# Text mining final project

# Context-based movie search for user questions that ask the title of the movie

2018. 4. 18

장진규, 박정인

UNIST School of Business Administration

## Contents

## Contents

# Introduction

**People sometimes have a craving for find a movie that they once glimpsed. At that time, they used to ask the movie name through Q&A sites and get the result. Answerers often seems 'god of movie', so we want to imitate their prophecy.**

## Question Examples

영화 제목 좀 찾아주세요 해외영화였던...

비공개 질문 7건 | 질문마감률 60% | 질문채택률 60% 2018.04.15. 18:20 | 조회수 20

영화 제목 좀 찾아주세요

해외영화였던 거 같고 사람들이 무슨 별장 같은 곳에 갇혀서 어떤 여자가 육상부였다고 문 여는 동시에 뛰어나갔는데 무슨 낚시줄? 같은 투명한 줄 같은 거에 목이 잘렸는데 무슨 영화였는 지 기억이 안 나요

영화 제목 질문합니다 여자가 안약 개...

비공개 질문 18건 | 질문마감률 100% | 질문채택률 100% | 2018.04.15. 11:50 | 조회수 19

영화 제목 질문합니다
여자가 안약 개발한 사람인데 동생은 식물인간이고 그 안약 넣으면 가상세계로 체험 할 수 있고 나중에 공동개발한 사람한테 배신당했다가 복수하는 내용?? 혹시 아시나요?!

# Data Gathering

**We chose one expert of this field and gather his answers.**

**Gathered Data Information**

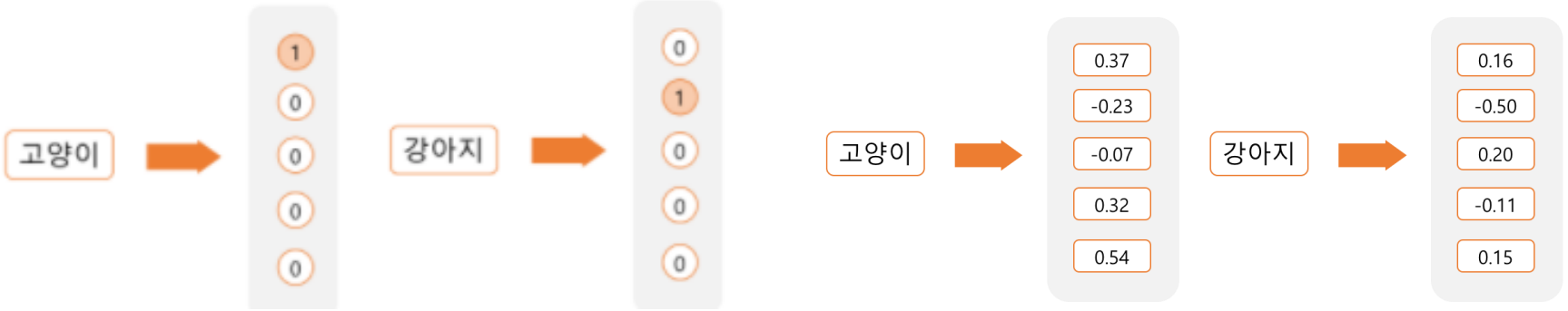
| | |
|---|---|
| **Q&A Site** | http://kin.naver.com |
| **Expert ID** | xedz**** |
| **Question & Answer data** | 39,758 |
| **Date** | 2012 December ~ 2018 March |
| **Unique Movie** | 5,900 |

# 2 Types of Text Representation

## There are 2 kinds of text representation: sparse and dense

### Sparse: One-Hot Encoding

### Dense: Word Embedding




### Comparison of Text Representations

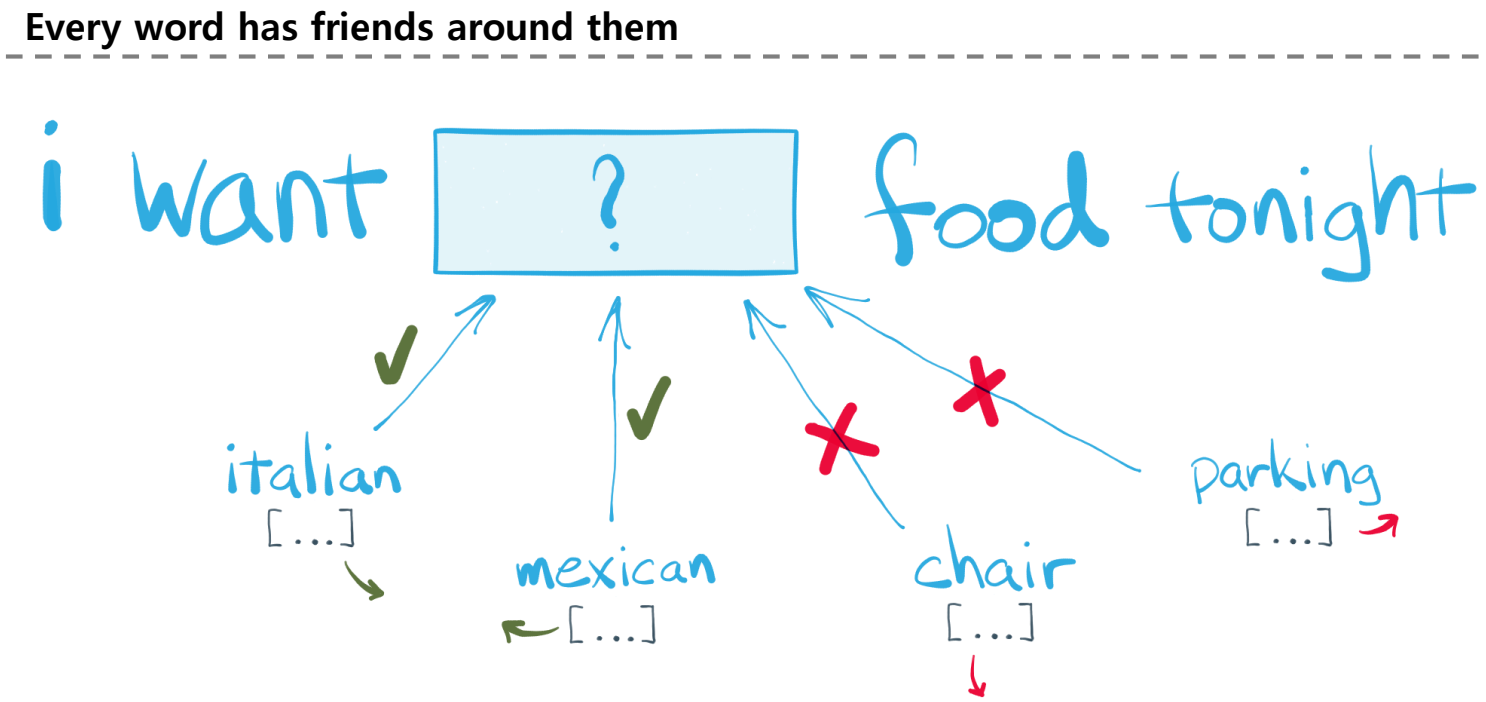
| | Sparse | Dense |
|---|---|---|
| **Dimension** | ▪ As many as unique words | • Autonomous setting<br>• Usually 20~200 dimensions |
| **Information** | • Lots of 0 value<br>• No Information | • Every element has value<br>• Abundant Information |

source: https://dreamgonfly.github.io/machine/learning,/natural/language/processing/2017/08/16/word2vec_explained.html

# Main Idea of Word2Vec

**Word2Vec is one of the word embedding methods.**

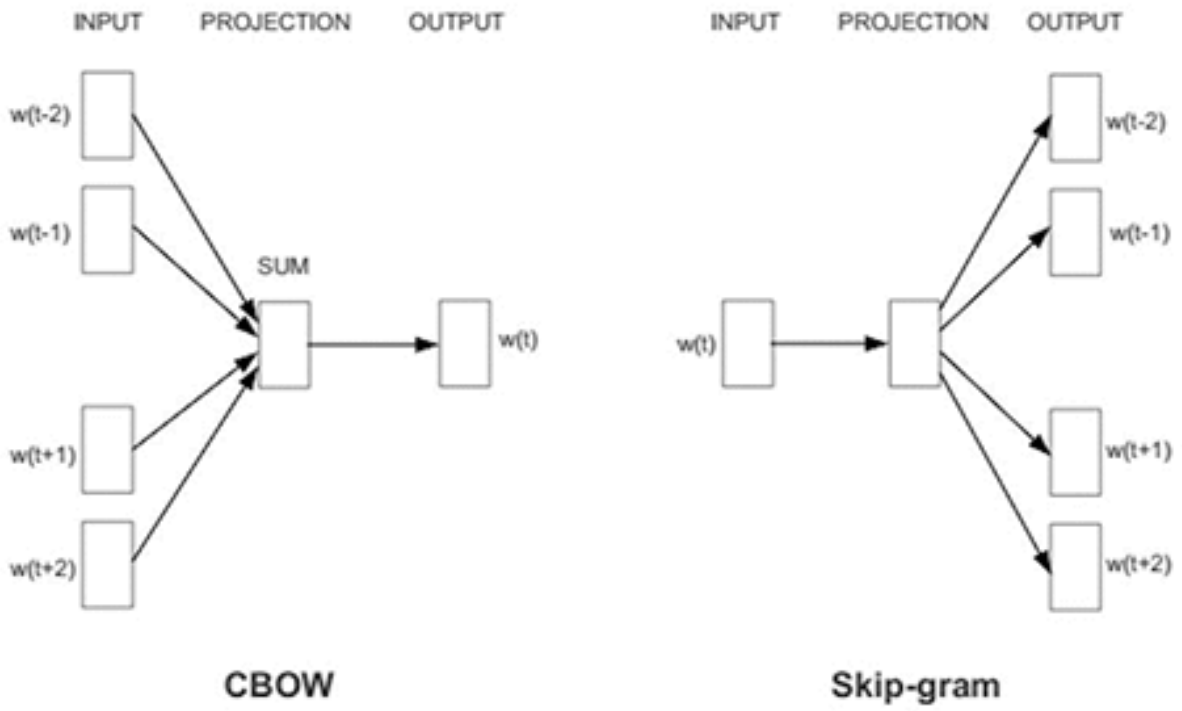
**Its main idea is "You shall know a word by the company it keeps."**



# Algorithms of Word2Vec

**Word2vec has two model architectures: continuous-bag-of-words (CBOW), skip-gram.**

**Diagrams of CBOW and Skip-gram**

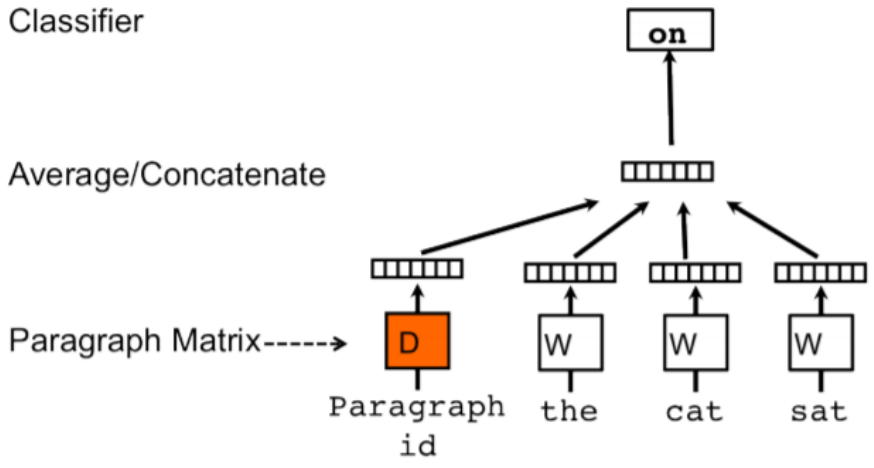



source: https://aws.amazon.com/ko/blogs/korea/amazon-sagemaker-blazingtext-parallelizing-word2vec-on-multiple-cpus-or-gpus/

# Algorithms of Doc2Vec

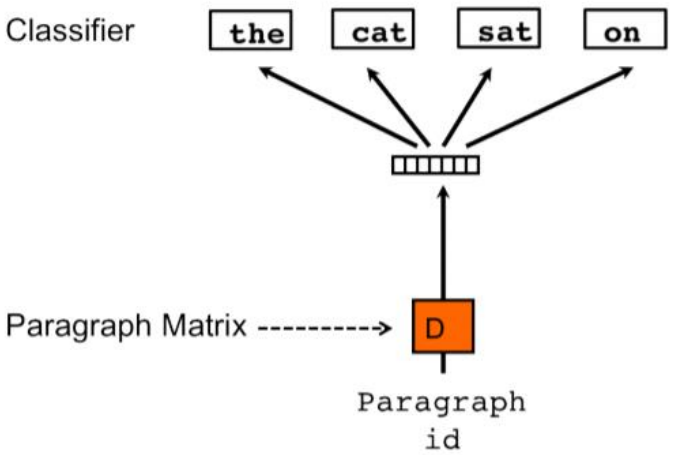
**Doc2vec has two model architectures: distributed memory model (PV-DM) and Distributed bag of words model(PV-DBOW).**

### Diagrams of PV-DM and PV-DBOW




**PV-DM**

The concatenation or average of vector with a context of three words is used to predict the fourth word. The paragraph vector represents the missing information from the current context




**PV-DBOW**

Ignore the context words in the input, but force the model to predict words randomly sampled from the paragraph in the output. Similar to Skip-gram model

source: Distributed Representations of Sentences and Documents

## Contents

# Preprocessing

**We did preprocessing for better performance and it is processed by 2 steps: whole text data and tokenized data.**

## Raw Preprocessing

- **Remove unnecessary words**
  - URL, Special characters (!, ?, *, @, <. >), Emoticon(ㅋㅋ, ㅠㅠ), multispacer
- **Stem words that dictionary cannot correct**
  - (남주 → 남자주인공), (페북 → 페이스북), (영환 → 영화인데), (여자애 → 여자)
- **Delete unnecessory phrase in question**
  - 좀 옛날 영화인데 ~, 페북에서 봤는데, ~ 장면이 있었는데 기억이안나네요
- **Delete questions of which length are less than 30**

## Tokenizing

- **Tokenize with KoNLPy**
  - using Twitter package
- **Pos-tagging**
  - only get noun, verb, and adjective
- **Remove Token which has only one character**
- **Remove Stop-words**
- **Delete questions of which token length are less than 10**

# Select Movies and Split dataset

**There are 5,900 movies in dataset, but many movies has few questions. So we remove certain movies that have questions below cutoff value. Then we split the dataset with 8:2 ratio to test the model.**

## The number of question per movies

| Movie | Count |
|---|---|
| 스파이더위크가의 비밀 | 259 |
| 캐빈 인 더 우즈 | 222 |
| 비밀의 숲 테라비시아 | 179 |

Cutoff

| | |
|---|---|
| 무서운 영화 2 | 1 |
| 전우 | 1 |
| 전우치 | 1 |

## Split Train and Test

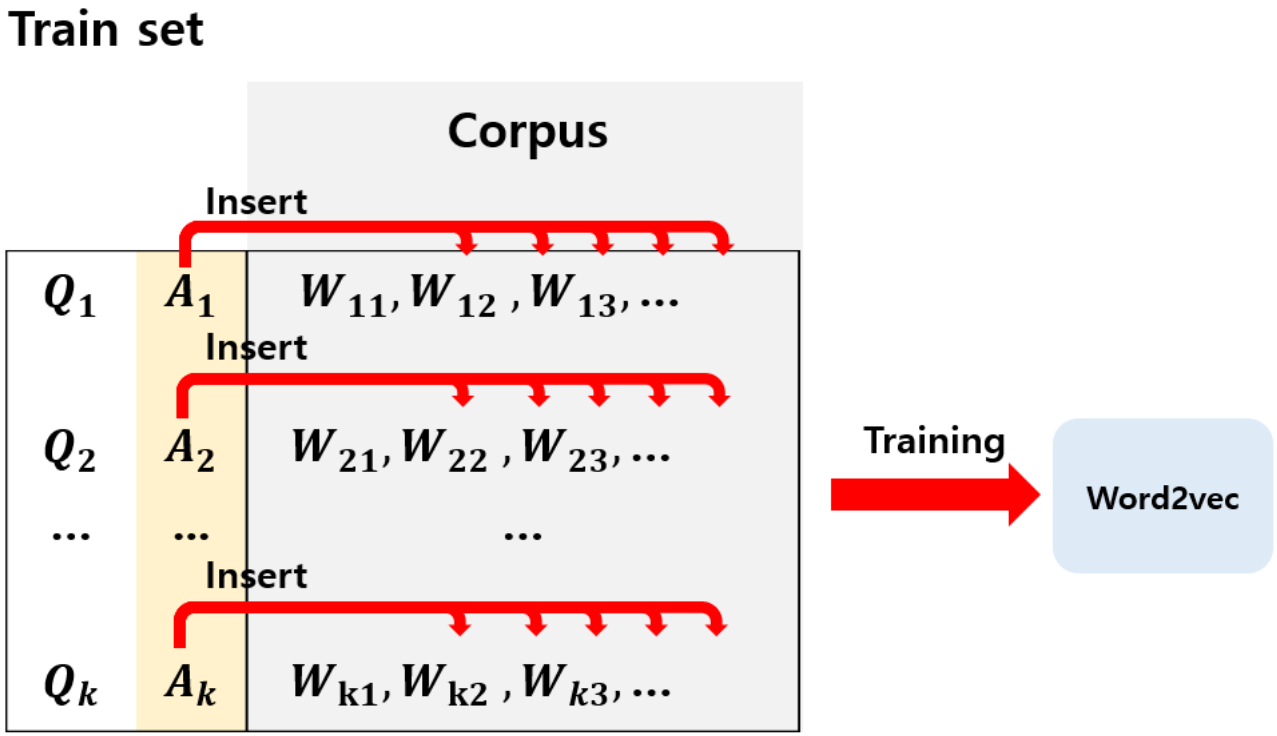
| Movie | Train | Test |
|---|---|---|
| **스파이더위크가의 비밀** | 207 | 52 |
| **케빈 인 더 우즈** | 177 | 45 |
| **비밀의 숲 테라비시아** | 143 | 36 |
| **레모니 스니켓의 위험한 대결** | 142 | 36 |
| **플립** | 141 | 36 |
| … | … | … |

*Basic cutoff = 3
*Using stratified method

## Contents

# Modeling – Word2vec

**To train word2vec model, we put the answers (label) between the tokenized words in the question. Using this corpus, we trained word2vec model.**

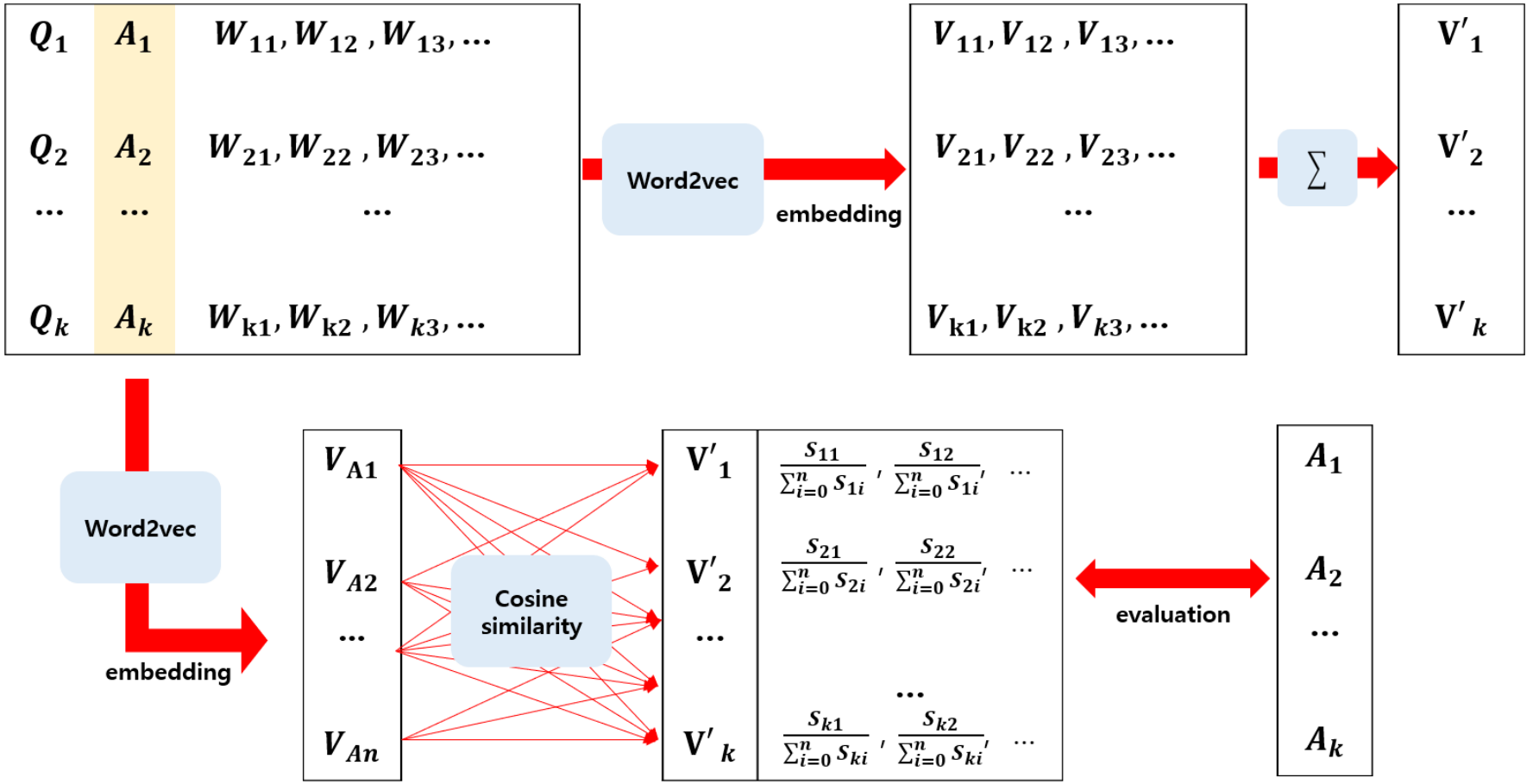



*put labels in every 5 words
Q: question, A: answer(label), W: word
The number of labels in train, test data : 2021
Train data set : 22620 ,Test data set: 5655

skip-gram is employed
Dimensionality of the feature vectors - 300
Window size - 10
Hierarchical softmax used

# Modeling – Word2vec

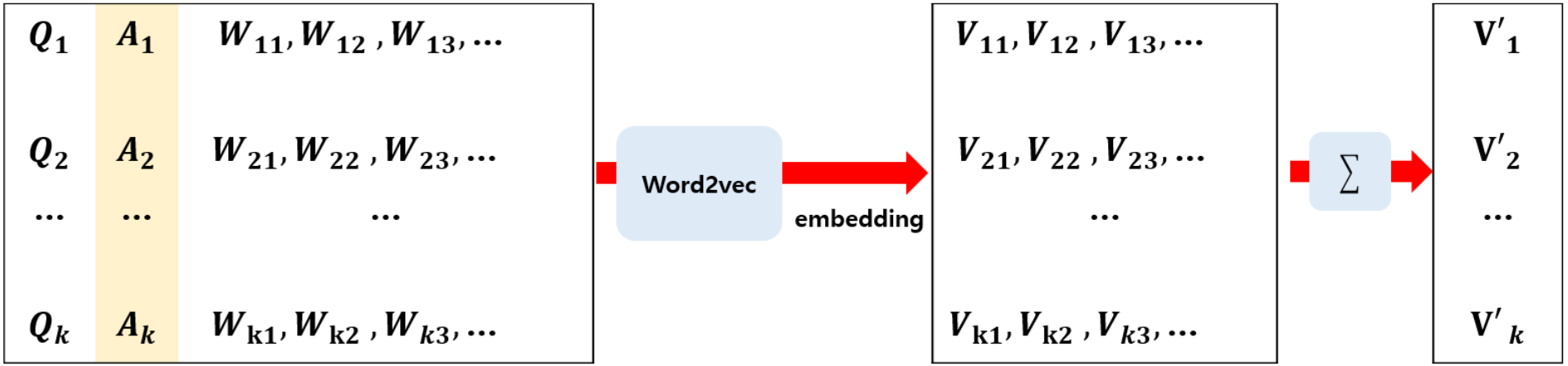
## Test set



# Modeling – Word2vec

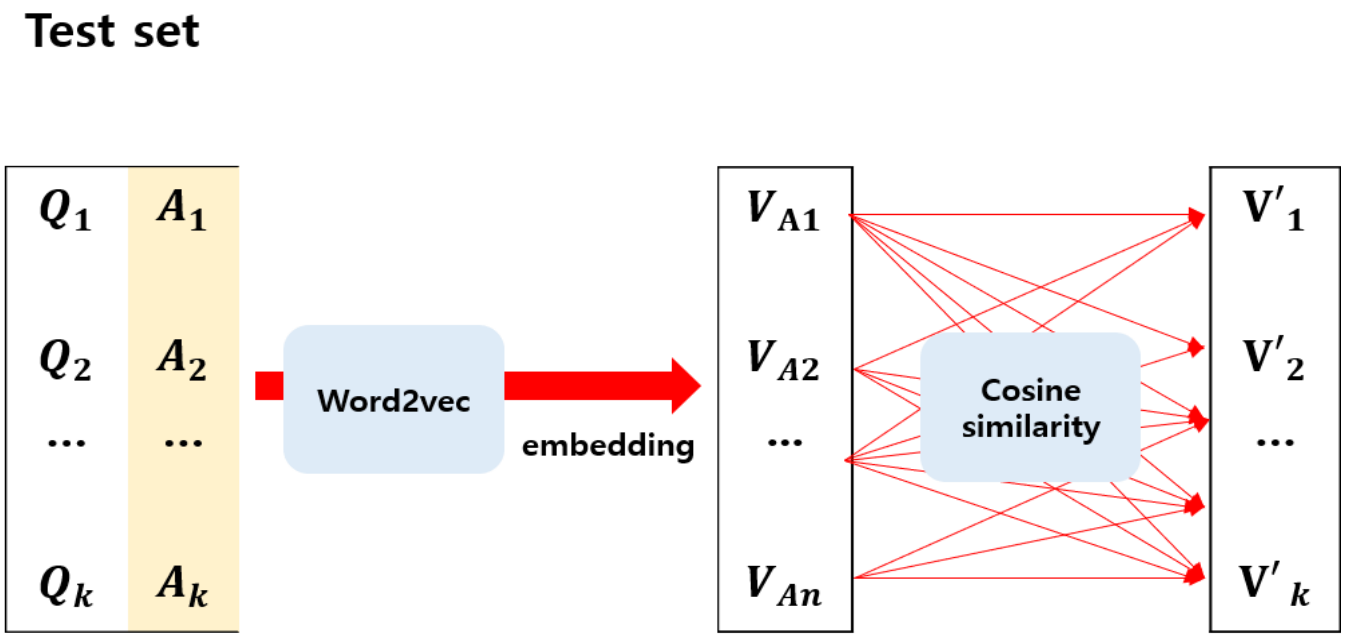
**Each word in the test set is embedded into the model to obtain a word vector.**
**Combine all the vectors into one vector on a question-by-question basis (Document vector)**

Test set



# Modeling – Word2vec

**Also embedding the unique answers (label) into the model to obtain label vector. After that, Calculate pairwise cosine similarity between the label vectors ($V_{An}$ ) and the document vectors ($V'_k$)**

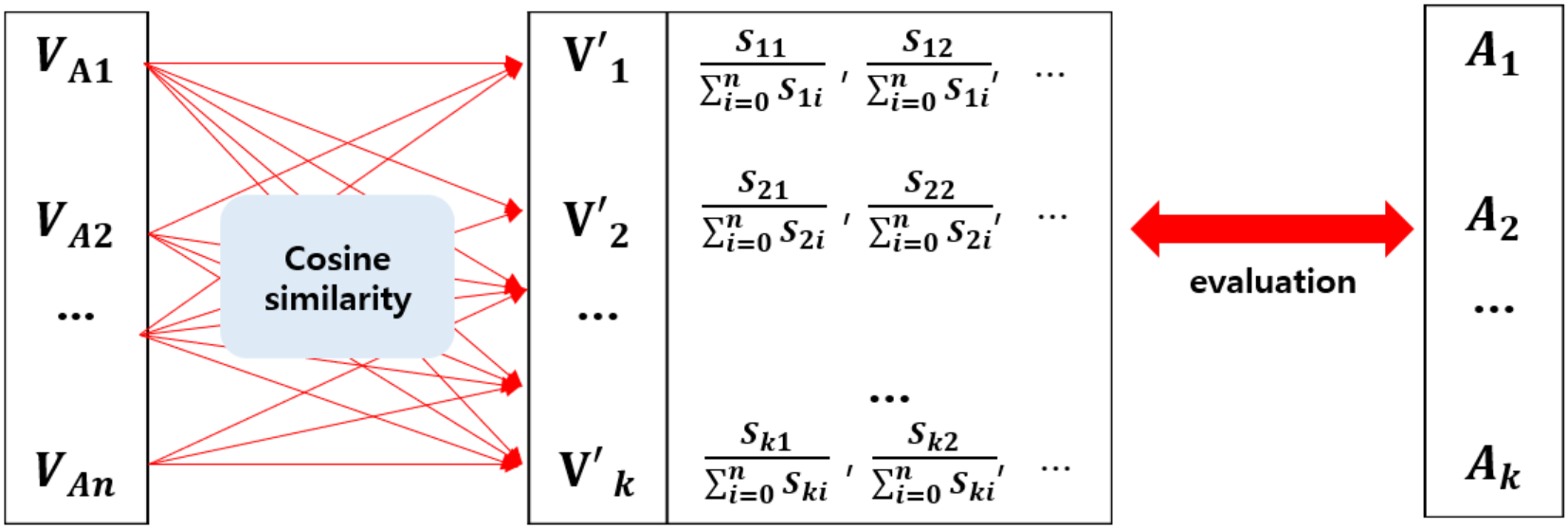



K: the number test set data
n: the number of unique labels

# Modeling – Word2vec

**Finally, Normalize the cosine similarity result for each document vectors, binarize the answers(label), and evaluate the performance of the model**

## Test set




*Ex)*

$V'_k = [0.05, 0.001, 0.003 \ \ldots . \ 0.002]$

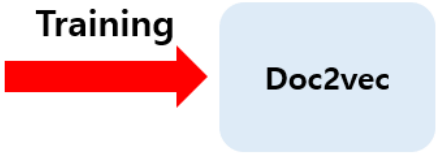
$A_k = [1,0,0 \ \ldots . \ 0]$

# Modeling – Doc2vec

**In the Doc2vec model, we do not need to put the correct answer like in the word2vec model, because the answer (label) is also learned as a tag.**

## Train set

| | Tag | Corpus |
|---|---|---|
| $Q_1$ | $A_1$ | $W_{11}, W_{12}, W_{13}, \dots$ |
| $Q_2$ | $A_2$ | $W_{21}, W_{22}, W_{23}, \dots$ |
| … | … | … |
| $Q_k$ | $A_k$ | $W_{k1}, W_{k2}, W_{k3}, \dots$ |

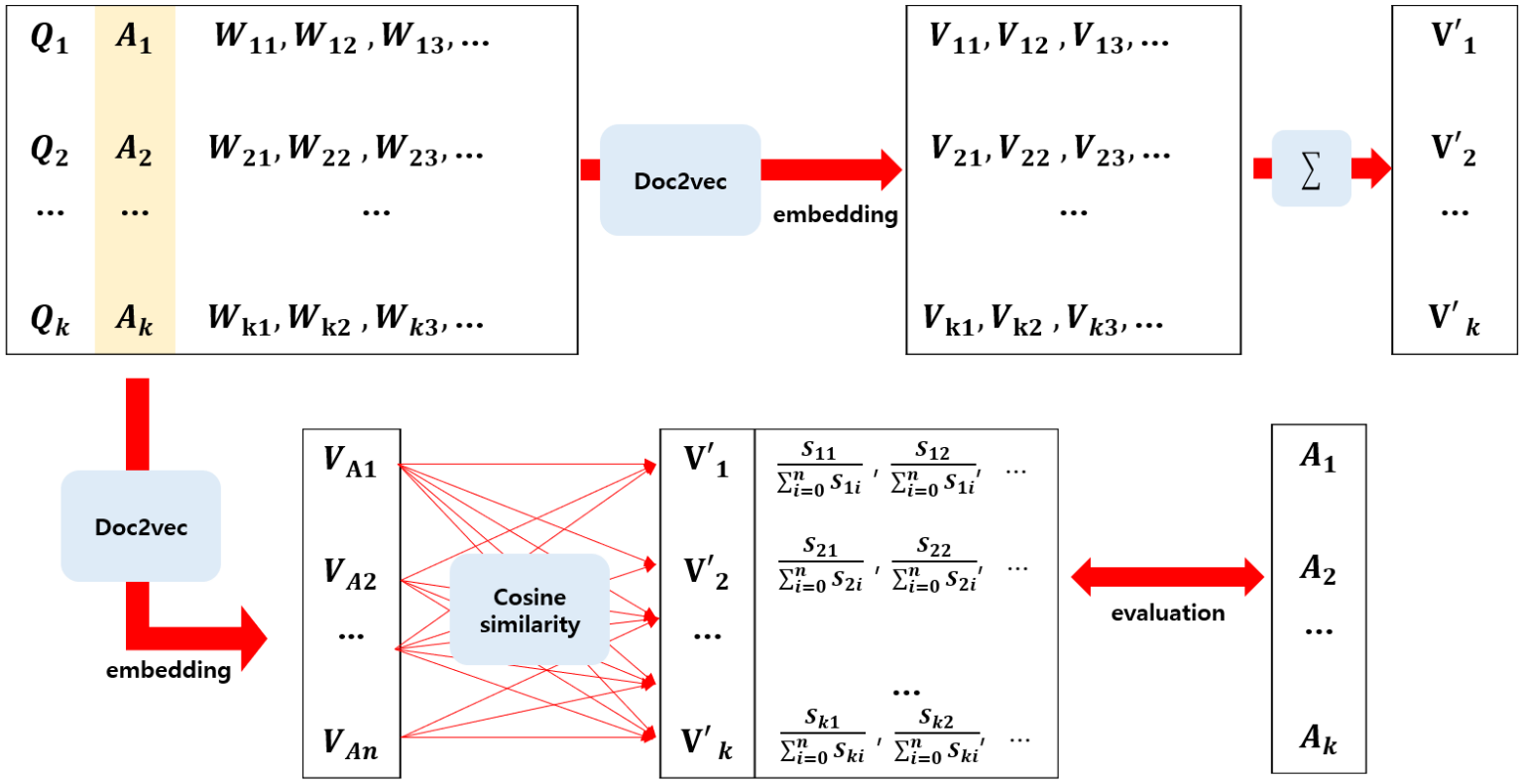



The paragraph vectors(label vectors) are asked to a prediction task about the next word in the sentence. Every paragraph is mapped to a unique vector. The paragraph vector and word vectors are averaged to predict the next word in a context.

PV distributed memory is employed.
Dimensionality of the feature vectors. - 300
Window size - 3
hierarchical softmax used
use the sum of the context word vectors.

# Modeling – Doc2vec

## Test set

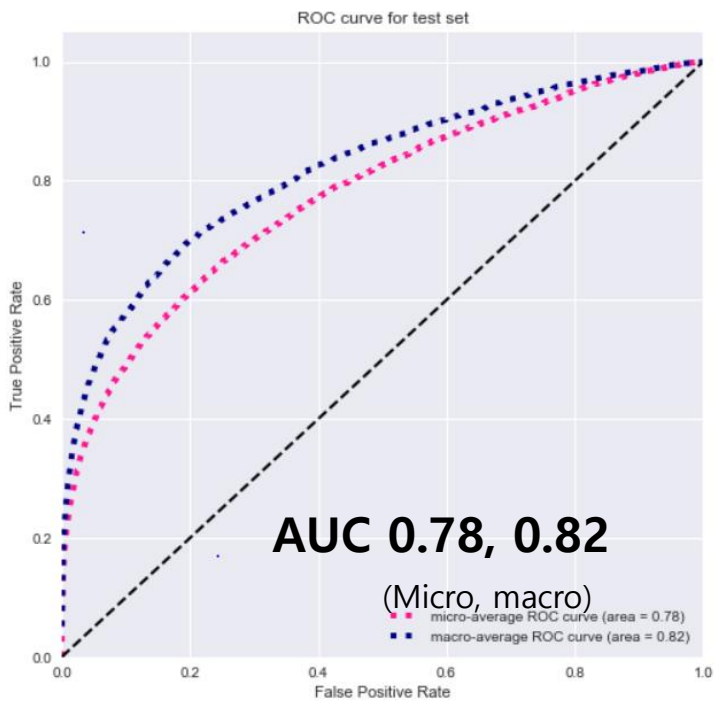



*Computes cosine similarity between a simple mean of the projection weight vectors of the given docs.

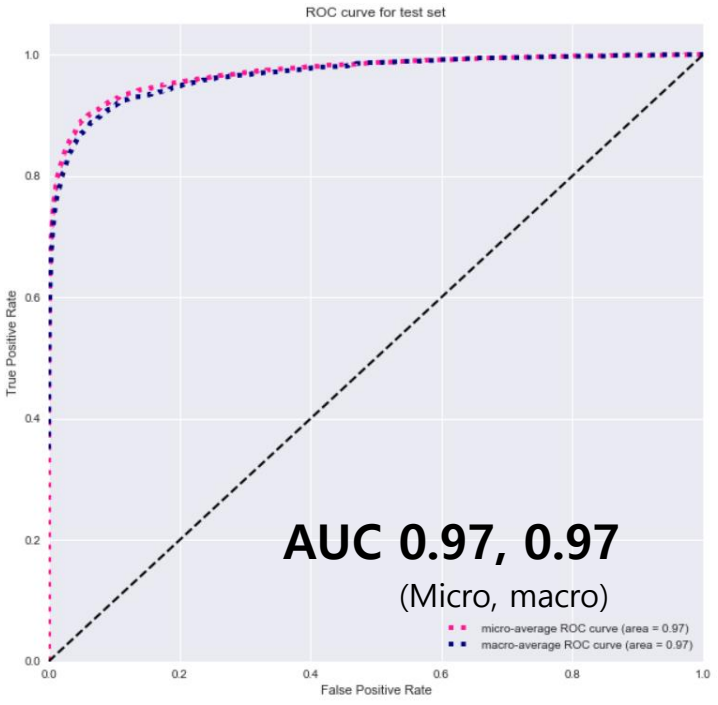
## Contents

# Model evaluation – ROC curve

**The ROC curve results for each labels are evaluated by two methods: micro-averaging and macro-averaging.**




**word2vec**

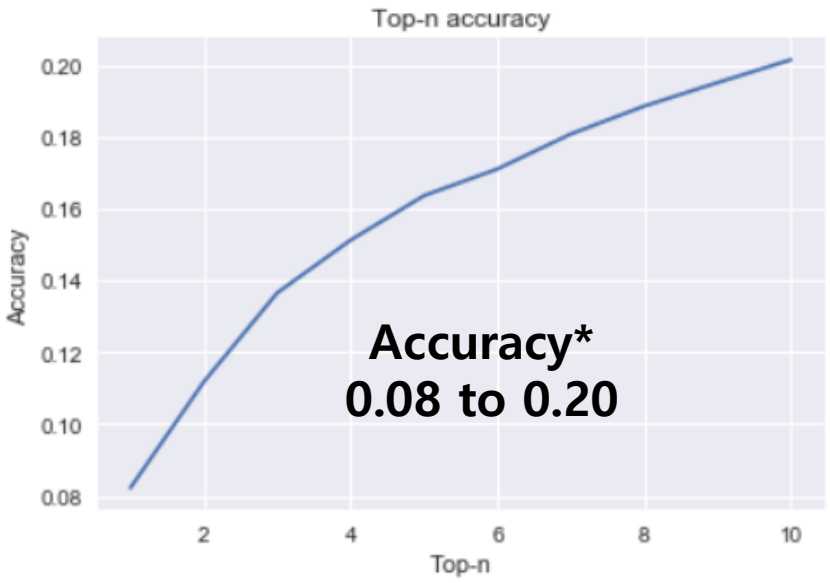



**doc2vec**

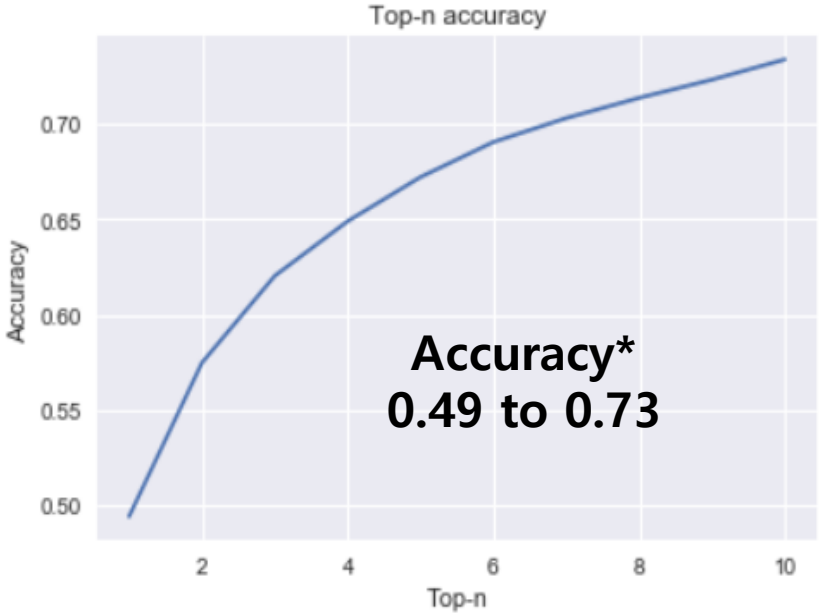
micro-averaging - considering each element of the label indicator matrix as a binary prediction
macro-averaging - gives equal weight to the classification of each label.

# Model evaluation – Top-n Accuracy approach

**Top-n accuracy approach results for each labels are evaluated. For example, Top-5 accuracy means that any of our model 5 highest probability answers must match the expected answer. (n:  1~10)**




**word2vec**




**doc2vec**

Accuracy* : top 1 and top 10 accuracy

# Discussion

- **Conclusion**

✓ Overall, doc2vec shows better performance than word2vec model

✓ Building a service by presenting n (at least 5) correct answer lists for new questions

✓ Application to speech recognition based movie recommendation service

- **Further study**

✓ Problems that questions about untrained movies

- complementing through learning synopsis of the movies

✓ A method for dealing with imbalanced movie data is needed

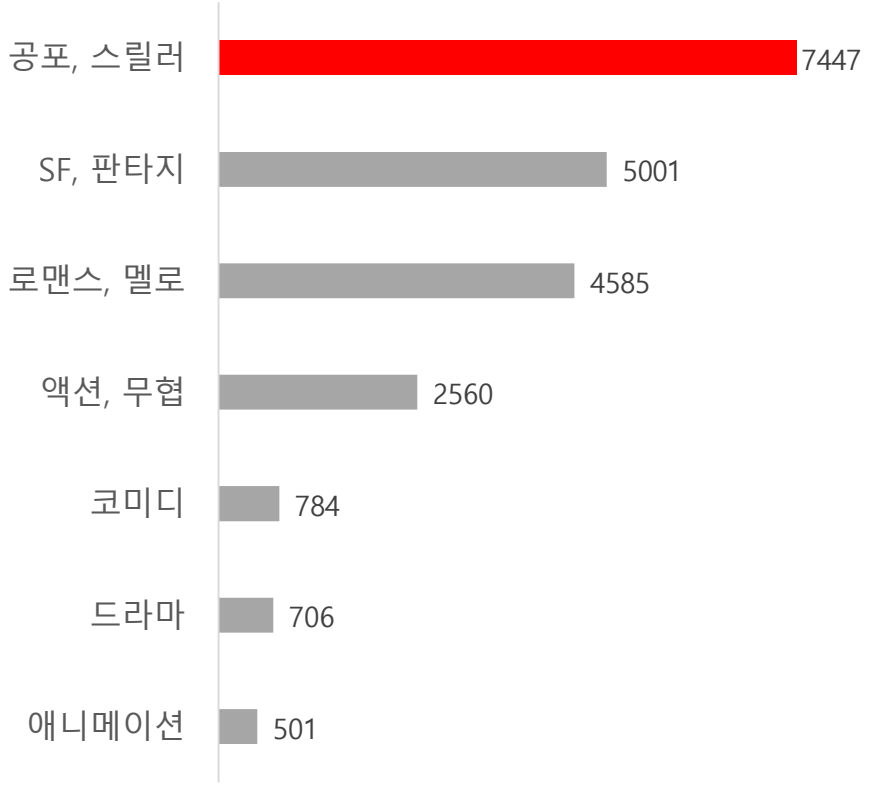
# Thank you

# APPENDIX

**We drew graphs to find which movies and which genres are highly asked. We could find that people wanted to find mysterious and thrilling movies**

## Asked Movie Ranking


## Asked Movie Genre



## Delete unnecessory phrase in question

At the beginning of the question and at the end, remove all phrases before and after the word.
If the word in the check words list (within 20% of the length of the question)

```
before cleaning...

좀 옛날 영화인데 방독면을 쓴살인마가 사람들을
죽이고 다니는영화였는데 내용은 잘기억은안나는데
후반부에 그살인마가 방독면을 벗어 던지고 곡괭이와 함께 동굴?같은데를 전구를 하나씩 곡괭이로 부숴가면서
걸어가는 장면이있었는데 기억이안나네요..


after cleaning...

방독면을 쓴살인마가 사람들을
죽이고 다니는영화였는데 내용은 잘기억은안나는데
후반부에 그살인마가 방독면을 벗어 던지고 곡괭이와 함께 동굴?같은데를 전구를 하나씩 곡괭이로 부숴가면서
걸어가는
```

Macro-average, Micro average

$$\text{Macro-average of precision} = \frac{\text{p1} + \text{p2} \ldots}{n}$$

$$\text{Macro-average of recall} = \frac{\text{r1} + \text{r2} \ldots}{n}$$

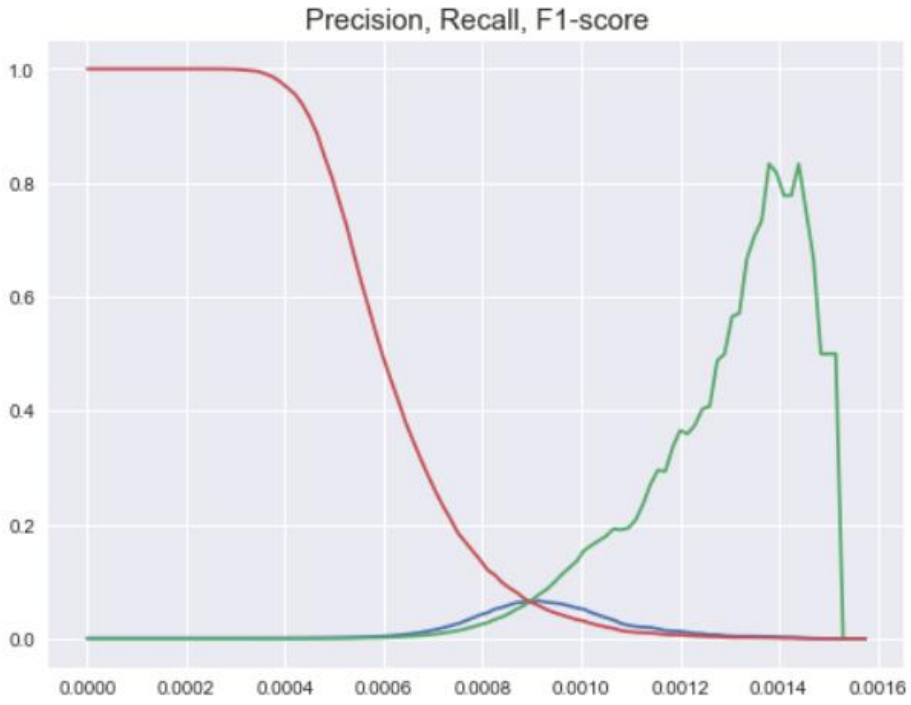
$$\text{Micro-average of precision} = \frac{\text{TP1+TP2} \ldots}{\text{TP1+TP2+FP1+FP2} \ldots}$$

$$\text{Micro-average of recall} = \frac{\text{TP1+TP2} \ldots}{\text{TP1+TP2+FN1+FN2} \ldots}$$

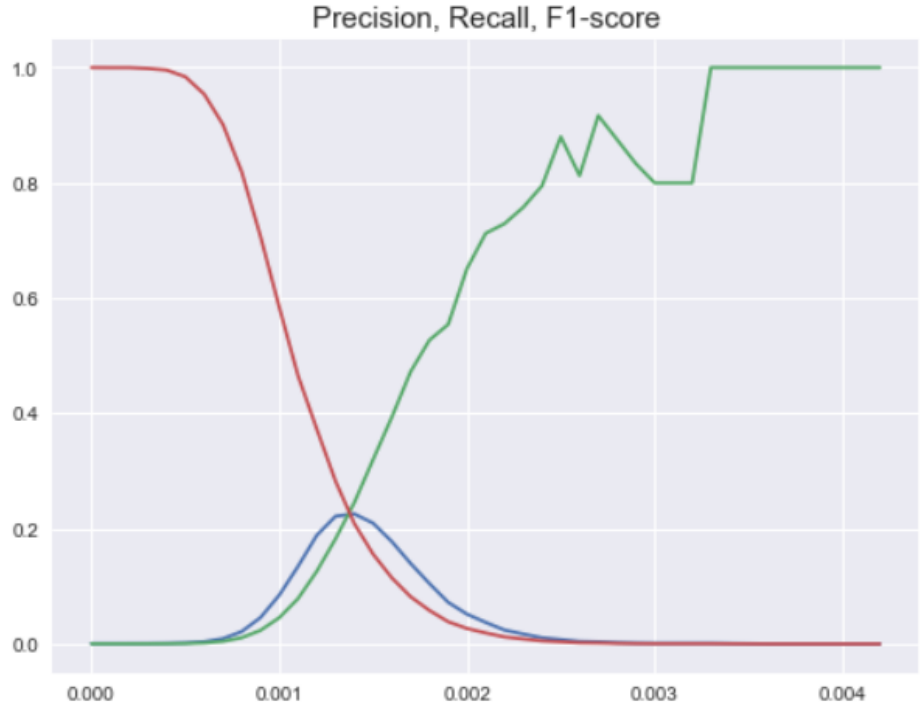
p = precision
r = recall
n= number of classes

# APPENDIX

## Precision, Recall, F1-score



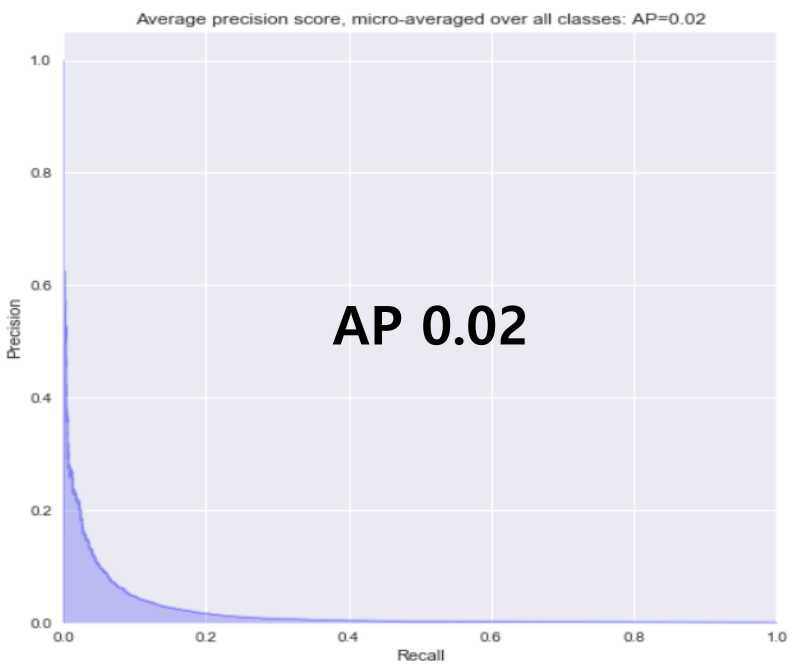

**word2vec**



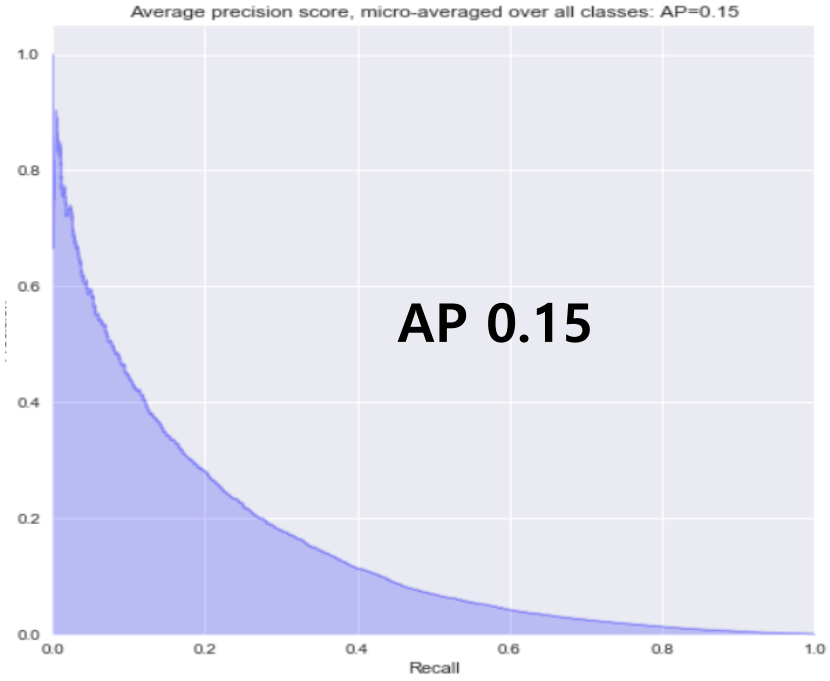

**doc2vec**

Red : precision, green : recall, blue : f1-score
*using 2021 labels, threshold step : 15e-6(word2vec),1e-04(doc2vec)

## Precision-recall curve for AP (Average Precision) Score - micro-average
## (y-axis : precision, x-axis : recall)




word2vec

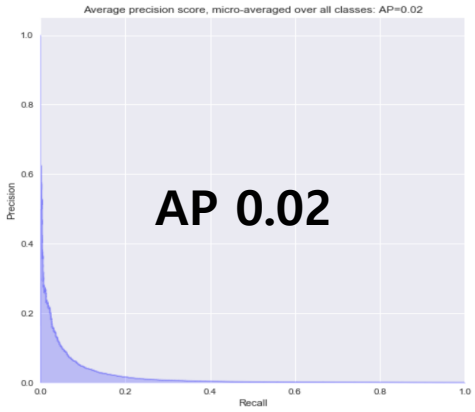
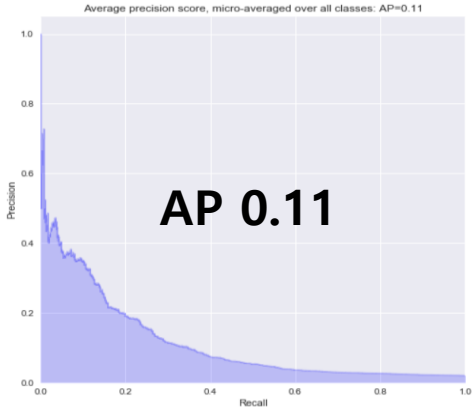
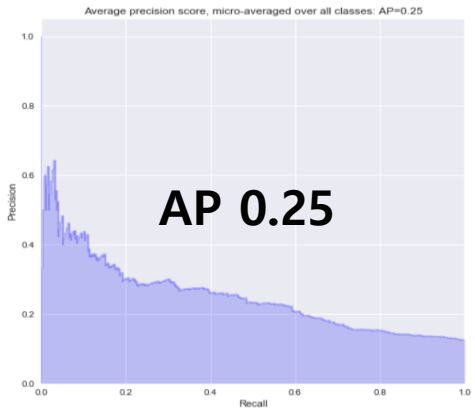
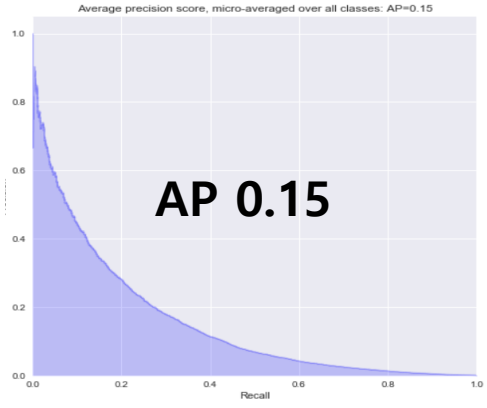
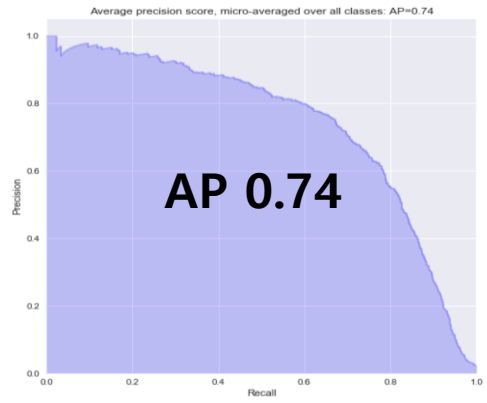
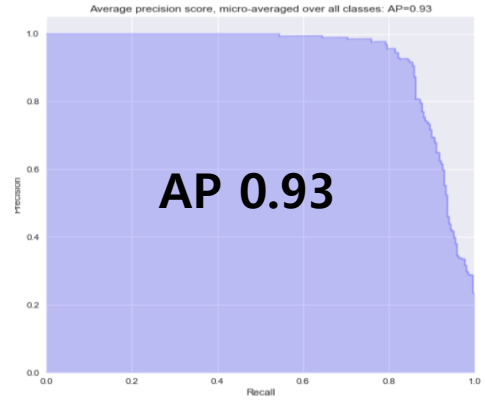



doc2vec

precision - a measure of result relevancy
recall - a measure of how many truly relevant results are returned.

# APPENDIX

## AP scores for another cutoff

| | Cutoff 3 | Cutoff 60 | Cutoff 130 |
|---|---|---|---|
| word2vec |   |   |   |
| Doc2vec |   |   |   |

*number of labels = 2021, 51 ,8 (duplicated movies cutoff – 3, 60, 130)
precision is a measure of result relevancy, recall is a measure of how many truly relevant results are returned.