracted from the collected files and refined into the preferred format.

[1] https://www.sbert.net/

Notation	Area
CLA	Reading Classics
ETC	General
FUT	Future Industries & Entrepreneurship
GLO	Global Language and Culture
LIT	Humanities and Art
SCI	Science and Technology
SOC	Society and The World
SOF	Software
VIR	Virtual University

Table 1: Notation for the Areas of Liberal Arts Courses at Hanyang University.

3.1 Collecting HTML data

The format of the collected course reviews is predominantly in Korean, and for ease of collection, liberal arts courses from Hanyang University offered on the 'Everytime[2]'website were selected as the data source. To ensure a diverse dataset for performance comparison, data were collected independently for each liberal arts area. Table 1 indicates the notation for each area of liberal arts courses at Hanyang University. Each course's review webpage was saved as an individual HTML file which includes the course title, professor's name, and multiple reviews.

3.2 Preprocessing data

Using 'BeautifulSoup4[3]', necessary information is extracted from the collected HTML course review files. The course title and professor's name are concatenated into a single label, which is then paired with the reviews for that course. The aggregated data is randomly divided into separate files for training and testing, maintaining a consistent ratio, which is set by manually as a hyperparameter.

4 Model

The model for the course recommendation system is structured as follows. Firstly, a pretrained model is loaded. Next, a triplet(Schroff et al., 2015) dataset for training is prepared. Using this dataset, the model is trained with appropriate parameters, and the results are tested.

4.1 Loading pre-trained model

Given that the dataset for model training primarily consists of Korean lecture reviews, the KR-SBERT(Lee et al., 2020), a Korean pre-trained model, is utilized. The choice of this model over others is based on its suitability and superior performance for the task at hand. KoBERT[4] showed lower performance, KoNLPy[5] is designed as a model predicting the next word and KoBART-sum[6] is oriented towards summarization tasks. Thus, these models were not appropriate for the task. Unlike them, KR-SBERT is well-suited for the task of course recommendation.

4.2 Preparing train dataset (triplets)

The model's training data is structured in triplet form. From the preprocessed train dataset, two reviews for the same course and one review for a different course are combined to form a triplet. This approach allows for generating numerous combinations with a minimal amount of data, and the quantity can be adjusted accordingly. The three reviews in the triplet correspond to anchor, positive, and negative, where the model is trained to bring the anchor closer to the positive while distancing it from the negative.

4.3 Training

The model is trained using the prepared triplet training dataset. TripletLoss[7] is employed as the loss function, and after training with appropriately adjusted hyperparameters, the trained model is saved.

4.4 Testing

The trained model is loaded, and testing is conducted to assess its performance. Firstly, a review from the test dataset is selected as the query. Next, for each course, the vectors of the reviews are averaged to create an average vector for that lecture. The cosine similarity between vectors is calculated to identify lectures with vectors closest to the query, and these are ranked accordingly. This process is repeated for each review in the test dataset, comparing the results with the actual correct labels and evaluating them using various metrics.

5 Experiments

Evaluation metrics for the model include the following: acc, representing the proportion of correctly predicted recommendations; hitsAt3, hitsAt5, and hitsAt10, representing the proportions of correct

[2]https://everytime.kr/
[3]https://pypi.org/project/beautifulsoup4/
[4]https://github.com/SKTBrain/KoBERT
[5]https://konlpy.org/ko/latest/
[6]https://github.com/seujung/KoBART-summarization
[7]https://www.sbert.net/docs/package_reference/losses.html

recommendations within the top 3, 5, and 10 suggestions, respectively; and rankingBasedMetric, a metric that assigns scores based on the ranking of the actual correct answer from the predicted recommendations.

5.1 Data Preparation

Prepare a test dataset containing pairs of queries and actual correct recommendations (ground truth). For each query, obtain the model's predicted recommendations.

5.2 Prediction and Vectorization

For each query, follow the process described earlier to obtain the average vector representation for the recommendations. Calculate the cosine similarity between the query vector and each recommendation vector.

5.3 Ranking

Rank the recommendations based on their cosine similarity scores in descending order.

5.4 Metric Computation

- Accuracy (acc)

$$Accuracy = \frac{\#\ of\ True\ Predictions}{\#\ of\ Total\ Predictions}$$

- Hits@K (where K is 3, 5, or 10)

$$Hits@K = \frac{\#\ of\ Hit\ Predictions}{\#\ of\ Total\ Predictions}$$

- Ranking-Based Metric
  The prediction of the 'correct course review' as the top-ranked among L predictions is assigned a score of 1 point, predicting it as the second-ranked results in a score of $1 - \frac{1}{L}$, predicting it as the third-ranked results in a score of $1 - \frac{2}{L}$, and so forth. Subsequently, these scores are calculated and divided by the total number of predictions.

$$RBM = \frac{1}{N}\sum_{i=1}^{N}(1 - \frac{rank_i - 1}{\#\ of\ Lectures})$$

5.5 Experiment Execution

Iterate through each query in the test dataset, following steps 2-4. Accumulate the metrics (accuracy, hits@3, hits@5, hits@10, ranking-based metric) for the entire test dataset.

5.6 Analysis and Reporting

Calculate the average values of the metrics over the entire test dataset. Analyze the results to understand the model's performance in terms of correct predictions, recommendation rankings, and other specified metrics.

6 Results

We obtained results by training for each course area using a batch size of 16 and Adam optimizer with a learning rate of 2e-5, as depicted in Table 2 through 3.

7 Analysis and Conclusion

We conclude that our recommendation system has the potential to greatly contribute to students' university experiences by aiding them in discovering

dataset	epochs	acc	hits@3	hits@5	hits@10	RankingBasedMetric
CLA	64	0.864	0.910	0.923	0.940	0.964
	96	0.867	0.912	0.934	0.961	0.974
	128	0.866	0.906	0.924	0.953	0.969
FUT	64	0.840	0.914	0.938	0.968	0.981
	96	0.821	0.913	0.925	0.955	0.977
	128	0.834	0.917	0.931	0.964	0.981
GLO	64	0.821	0.919	0.943	0.961	0.979
	96	0.829	0.919	0.935	0.961	0.982
	128	0.834	0.920	0.941	0.957	0.980
LIT	64	0.848	0.921	0.938	0.970	0.984
	96	0.864	0.920	0.941	0.960	0.984
	128	0.868	0.938	0.955	0.964	0.986

Table 2: Trained result for CLA, FUT, GLO, and LIT dataset.

dataset	epochs	acc	hits@3	hits@5	hits@10	RankingBasedMetric
ETC		0.938	0.963	0.972	0.986	0.985
SCI		0.872	0.916	0.934	0.966	0.973
SOC	128	0.894	0.941	0.958	0.970	0.985
SOF		0.918	0.971	0.976	1.000	0.971
VIR		0.870	0.916	0.951	0.979	0.972

Table 3: Trained result for other dataset.

the desired general education courses for a more enriching academic journey.

In addition to this, our interdisciplinary course recommendation machine learning pipeline yielded several key findings. Firstly, increasing the number of training epochs significantly improved performance. Secondly, leveraging specific keywords in course reviews proved effective, although handling ambiguous expressions posed challenges. Lastly, there remains uncertainty regarding whether prioritizing the average cosine similarity order or factoring in the number of reviews is more optimal.

References

Sangah Lee, Hansol Jang, Yunmee Baik, Suzi Park, and Hyopil Shin. 2020. Kr-bert: A small-scale korean-specific language model. *arXiv preprint arXiv:2008.03979*.

Nils Reimers and Iryna Gurevych. 2019. Sentence-bert: Sentence embeddings using siamese bert-networks. *arXiv preprint arXiv:1908.10084*.

Florian Schroff, Dmitry Kalenichenko, and James Philbin. 2015. Facenet: A unified embedding for face recognition and clustering. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 815–823.