

Deep Learning Analysis of **Korean Certificate** **Exam Questions**

DEEP LEARNING AND ITS APPLICATIONS TEAM 15

2017232044 KIM JINU
2017143004 CHOI HYEONGHO
2020147046 LEE HAKMIN

Table Of Contents

01. Abstract

02. Introduction

03. Problem Statement

04. Data Collection & Preprocessing

05. Model

- 1** Domain of Choice Categorization Model
- 2** Domain of Question Classification Model
- 3** Similar Questions Extraction Model

06. Conclusion

07. References

Abstract

Goal : Developing a model that can make managing exam data efficiently.

Data : Using Korean certificate examination.

Method : LSTM, BERT, KoNLPy

Model name

Method

Modeling data

Domain of Choice Categorization Model

LSTM

Our data

+

KoNLPy

Domain of Question Classification Model

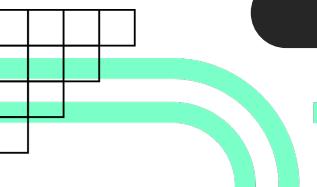
Bert

Our data

Similar Question Extraction Model

Bert

External data



Introduction

Why did we get to work on the project?

- Serious education gap in Korea

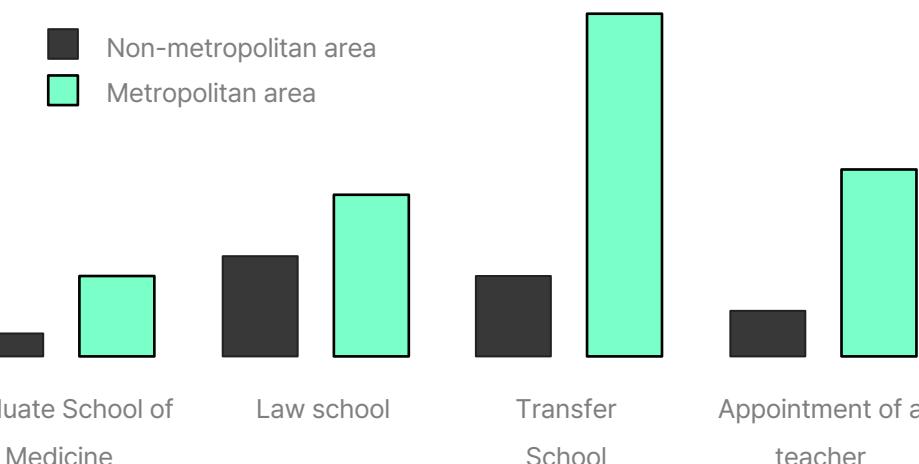
- Regional factor

- Education infrastructures are concentrated in certain regions.

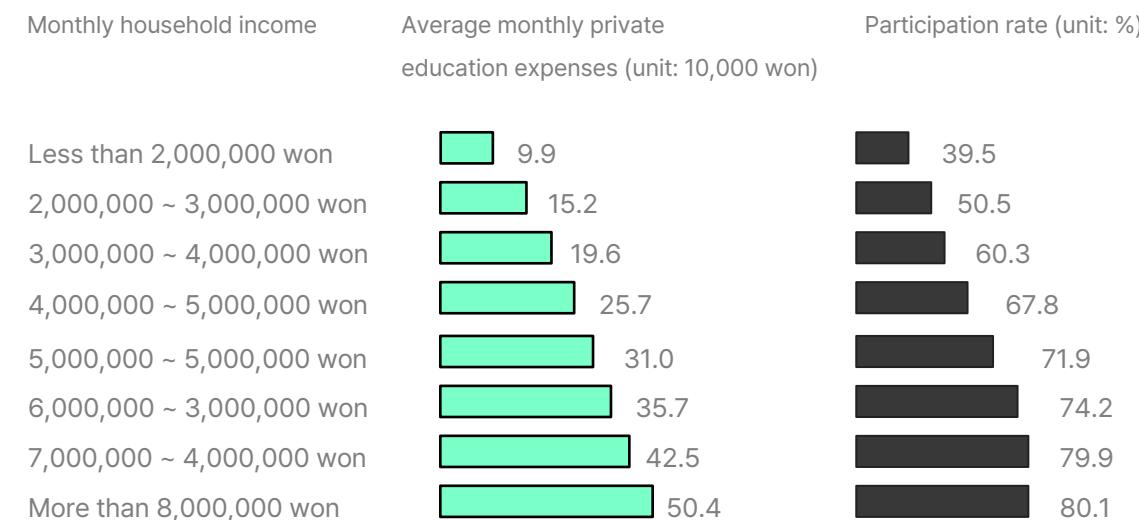
- Economic factor

- Differences in ability to pay for education create educational gaps.

Comparison of the number of metropolitan, non-metropolitan academies (2022)



Current status of private education expenses by income class (2020)



Introduction

Why did we get to work on the project?

- Can we use AI to solve this problem?

AI가 채점하고 선생님은 학생 관리…13兆 사교육 시장 판이 바뀐다

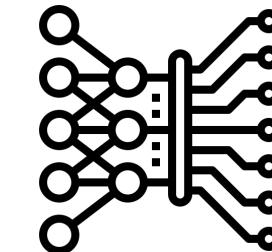
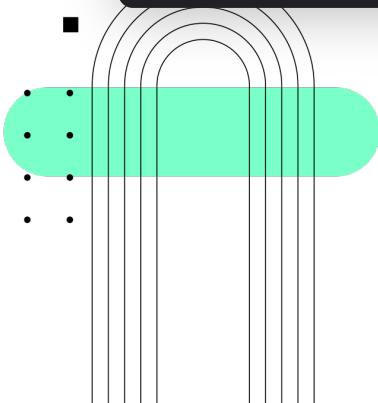
2022. 9. 15. — 세계는 지금 DX혁명 (4) DX로 신시장 여는 中企 에듀테크 스타트업 밀당PT 오프라인 교육이 강한 교육 시장 인공지능 학습관리 시스템 적용

교육시장에 AI 기술 도입 사례 확산 - 블록체인밸리

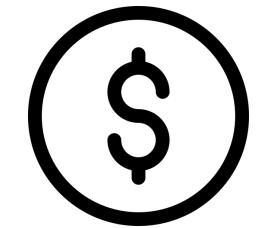
2022. 8. 2. — 에듀테크의 중심에 있는 인공지능(AI)은 분야를 막론하고 각 산업의 디지털 대전환을 이끌며 교육 시장 변화에 속도감을 더해주고 있다.

교육시장에 부는 인공지능 열풍 '에듀테크 진화' - 넥스트유니콘

AI, 빅데이터, IoT 등 소프트웨어 기반 에듀테크 관련 기업 가파른 성장 돌보여 '기대' - 비대면 교육 문화 확대로 학습도 편리하고 경제적으로!



Overcome
regional limitation



No expert
needed

Problem Statement

Existing problem

- Test question data has been managed in an analog method.

[1과목] 건축학개론 (20문제)

1. 경애영인 노인 일부 주민의 편의 증진 보상에 관한 법령에 따른
법률 청탁 및 대가로 실물 송부하지 않는 것은?
 ① 주택보수금
 ② 유동 및 건내설비
 ③ 장애인전용 주거구역
 ④ 주거법률 노이지 차지
 ⑤ 주거법률 노이지 차지 대수
 ⑥ 주거법률 노이지 차지 대수

2. 다음 중 주사파 건축의 기동간격 결정 요소와 가장 거리가 먼 것은?
 ① 핵심배치의 단위
 ② 주거배치의 단위
 ③ 주거법률 노이지 차지 대수
 ④ 주거법률 노이지 차지 대수

3. 우리나라 주택 환경부에서 문풀부분(개구부)의 면적이 큰
여러분 가능한 것은?
 ① 거점으로 허가를 원하는
 ② 허가를 원하는 주거구역에서
 ③ 충분하도록 허가를 하기 위해서
 ④ 상당의 대상을 고려해 허가를 원하는 허가를 위해서

4. 공간간격을 레이아웃(Lay out)한 설정으로 옮겨 않은 것은?
 ① 세대 중심의 대량생산에 유리하며 생산성이 높다.
 ② 허가되어야 하는 경우 소요 간의 위험 관계를 결정하는 것을 막는다.
 ③ 고밀도 주거구역은 조소노출이 많이 제한이 크고 수령이 적어 경쟁력이 있다.
 ④ 중앙화 공업, 시멘트, 공업 등 창작공간 등을 중심의 유통성이 크기 때문에 신속히 창작성이 대단히 고려가 필요 없다.

5. 예상되는 대상에게는 관할 설계로 옮겨 않은 것은?
 ① 단기형 영구구역이 가능하다.
 ② 소규모 주거구역에 적합하다.
 ③ 통증면적을 감소시킬 수 있는 주거구역이다.
 ④ 허가 및 계획승인을 가능하게 한다.

6. 고장터 형 병원에 관한 설계로 옮겨 않은 것은?
 ① 병원에 조성되는 병원을 확장할 수 있다.
 ② 병원에 조성되는 병원을 확장할 수 있다.
 ③ 각 병원 병원에 대단히 높다.
 ④ 병원의 규모에 대한 대단히 용이하다.

7. 주당 평균 40시간 이상되는 어느 학교에서 유성장에서의
수업이 1주에 10시간 이상되는 경우에 배정되어야 한다.
 ① 대학 1주에 10시간 이상되는 경우에 배정되어야 한다.
 ② 대학 1주에 10시간 이상되는 경우에 배정되어야 한다.
 ③ 대학 1주에 10시간 이상되는 경우에 배정되어야 한다.
 ④ 대학 1주에 10시간 이상되는 경우에 배정되어야 한다.

8. 폐기물 관리에 대한 대단히 용이하다.

9. 주거법률 노이지 차지에 대한 대단히 용이하다.

10. 주거법률 노이지 차지에 대한 대단히 용이하다.

11. 주거법률 노이지 차지에 대한 대단히 용이하다.

12. 주거법률 노이지 차지에 대한 대단히 용이하다.

13. 주거법률 노이지 차지에 대한 대단히 용이하다.

14. 주거법률 노이지 차지에 대한 대단히 용이하다.

15. 주거법률 노이지 차지에 대한 대단히 용이하다.

16. 주거법률 노이지 차지에 대한 대단히 용이하다.

17. 주거법률 노이지 차지에 대한 대단히 용이하다.

18. 주거법률 노이지 차지에 대한 대단히 용이하다.

19. 주거법률 노이지 차지에 대한 대단히 용이하다.

20. 주거법률 노이지 차지에 대한 대단히 용이하다.

21. 주거법률 노이지 차지에 대한 대단히 용이하다.

22. 주거법률 노이지 차지에 대한 대단히 용이하다.

23. 주거법률 노이지 차지에 대한 대단히 용이하다.

24. 주거법률 노이지 차지에 대한 대단히 용이하다.

25. 주거법률 노이지 차지에 대한 대단히 용이하다.

26. 주거법률 노이지 차지에 대한 대단히 용이하다.

27. 주거법률 노이지 차지에 대한 대단히 용이하다.

28. 주거법률 노이지 차지에 대한 대단히 용이하다.

29. 주거법률 노이지 차지에 대한 대단히 용이하다.

30. 주거법률 노이지 차지에 대한 대단히 용이하다.

31. 주거법률 노이지 차지에 대한 대단히 용이하다.

32. 주거법률 노이지 차지에 대한 대단히 용이하다.

33. 주거법률 노이지 차지에 대한 대단히 용이하다.

34. 주거법률 노이지 차지에 대한 대단히 용이하다.

35. 주거법률 노이지 차지에 대한 대단히 용이하다.

36. 주거법률 노이지 차지에 대한 대단히 용이하다.

37. 주거법률 노이지 차지에 대한 대단히 용이하다.

38. 주거법률 노이지 차지에 대한 대단히 용이하다.

39. 주거법률 노이지 차지에 대한 대단히 용이하다.

40. 주거법률 노이지 차지에 대한 대단히 용이하다.

41. 주거법률 노이지 차지에 대한 대단히 용이하다.

42. 주거법률 노이지 차지에 대한 대단히 용이하다.

43. 주거법률 노이지 차지에 대한 대단히 용이하다.

44. 주거법률 노이지 차지에 대한 대단히 용이하다.

45. 주거법률 노이지 차지에 대한 대단히 용이하다.

46. 주거법률 노이지 차지에 대한 대단히 용이하다.

47. 주거법률 노이지 차지에 대한 대단히 용이하다.

48. 주거법률 노이지 차지에 대한 대단히 용이하다.

49. 주거법률 노이지 차지에 대한 대단히 용이하다.

50. 주거법률 노이지 차지에 대한 대단히 용이하다.

51. 주거법률 노이지 차지에 대한 대단히 용이하다.

52. 주거법률 노이지 차지에 대한 대단히 용이하다.

53. 주거법률 노이지 차지에 대한 대단히 용이하다.

54. 주거법률 노이지 차지에 대한 대단히 용이하다.

55. 주거법률 노이지 차지에 대한 대단히 용이하다.

56. 주거법률 노이지 차지에 대한 대단히 용이하다.

57. 주거법률 노이지 차지에 대한 대단히 용이하다.

58. 주거법률 노이지 차지에 대한 대단히 용이하다.

59. 주거법률 노이지 차지에 대한 대단히 용이하다.

60. 주거법률 노이지 차지에 대한 대단히 용이하다.

61. 주거법률 노이지 차지에 대한 대단히 용이하다.

62. 주거법률 노이지 차지에 대한 대단히 용이하다.

63. 주거법률 노이지 차지에 대한 대단히 용이하다.

64. 주거법률 노이지 차지에 대한 대단히 용이하다.

65. 주거법률 노이지 차지에 대한 대단히 용이하다.

66. 주거법률 노이지 차지에 대한 대단히 용이하다.

67. 주거법률 노이지 차지에 대한 대단히 용이하다.

68. 주거법률 노이지 차지에 대한 대단히 용이하다.

69. 주거법률 노이지 차지에 대한 대단히 용이하다.

70. 주거법률 노이지 차지에 대한 대단히 용이하다.

71. 주거법률 노이지 차지에 대한 대단히 용이하다.

72. 주거법률 노이지 차지에 대한 대단히 용이하다.

73. 주거법률 노이지 차지에 대한 대단히 용이하다.

74. 주거법률 노이지 차지에 대한 대단히 용이하다.

75. 주거법률 노이지 차지에 대한 대단히 용이하다.

76. 주거법률 노이지 차지에 대한 대단히 용이하다.

77. 주거법률 노이지 차지에 대한 대단히 용이하다.

78. 주거법률 노이지 차지에 대한 대단히 용이하다.

79. 주거법률 노이지 차지에 대한 대단히 용이하다.

80. 주거법률 노이지 차지에 대한 대단히 용이하다.

81. 주거법률 노이지 차지에 대한 대단히 용이하다.

82. 주거법률 노이지 차지에 대한 대단히 용이하다.

83. 주거법률 노이지 차지에 대한 대단히 용이하다.

84. 주거법률 노이지 차지에 대한 대단히 용이하다.

85. 주거법률 노이지 차지에 대한 대단히 용이하다.

86. 주거법률 노이지 차지에 대한 대단히 용이하다.

87. 주거법률 노이지 차지에 대한 대단히 용이하다.

88. 주거법률 노이지 차지에 대한 대단히 용이하다.

89. 주거법률 노이지 차지에 대한 대단히 용이하다.

90. 주거법률 노이지 차지에 대한 대단히 용이하다.

91. 주거법률 노이지 차지에 대한 대단히 용이하다.

92. 주거법률 노이지 차지에 대한 대단히 용이하다.

93. 주거법률 노이지 차지에 대한 대단히 용이하다.

94. 주거법률 노이지 차지에 대한 대단히 용이하다.

95. 주거법률 노이지 차지에 대한 대단히 용이하다.

96. 주거법률 노이지 차지에 대한 대단히 용이하다.

97. 주거법률 노이지 차지에 대한 대단히 용이하다.

98. 주거법률 노이지 차지에 대한 대단히 용이하다.

99. 주거법률 노이지 차지에 대한 대단히 용이하다.

100. 주거법률 노이지 차지에 대한 대단히 용이하다.

101. 주거법률 노이지 차지에 대한 대단히 용이하다.

102. 주거법률 노이지 차지에 대한 대단히 용이하다.

103. 주거법률 노이지 차지에 대한 대단히 용이하다.

104. 주거법률 노이지 차지에 대한 대단히 용이하다.

105. 주거법률 노이지 차지에 대한 대단히 용이하다.

106. 주거법률 노이지 차지에 대한 대단히 용이하다.

107. 주거법률 노이지 차지에 대한 대단히 용이하다.

108. 주거법률 노이지 차지에 대한 대단히 용이하다.

109. 주거법률 노이지 차지에 대한 대단히 용이하다.

110. 주거법률 노이지 차지에 대한 대단히 용이하다.

111. 주거법률 노이지 차지에 대한 대단히 용이하다.

112. 주거법률 노이지 차지에 대한 대단히 용이하다.

113. 주거법률 노이지 차지에 대한 대단히 용이하다.

114. 주거법률 노이지 차지에 대한 대단히 용이하다.

115. 주거법률 노이지 차지에 대한 대단히 용이하다.

116. 주거법률 노이지 차지에 대한 대단히 용이하다.

117. 주거법률 노이지 차지에 대한 대단히 용이하다.

118. 주거법률 노이지 차지에 대한 대단히 용이하다.

119. 주거법률 노이지 차지에 대한 대단히 용이하다.

120. 주거법률 노이지 차지에 대한 대단히 용이하다.

121. 주거법률 노이지 차지에 대한 대단히 용이하다.

122. 주거법률 노이지 차지에 대한 대단히 용이하다.

123. 주거법률 노이지 차지에 대한 대단히 용이하다.

124. 주거법률 노이지 차지에 대한 대단히 용이하다.

125. 주거법률 노이지 차지에 대한 대단히 용이하다.

126. 주거법률 노이지 차지에 대한 대단히 용이하다.

127. 주거법률 노이지 차지에 대한 대단히 용이하다.

128. 주거법률 노이지 차지에 대한 대단히 용이하다.

129. 주거법률 노이지 차지에 대한 대단히 용이하다.

130. 주거법률 노이지 차지에 대한 대단히 용이하다.

131. 주거법률 노이지 차지에 대한 대단히 용이하다.

132. 주거법률 노이지 차지에 대한 대단히 용이하다.

133. 주거법률 노이지 차지에 대한 대단히 용이하다.

134. 주거법률 노이지 차지에 대한 대단히 용이하다.

135. 주거법률 노이지 차지에 대한 대단히 용이하다.

136. 주거법률 노이지 차지에 대한 대단히 용이하다.

137. 주거법률 노이지 차지에 대한 대단히 용이하다.

138. 주거법률 노이지 차지에 대한 대단히 용이하다.

139. 주거법률 노이지 차지에 대한 대단히 용이하다.

140. 주거법률 노이지 차지에 대한 대단히 용이하다.

141. 주거법률 노이지 차지에 대한 대단히 용이하다.

142. 주거법률 노이지 차지에 대한 대단히 용이하다.

143. 주거법률 노이지 차지에 대한 대단히 용이하다.

144. 주거법률 노이지 차지에 대한 대단히 용이하다.

145. 주거법률 노이지 차지에 대한 대단히 용이하다.

146. 주거법률 노이지 차지에 대한 대단히 용이하다.

147. 주거법률 노이지 차지에 대한 대단히 용이하다.

148. 주거법률 노이지 차지에 대한 대단히 용이하다.

149. 주거법률 노이지 차지에 대한 대단히 용이하다.

150. 주거법률 노이지 차지에 대한 대단히 용이하다.

151. 주거법률 노이지 차지에 대한 대단히 용이하다.

152. 주거법률 노이지 차지에 대한 대단히 용이하다.

153. 주거법률 노이지 차지에 대한 대단히 용이하다.

154. 주거법률 노이지 차지에 대한 대단히 용이하다.

155. 주거법률 노이지 차지에 대한 대단히 용이하다.

156. 주거법률 노이지 차지에 대한 대단히 용이하다.

157. 주거법률 노이지 차지에 대한 대단히 용이하다.

158. 주거법률 노이지 차지에 대한 대단히 용이하다.

159. 주거법률 노이지 차지에 대한 대단히 용이하다.

160. 주거법률 노이지 차지에 대한 대단히 용이하다.

161. 주거법률 노이지 차지에 대한 대단히 용이하다.

162. 주거법률 노이지 차지에 대한 대단히 용이하다.

163. 주거법률 노이지 차지에 대한 대단히 용이하다.

164. 주거법률 노이지 차지에 대한 대단히 용이하다.

165. 주거법률 노이지 차지에 대한 대단히 용이하다.

166. 주거법률 노이지 차지에 대한 대단히 용이하다.

167. 주거법률 노이지 차지에 대한 대단히 용이하다.

168. 주거법률 노이지 차지에 대한 대단히 용이하다.

169. 주거법률 노이지 차지에 대한 대단히 용이하다.

170. 주거법률 노이지 차지에 대한 대단히 용이하다.

171. 주거법률 노이지 차지에 대한 대단히 용이하다.

172. 주거법률 노이지 차지에 대한 대단히 용이하다.

173. 주거법률 노이지 차지에 대한 대단히 용이하다.

174. 주거법률 노이지 차지에 대한 대단히 용이하다.

175. 주거법률 노이지 차지에 대한 대단히 용이하다.

176. 주거법률 노이지 차지에 대한 대단히 용이하다.

177. 주거법률 노이지 차지에 대한 대단히 용이하다.

178. 주거법률 노이지 차지에 대한 대단히 용이하다.

Problem Statement

Solution proposal

Input

Output

Sol 1

Choice text

Exam domain

Sol 2

Question main text

Exam domain

Sol 3

Keyword

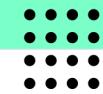
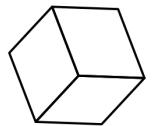
Question type

Sol 4

Main text

Similar Main Text

	Exam name	Question type	Main text	Choice text	Keyword
	건축기사	1과목 : 건축계획			
Sol 1		1. 장애인·노인·임산부 등의 편의증진 보장에 관한 법령에 따른 편의시설 중 매개시설에 속하지 않는 것은? ① 주출입구 접근로 ② 유도 및 안내설비 ③ 장애인전용 주차구역 ④ 주출입구 높이차이 제거			
Sol 2		2. 다음 중 사무소 건축의 기동간격 결정 요소와 가장 거리가 먼 것은? ① 책상배치의 단위 ② 주차배치의 단위 ③ 엘리베이터의 설치 대수 ④ 채광상 층높이에 의한 깊이			
Sol 3		3. 우리나라 전통 한식주택에서 문꼴부분(개구부)의 면적이 큰 이유로 가장 적합한 것은? ① 겨울의 방한을 위해서 ② 하절기 고온다습을 견디기 위해서 ③ 출입하는데 편리하게 하기 위해서 ④ 상부의 하중을 효과적으로 지지하기 위해서			
Sol 4		4. 공장건축의 레이아웃(Lay out)에 관한 설명으로 옳지 않은 것은? ① 제품종심의 레이아웃은 대량생산에 유리하며 생산성이 높다. ② 레이아웃이란 공장건축의 평면요소간의 위치 관계를 결정하는 것을 말한다. ③ 고정식 레이아웃은 조선소와 같이 제품이 크고 수량이 적은 경우에 행해진다. ④ 중화학 공업, 시멘트 공업 등 장치공업 등은 시설의 유통성이 크기 때문에 신설시 장래성에 대한 고려가 필요 없다.			



Problem Statement

Solution proposal

Input

Sol 1

Choice text

Output

Exam domain

Sol 2

Question main text

Exam domain

Sol 3

Keyword

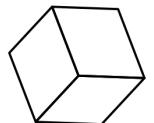
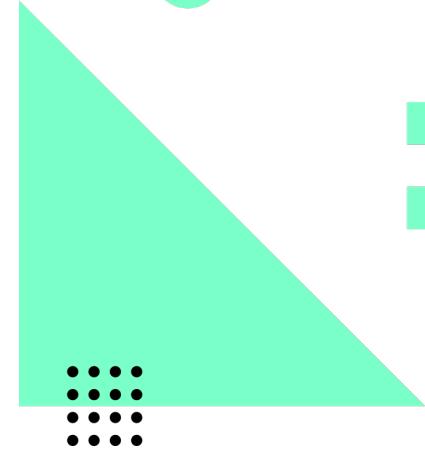
Question type

Sol 4

Main text

Similar Main Text

	Exam name	Question type
	1과목 : 건축계획	
1.	장애인 · 노인 · 임산부 등의 편의증진 보장에 관한 법령에 따른 편의시설 중 매개시설에 속하지 않는 것은?	
①	주출입구 접근로	② 유도 및 안내설비
③	장애인전용 주차구역	④ 주출입구 높이차이 제거
2.	다음 중 사무소 건축의 기동간격 결정 요소와 가장 거리가 먼 것은?	
①	책상배치의 단위	② 주차배치의 단위
③	엘리베이터의 설치 대수	④ 채광상 층높이에 의한 깊이
3.	우리나라 전통 한식주택에서 문꼴부분(개구부)의 면적이 큰 이유로 가장 적합한 것은?	
①	겨울의 방한을 위해서	
②	하절기 고온다습을 견디기 위해서	
③	출입하는데 편리하게 하기 위해서	
④	상부의 하중을 효과적으로 지지하기 위해서	
4.	공장건축의 레이아웃(Lay out)에 관한 설명으로 옳지 않은 것은?	
①	제품종심의 레이아웃은 대량생산에 유리하며 생산성이 높다.	
②	레이아웃이란 공장건축의 평면요소간의 위치 관계를 결정하는 것을 말한다.	
③	고정식 레이아웃은 조선소와 같이 제품이 크고 수량이 적은 경우에 행해진다.	
④	중화학 공업, 시멘트 공업 등 장치공업 등은 시설의 유통성이 크기 때문에 신설시 장래성에 대한 고려가 필요 없다.	



Problem Statement

Solution proposal

Input

Sol 1

Choice text

Output

Exam domain

Sol 2

Question main text

Exam domain

Sol 3

Keyword

Question type

Sol 4

Main text

Similar Main Text

건축기사

Exam name

1교시 : 건축계획

Question type

1. 장애인·노인·임산부 등의 편의증진 보장에 관한 법령에 따른 편의시설 중 매개시설에 속하지 않는 것은?

- ① 주출입구 접근로
- ② 유도 및 안내설비
- ③ 장애인전용 주차구역
- ④ 주출입구 높이차이 제거

2. 다음 중 사무소 건축의 기동간격 결정 요소와 가장 거리가 먼 것은?

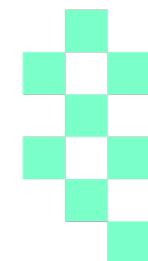
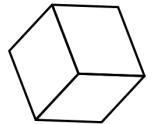
- ① 책상배치의 단위
- ② 주차배치의 단위
- ③ 엘리베이터의 설치 대수
- ④ 채광상 층높이에 의한 깊이

3. 우리나라 전통 한식주택에서 문꼴부분(개구부)의 면적이 큰 이유로 가장 적합한 것은?

- ① 겨울의 방한을 위해서
- ② 하절기 고온다습을 견디기 위해서
- ③ 출입하는데 편리하게 하기 위해서
- ④ 상부의 하중을 효과적으로 지지하기 위해서

4. 금장건축은 레이아웃(Lay out)에 관한 설명으로 옳지 않은 것은?

- ① 제품중심의 레이아웃은 대량생산에 유리하며 생산성이 높다.
- ② 레이아웃이란 공장건축의 평면요소간의 위치 관계를 결정하는 것을 말한다.
- ③ 고정식 레이아웃은 조선소와 같이 제품이 크고 수량이 적은 경우에 행해진다.
- ④ 중화학 공업, 시멘트 공업 등 장치공업 등은 시설의 유통성이 크기 때문에 신설시 장래성에 대한 고려가 필요 없다.



Main text

Choice text

Keyword

Problem Statement

Solution proposal

Input

Sol 1

Choice text

Output

Exam domain

Sol 2

Question main text

Exam domain

Sol 3

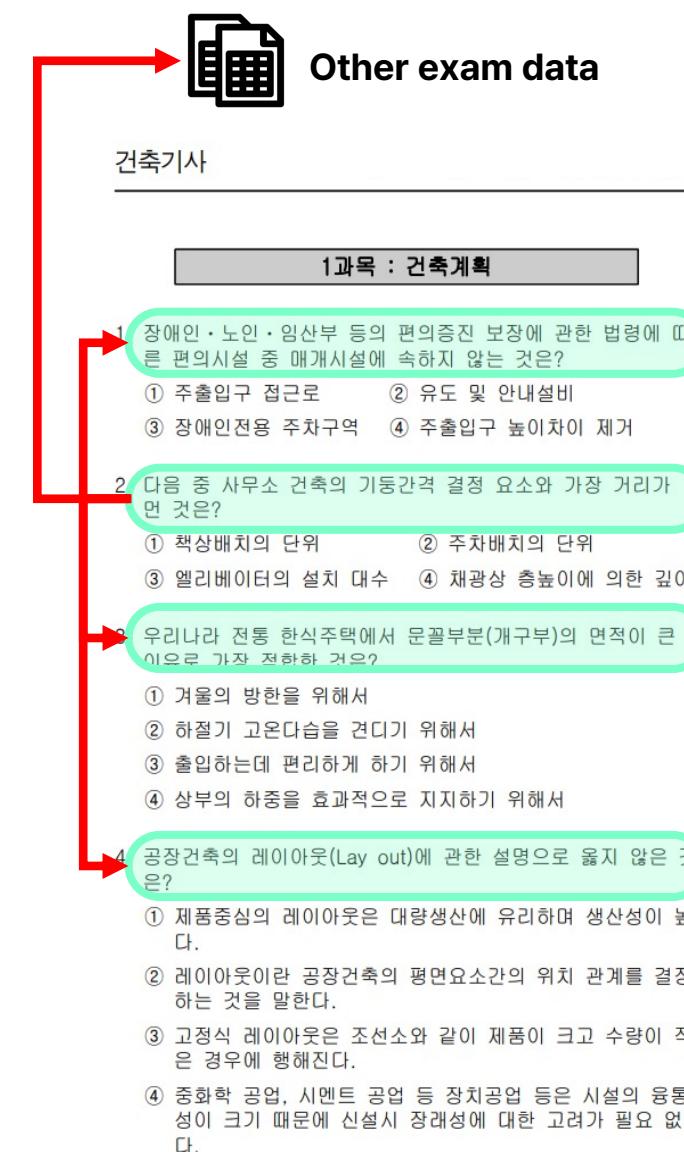
Keyword

Question type

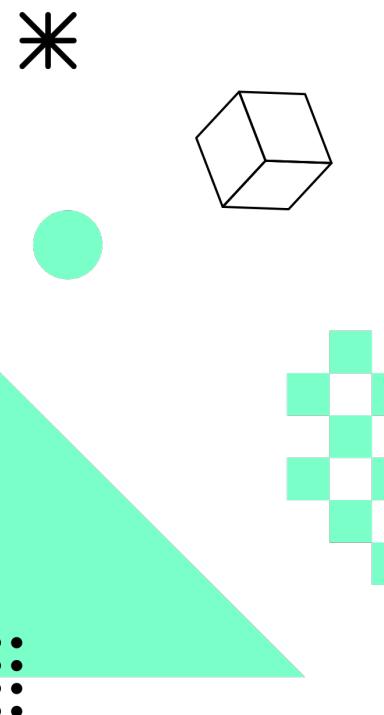
Sol 4

Main text

Similar Main Text



Main text



Problem Statement

Solution proposal

Input

Output

Sol 1 Choice text

Exam domain

Model 1

Sol 2 Question main text

Exam domain

Model 2

Sol 3 Keyword

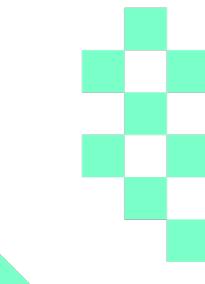
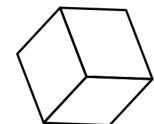
Question type

Failed

Sol 4 Main text

Similar Main Text

Model 4



Data Collection & Preprocessing

Data collection

- **What** : Korean certificate exam data
- **Where** : Korean testbank website (<http://q.fran.kr>)
- **How** : Used the LISTLY Chrome extension

* We manually extracted only necessary data and added 'question keyword' column.

** Necessary data : Certificate name, Problem domain, Question text, Choice text

LABEL-1	LABEL-2	LABEL-3	LABEL-4	LABEL-5	LABEL-6	LABEL-7	LABEL-8	LABEL-9
안전관리론				1. 매슬로우(Maslow)의	①	https://q.fran.kr/img/ch ① 안전 욕구	②	
	https://q.fran.kr/img/o/	https://q.fran.kr/img/x.p	https://q.fran.kr/img/t.p	2. A4사업장의 현황이 다	①	https://q.fran.kr/img/ch ① 0.22	②	
	https://q.fran.kr/img/o/	https://q.fran.kr/img/x.p	https://q.fran.kr/img/t.p	3. 보호구 사용 안전 확인	①	https://q.fran.kr/img/ch ① ㄱ, ㄴ, ㄷ	②	
	https://q.fran.kr/img/o/	https://q.fran.kr/img/x.p	https://q.fran.kr/img/t.p	4. 학습지도의 형태 중	①	https://q.fran.kr/img/ch ① 포럼(Forum)	②	
	https://q.fran.kr/img/o/	https://q.fran.kr/img/x.p	https://q.fran.kr/img/t.p	5. 보호구 안전 인증 고시	①	https://q.fran.kr/img/ch ① #2 ~ #3	②	
	https://q.fran.kr/img/o/	https://q.fran.kr/img/x.p	https://q.fran.kr/img/t.p	6. 산업재해의 분석 및	①	https://q.fran.kr/img/ch ① 관리도	②	
	https://q.fran.kr/img/o/	https://q.fran.kr/img/x.p	https://q.fran.kr/img/t.p	7. 산업안전보건법 평상	①	https://q.fran.kr/img/ch ① ㄱ, ㄴ, ㄷ, ㄹ	②	
	https://q.fran.kr/img/o/	https://q.fran.kr/img/x.p	https://q.fran.kr/img/t.p	8. 억측판단이 발생하는	①	https://q.fran.kr/img/ch ① 정보가 불확실할 때	②	
	https://q.fran.kr/img/o/	https://q.fran.kr/img/x.p	https://q.fran.kr/img/t.p	9. 하인리히의 사고예방	①	https://q.fran.kr/img/ch ① 사실의 발견	②	
	https://q.fran.kr/img/o/	https://q.fran.kr/img/x.p	https://q.fran.kr/img/t.p	9. 하인리히의 사고예방	①	https://q.fran.kr/img/ch ① 사실의 발견	②	
	https://q.fran.kr/img/o/	https://q.fran.kr/img/x.p	https://q.fran.kr/img/t.p	11. 산업안전보건법 평상	①	https://q.fran.kr/img/ch ① 유해인자의 노출기준	②	
	https://q.fran.kr/img/o/	https://q.fran.kr/img/x.p	https://q.fran.kr/img/t.p	12. 버드(Bird)의 재해분	①	https://q.fran.kr/img/ch ① 200	②	
	https://q.fran.kr/img/o/	https://q.fran.kr/img/x.p	https://q.fran.kr/img/t.p	13. 산업안전보건법 평상	①	https://q.fran.kr/img/ch ① 비계의 조립순서 및	②	
	https://q.fran.kr/img/o/	https://q.fran.kr/img/x.p	https://q.fran.kr/img/t.p	14. 산업안전보건법 평상	①	https://q.fran.kr/img/ch ① 위험장소 경고	②	
	https://q.fran.kr/img/o/	https://q.fran.kr/img/x.p	https://q.fran.kr/img/t.p	15. 학습정도(Level of	①	https://q.fran.kr/img/ch ① 인지 → 이해 → 지각	②	
	https://q.fran.kr/img/o/	https://q.fran.kr/img/x.p	https://q.fran.kr/img/t.p	16. 기업 내 정형교육 중	①	https://q.fran.kr/img/ch ① Job Method	②	
	https://q.fran.kr/img/o/	https://q.fran.kr/img/x.p	https://q.fran.kr/img/t.p	17. 레빈(Lewin)의 법칙	①	https://q.fran.kr/img/ch ① 행동	②	
	https://q.fran.kr/img/o/	https://q.fran.kr/img/x.p	https://q.fran.kr/img/t.p	18. 재해원인을 직접원	①	https://q.fran.kr/img/ch ① 물적 원인	②	
	https://q.fran.kr/img/o/	https://q.fran.kr/img/x.p	https://q.fran.kr/img/t.p	19. 산업안전보건법 평상	①	https://q.fran.kr/img/ch ① 일무 수행 내용의 기	②	
	https://q.fran.kr/img/o/	https://q.fran.kr/img/x.p	https://q.fran.kr/img/t.p	20. 헤드십(headship)의	①	https://q.fran.kr/img/ch ① 지휘형태는 권위주의	②	
인간공학 및 시스템안전								
	https://q.fran.kr/img/o/	https://q.fran.kr/img/x.p	https://q.fran.kr/img/t.p	21. 위험분석 기법 중 시	①	https://q.fran.kr/img/ch ① PHA	②	
	https://q.fran.kr/img/o/	https://q.fran.kr/img/x.p	https://q.fran.kr/img/t.p	22. 상황해석을 잘못하	①	https://q.fran.kr/img/ch ① 실수(Slip)	②	

Raw data example

The screenshot shows a web page titled '건축기사' (Architectural Engineer) with a search bar. Below the search bar are tabs for '문제' (Problem), '시험' (Exam), '노트' (Notes), and '포럼' (Forum). The main content area displays a question from the 2017년 2회 A exam. The question is: "47. 그림과 같은 부정정 라멘에서 A점의 MB는?" (What is the bending moment at point A for the given frame structure?). Below the question is a diagram of a frame structure with dimensions: height h=3m, width L=12m, and a uniformly distributed load w=10kN/m. The frame has two internal supports labeled K=1 and K=2. Point A is at the bottom-left corner, and point B is at the top-left corner. The diagram shows a coordinate system with x-axis horizontal and y-axis vertical. The question also lists four options: ① 0kN·m, ② 20kN·m, ③ 40kN·m, and ④ 60kN·m.

Website example

Data Collection & Preprocessing

Data Preprocessing

- Missing value handling

- Remove rows that has null value.

- Data type handling

- Replace numbers, special characters by regular expressions
- Remove other things except **Korean, English, and spacing.**
- Use this method when inferring a domain with choice text and main text.

건축기사

Exam name

1과목 : 건축계획

Question type

1. 장애인·노인·임산부 등의 편의증진 보장에 관한 법령에 따른 편의시설 중 매개시설에 속하지 않는 것은?

- ① 주출입구 접근로 ② 유도 및 안내설비
③ 장애인전용 주차구역 ④ 주출입구 높이차이 제거

2. 다음 중 사무소 건축의 **기둥간격 결정 요소**가 가장 거리가 먼 것은?

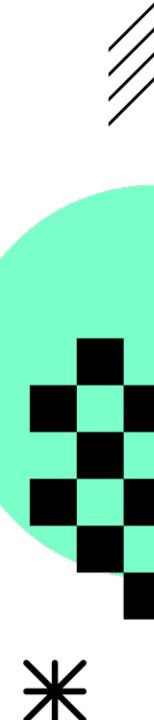
- ① 책상배치의 단위 ② 주차배치의 단위
③ 엘리베이터의 설치 대수 ④ 채광상 층높이에 의한 깊이

3. 우리나라 전통 한식주택에서 문꼴부분(개구부)의 면적이 큰 이유로 가장 적합한 것은?

- ① 겨울의 방한을 위해서
② 하절기 고온다습을 견디기 위해서
③ 출입하는데 편리하게 하기 위해서
④ 상부의 하중을 효과적으로 지지하기 위해서

4. 공장건축의 레이아웃(Lay out)에 관한 설명으로 옳지 않은 것은?

- ① 제품중심의 레이아웃은 대량생산에 유리하며 생산성이 높다.
② 레이아웃이란 공장건축의 평면요소간의 위치 관계를 결정하는 것을 말한다.
③ 고정식 레이아웃은 조선소와 같이 제품이 크고 수량이 적은 경우에 행해진다.
④ 중화학 공업, 시멘트 공업 등 장치공업 등은 시설의 융통성이 크기 때문에 신설시 장래성에 대한 고려가 필요 없다.

Main text**Keyword****Choice text**

Data Collection & Preprocessing

Data Preprocessing

- Dataset Example

	A	B	C	D	E	F	G	H	I
1	Exam_name	Question_Type	MainText	Keyword	Choice_Text_1	Choice_Text_2	Choice_Text_3	Choice_Text_4	
2	건축기사	건축계획	1. 기업체가 자사제품의 홍보, 판매 촉진 등을 위해 제품 및 기업에 관한 자료를 소비자들(전시공간)		① 쇼룸	② 런드리	③ 프로시니엄	④ 인포메이션	
3	건축기사	건축계획	2. 사무소 건축의 실단위 계획 중 개설 시스템에 관한 설명으로 옳지 않은 것은?	개설 시스템	① 공사비가 저렴하다.	② 독립성과 쾌적감이 높다.	③ 방길이에 변화를 줄 수 있다.	④ 방길이에 변화를 줄 수 없다.	
4	건축기사	건축계획	3. 주택단지계획에서 보차분리의 형태 중 평면분리에 해당하지 않는 것은?	방면분리	① T자형	② 루프(loop)	③ 쿨데삭(Cul-de-Sac)	④ 오버브리지(overbridge)	
5	건축기사	건축계획	4. 도서관의 출납 시스템 유형 중 이용자 자유롭게 도서를 꺼낼 수 있으나 열람석으로 기 출납 시스템		① 폐가식	② 반개가식	③ 자유개가식	④ 안전개가식	
6	건축기사	거주계획	5. 단독주택에서 다음과 같은 실들을 각각 적상층 및 적하층에 배치할 경우 가장 바람직한 단독주택		① 상층: 침실, 하층: 침실	② 상층: 부엌, 하층: 욕실	③ 상층: 욕실, 하층: 침실	④ 상층: 욕실, 하층: 부엌	
7	건축기사	건축계획	6. 다음 중 백화점 매장의 기동경계 결정 요소와 가장 거리가 먼 것은?	백화점 매장	① 엘리베이터의 배치방법	② 진열장의 치수와 배치방법	③ 지하주차장 주차방식과	④ 층별 매장 구성과 예상 이용 인원	
8	건축기사	건축계획	7. 학교 운영방식에 관한 설명으로 옳지 않은 것은?	학교 운영방식	① 종합교실형은 추도하고 ② 교과교실형은 교실이 10~30석으로 한글과 한글을 2부다으로 나누				
9	건축기사	건축계획	8. 종합병원에서 클로즈드 시스템(closed system)의 외래진료부에 관한 설명으로 옳지 않은 것은?	클로즈드 시스템	① 내과는 소규모 진료실을 ② 환자의 이용이 편리하도록 ③ 중앙주사실, 회의, 약국 등 ④ 전체병원에 대한 외래진료부의 면적				
10	건축기사	건축계획	9. 공장 건축의 레이아웃(layout)에 관한 설명으로 옳지 않은 것은?	레이아웃	① 세공공업의 데비아웃은 ② 데비아웃은 경대 쌓임과 ③ 쌓임 공업의 데비아웃은 기능이 종합화가				
11	건축기사	건축계획	10. 극장건축의 관련 제설에 관한 설명으로 옳지 않은 것은?	극장건축	① 앤티 룸(anti room)은 ② 그린 룸(green room)은 ③ 배경제작실의 위치는 무대 ④ 의상실은 실의 크기가 1인당 최소 8㎡				
12	건축기사	건축계획	11. 상점의 동선계획에 관한 설명으로 옳지 않은 것은?	동선계획	① 고객동선은 가능한 길게 ② 직원동선은 가능한 짧게 ③ 상품동선과 직원동선은 ④ 고객 출입구와 상품 반입/출출입구는				
13	건축기사	건축계획	12. 건축공간의 치수계획에서 "암박감을 느끼지 않을 만큼의 천장 높이 결정"은 다음 중 치수계획		① 물리적 스케일	② 생리적 스케일	③ 심리적 스케일	④ 입면적 스케일	
14	건축기사	건축계획	13. 고대 그리스 거조문 층 패테오(Pantheon)에 관한 설명으로 옳지 않은 것은?	판테온	① 로툰다 내부는 드럼과 둑 ② 직사각형의 입구 공간은 ③ 드럼 하부는 깊은 니치와 ④ 거대한 들판을 얹은 로툰다와 대형 열주				
15	건축기사	건축계획	14. 극장의 평면형식 중 오픈 스테이지(open stage)형에 관한 설명으로 옳지 않은 것은?	오픈 스테이지	① 연기자가 남측 방향으로 ② 강연, 음악회, 뮤지컬, 연극 ③ 가장 일반적인 극장의 형 ④ 무대와 객석이 동일공간에 있는 것으로				
16	건축기사	건축계획	15. 나무 숲에 밟았는 사무소 건축의 고려 유형은?	사무소 건축	① 편심형	② 독립형	③ 분리형	④ 중심형	
17	건축기사	건축계획	16. 조선시대에 토자형 주택으로 대표되는 서민주택의 지방 유형은?	서민 주택	① 서울지방형	② 남부지방형	③ 중부지방형	④ 함경도지방형	
18	건축기사	건축계획	17. 메ゾ넷형(Maisonette Type) 아파트에 관한 설명으로 옳지 않은 것은?	메조넷형	① 설비, 구조적인 해결이 유 ② 통로가 없는 층의 평면은 ③ 통로가 있는 층의 평면은 ④ 엘리베이터 정지층 및 통로 면적의 감				
19	건축기사	건축계획	18. 고딕 성당에 관한 설명으로 옳지 않은 것은?	고딕 성당	① 중앙집중식 배치를 지배 ② 건축 형태에서 수직성을 ③ 고딕 성당으로는 랭스 성 ④ 수평 방향으로 통일되고 연속적인 공				
20	건축기사	건축계획	19. 단독주택의 평면계획에 관한 설명으로 옳지 않은 것은?	단독주택	① 거실은 평면계획상 통로 ② 현관의 위치는 대지의 형 ③ 부엌은 주택의 서측이나 ④ 노인침실은 일조가 충분하고 전망이				
21	건축기사	건축계획	20. 다음 중 호텔의 성격상 면적에 대한 속박면적의 비가 가장 큰 것은?	속박면적	① 리조트 호텔	② 커머셜 호텔	③ 클럽 하우스	④ 레이던셜 호텔	
22	건축기사	건축시공	21. 벽두께 1.0B, 벽면적 30m² 쌓이에 소요되는 벽돌의 정미량은? (단, 벽돌은 표준형 정미량		① 3900매	② 4095매	③ 4470매	④ 4604매	
23	건축기사	건축시공	22. 석재의 일반적 성질에 관한 설명으로 옳지 않은 것은?	석재	① 석재의 비중은 조암광물 ② 석재의 강도에서 인장강 ③ 석재의 공극률이 클수록 ④ 석재의 강도는 조성결정형이 클수록				
24	건축기사	건축시공	23. Power shovel의 1시간당 추청 물적 작업량을 다음 조건에 따라 구하면?	굴착 작업량	① 67.2 m³/h	② 74.7 m³/h	③ 82.2 m³/h	④ 89.6 m³/h	
25	건축기사	건축시공	24. 도장작업 시 주의사항으로 옳지 않은 것은?	도장작업	① 도료의 적부를 검토하여 ② 도료량을 표준량보다 두 ③ 저온 다습 시에는 작업을 ④ 피막은 각층마다 충분히 건조 경화한				
26	건축기사	건축시공	25. 콘크리트의 내화, 내열성에 관한 설명으로 옳지 않은 것은?	콘크리트	① 콘크리트의 내화, 내열성 ② 콘크리트는 내화성이 우 ③ 철근콘크리트 부재의 내화 ④ 화재를 입은 콘크리트의 탄산화 속도				
27	건축기사	건축시공	26. 아스팔트 방수공사에서 아스팔트 프라이mer를 사용하는 가장 중요한 이유는?	아스팔트	① 콘크리트 면의 습기 제거 ② 방수층의 습기 침입 방지 ③ 콘크리트 면과 아스팔트 ④ 콘크리트 밀바닥의 균열방지				
28	건축기사	건축시공	27. 콘크리트 배합에 직접적으로 영향을 주는 요소가 아닌 것은?	콘크리트	① 단위수량	② 물-결합재 비	③ 철근의 품질	④ 골재의 입도	
29	건축기사	건축시공	28. 철근, 볼트 등 건축용 강재의 재료시험 항목에서 일반적으로 제외되는 항목은?	건축용 강재	① 압축강도시험	② 인장강도시험	③ 굽힘시험	④ 연신율시험	
30	건축기사	건축시공	29. 발주자에 의한 현장관리로 볼 수 없는 것은?	현장관리	① 착공신고	② 하도급계약	③ 현장회의 운영	④ 클레임 관리	
31	건축기사	건축시공	30. 어스앵커 공법에 관한 설명으로 옳지 않은 것은?	어스앵커	① 베팅대가 없어 굴착공간 ② 인접한 구조물의 기초나 ③ 대형기계의 반입이 용이 ④ 시공 후 검사가 어렵다.				

Data Collection & Preprocessing

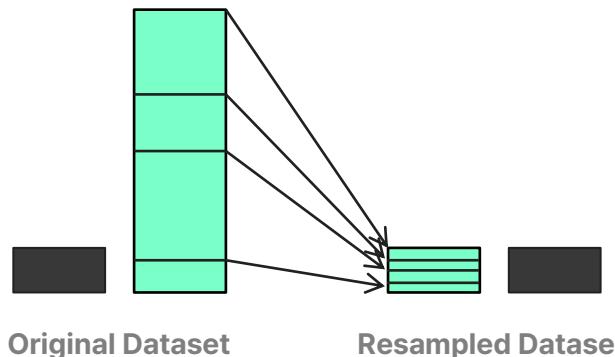
Data Augmentation

- Synthetic Minority Oversampling Technique (SMOTE)

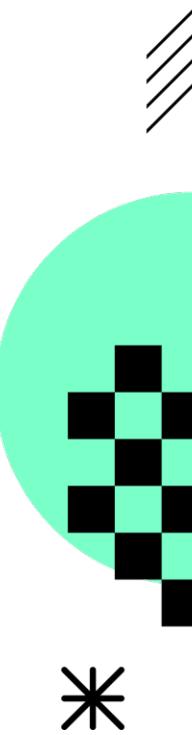
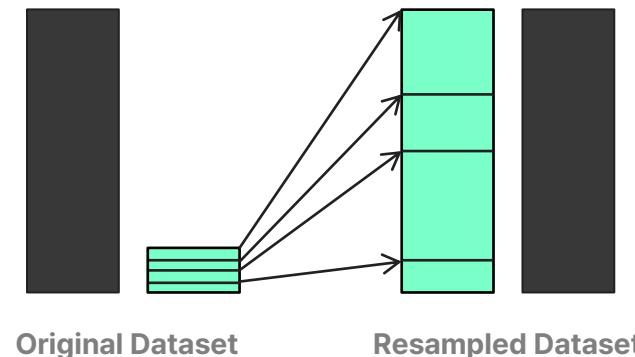
- A problem with imbalanced classification is that there are too few examples of the minority class for a model to effectively learn the decision boundary.
- SMOTE was used only for train data in the choice text and main text.

Label	Number_origin	Number_SMOTE
0	2690	2690
1	2531	2690
2	2155	2690
3	1868	2690
4	1354	2690

Undersampling

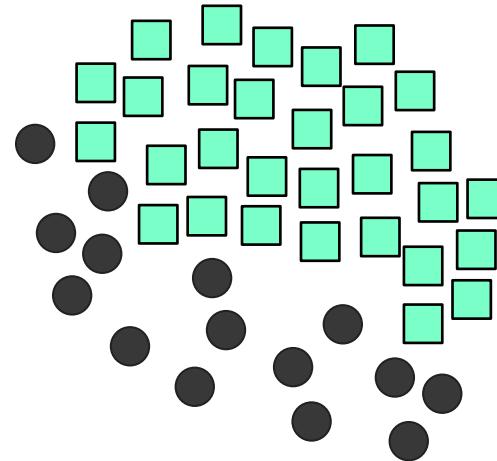


Oversampling



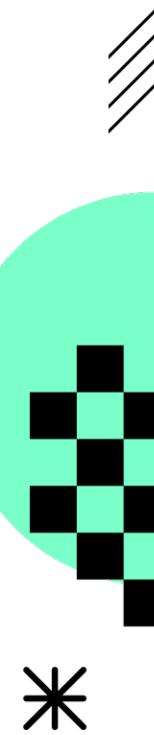
Data Collection & Preprocessing

Data Augmentation



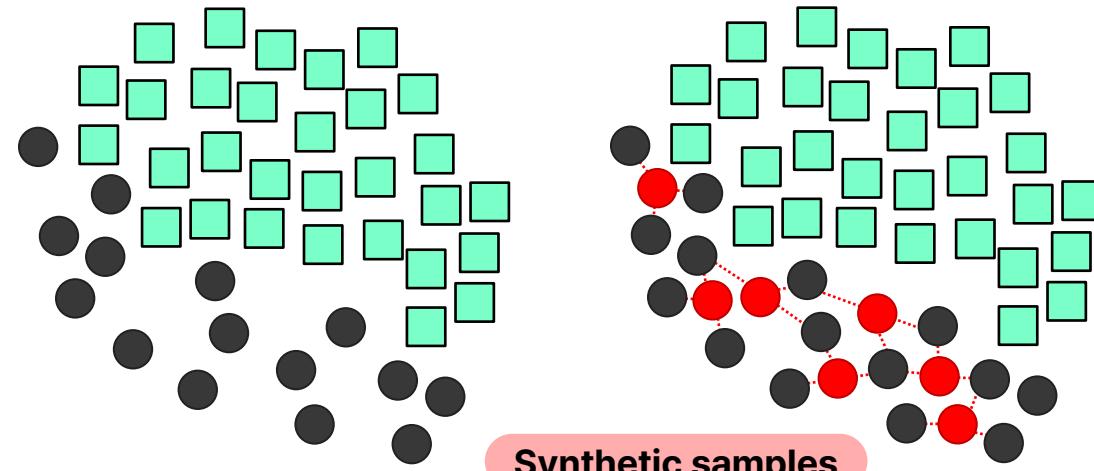
Label Original Dataset

0	2690
1	2531
2	2155
3	1868
4	1354



Data Collection & Preprocessing

Data Augmentation

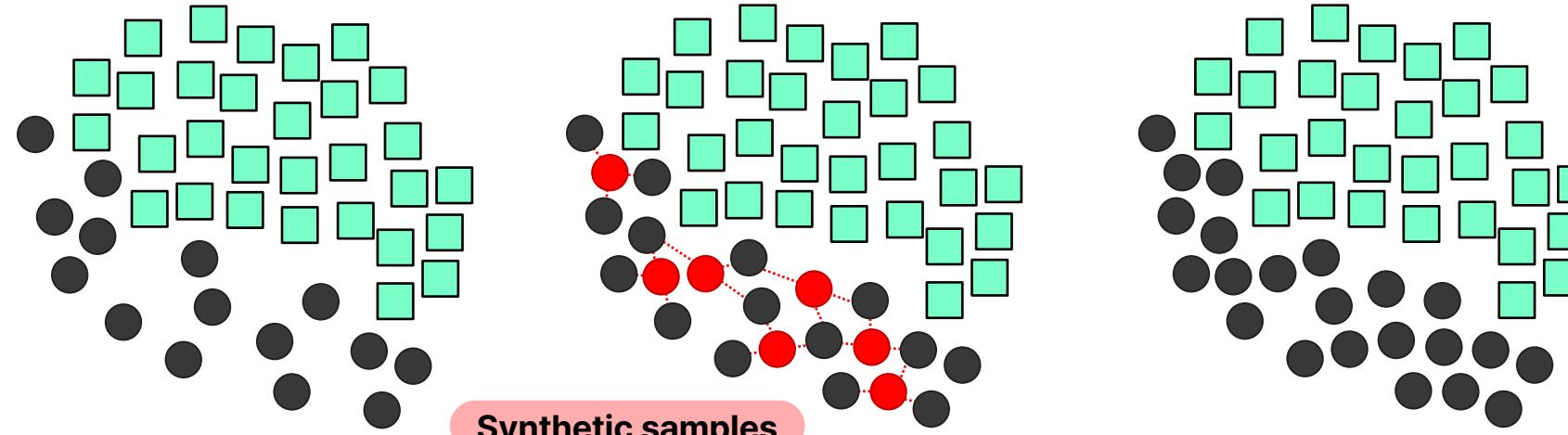


Label	Original Dataset	Generating Samples
0	2690	
1	2531	+ 159
2	2155	+ 535
3	1868	+ 822
4	1354	+ 1336

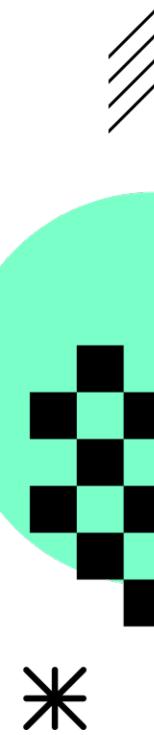


Data Collection & Preprocessing

Data Augmentation

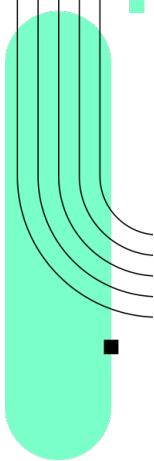
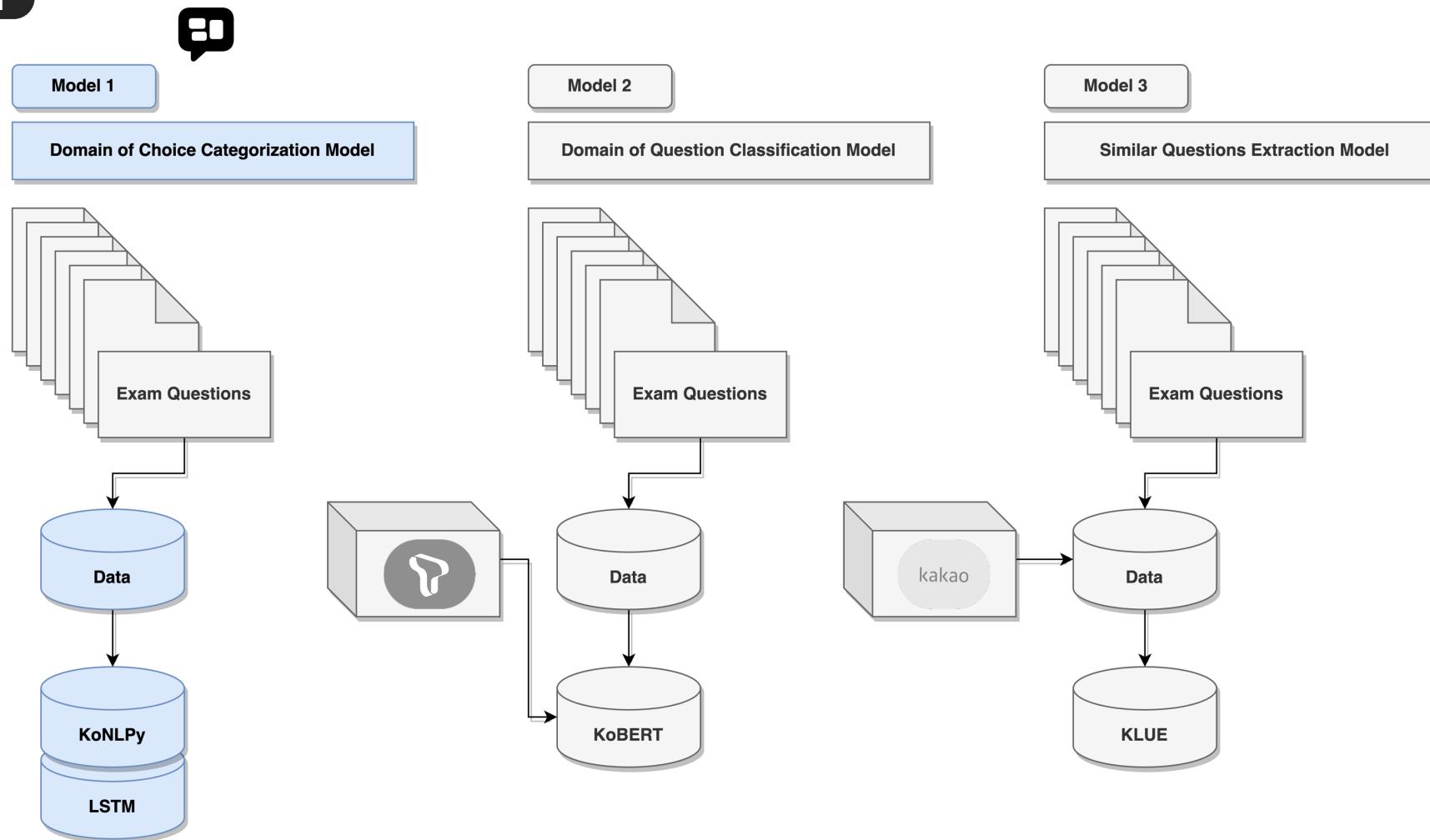


Label	Original Dataset	Generating Samples	Resampled Dataset
0	2690		2690
1	2531	+ 159	2690
2	2155	+ 535	2690
3	1868	+ 822	2690
4	1354	+ 1336	2690



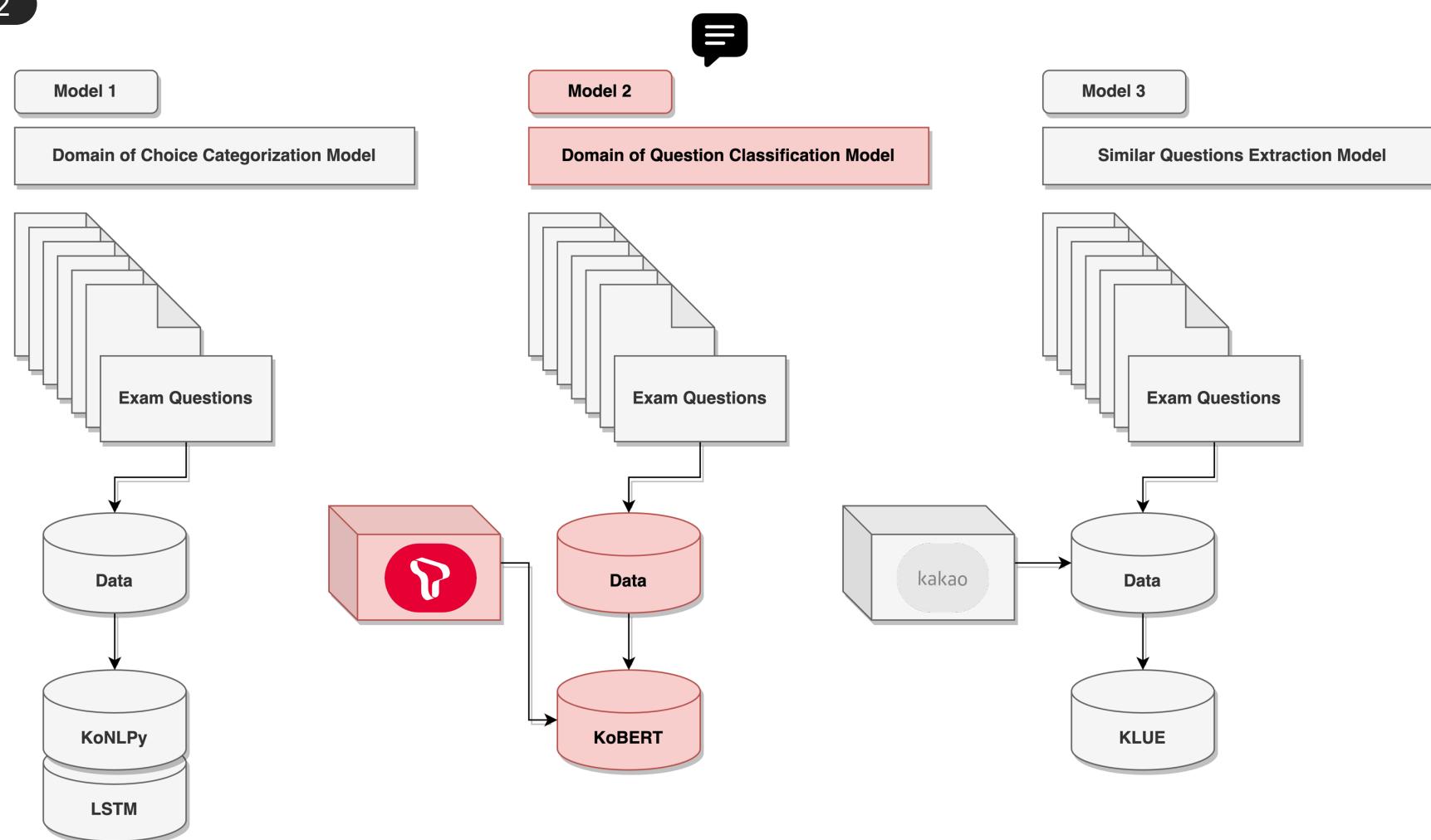
Model Flow

Model 1



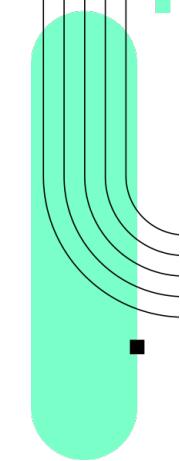
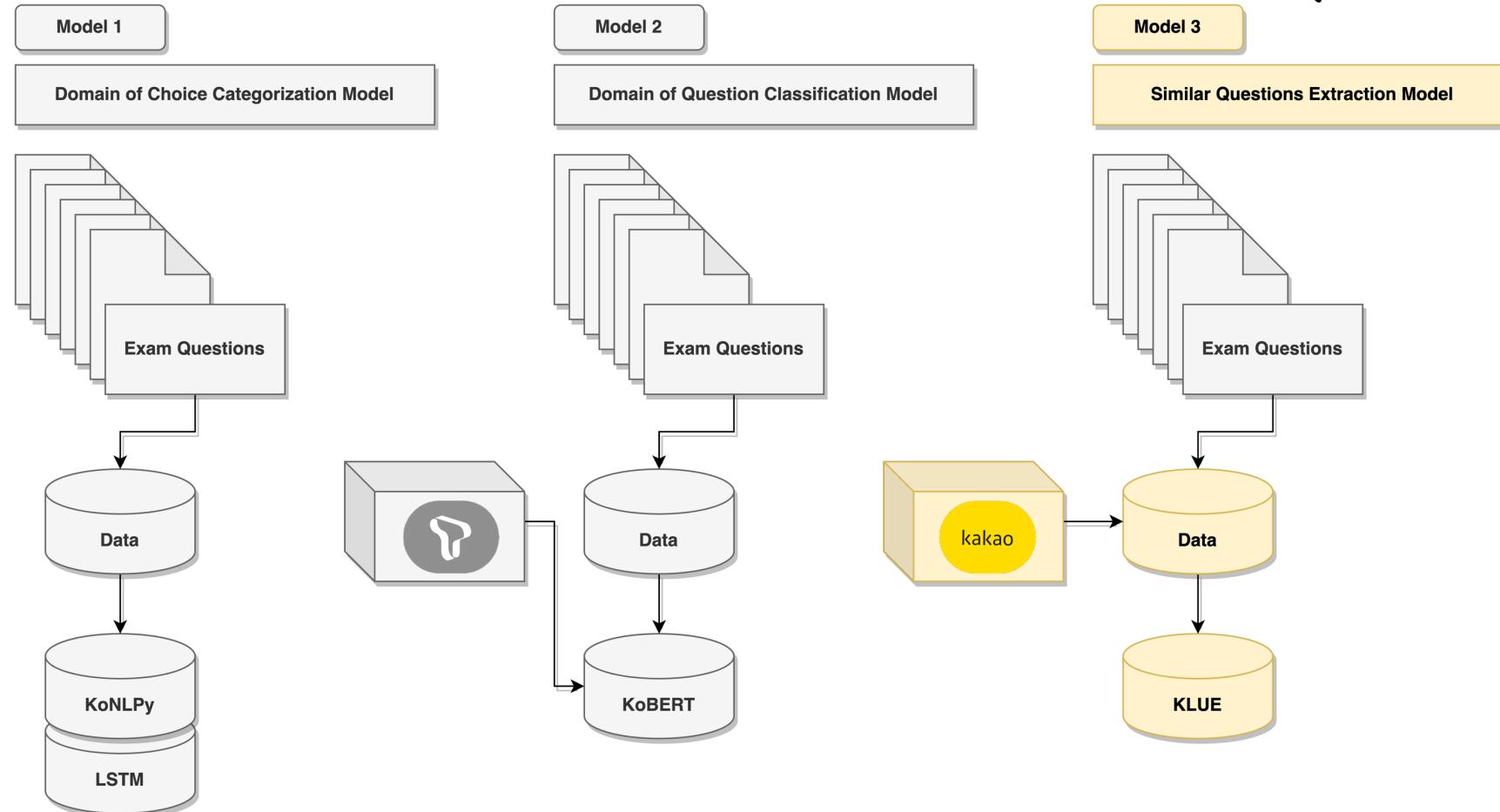
Model Flow

Model 2



Model Flow

Model 3

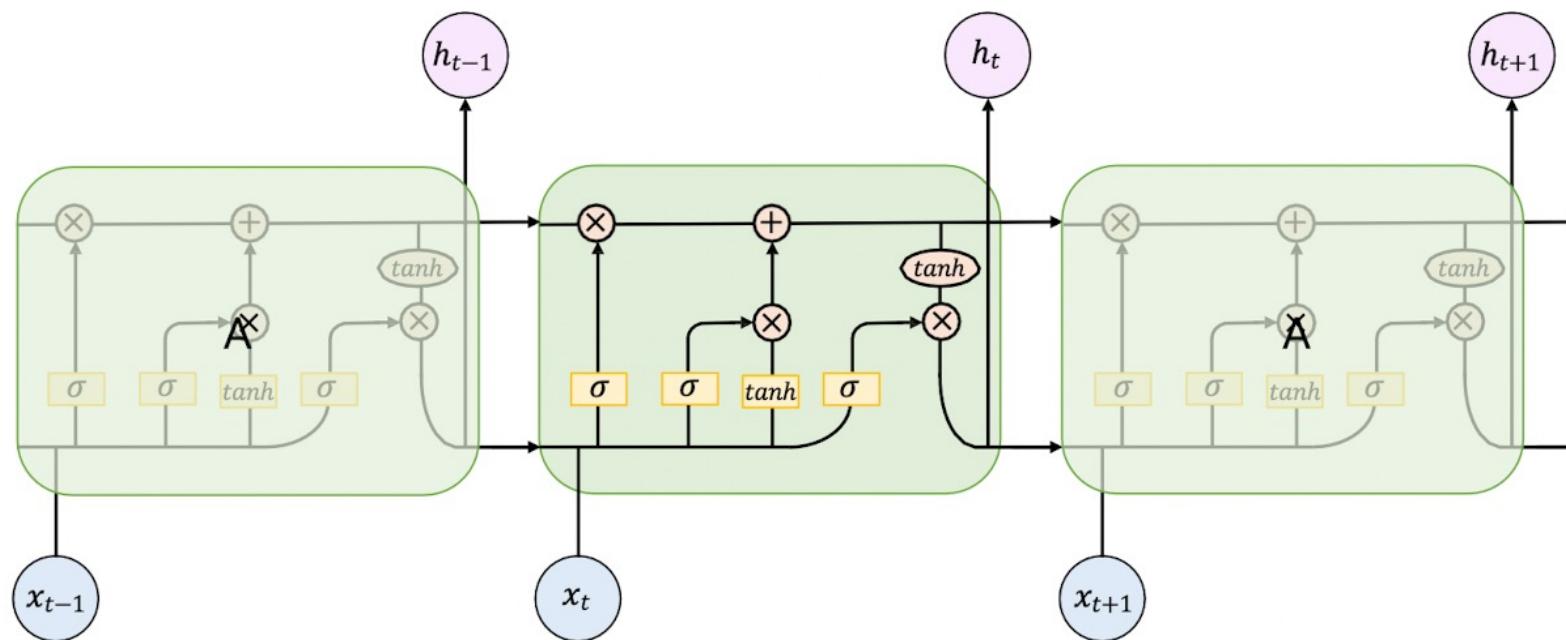


Model 1



Domain of Choice Categorization Model

- **LSTM** : Long Short-Term Memory
- A model that solves the problem of gradient vanishing near the input layer in the RNN model. So, it shows improved performance.



LSTM Model

Model 1



Domain of Choice Categorization Model

1. Install & Import required packages

```
▶ bash <(curl -s https://raw.githubusercontent.com/konlpy/konlpy/master/scripts/mecab.sh)

[ ] !pip install -q tweepy==3.10

try:
    import konlpy
except:
    !pip install -q konlpy
    import konlpy

[ ] import pandas as pd
import numpy as np
import re

from tensorflow.keras.preprocessing.text import Tokenizer
from tensorflow.keras.preprocessing.sequence import pad_sequences
from konlpy.tag import Okt

import matplotlib as mpl
import matplotlib.pyplot as plt

from imblearn.over_sampling import SMOTE
```

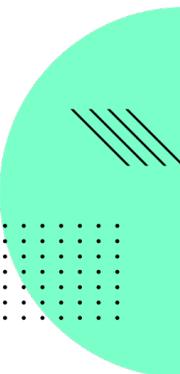
2. Import dataset

```
[ ] data_raw = pd.read_excel("Dataset.xlsx", engine = "openpyxl")

[ ] data1 = data_raw[['Testname', 'ChoiceText1']]
data2 = data_raw[['Testname', 'ChoiceText2']]
data3 = data_raw[['Testname', 'ChoiceText3']]
data4 = data_raw[['Testname', 'ChoiceText4']]
data1.columns = ['Testname', 'ChoiceText']
data2.columns = ['Testname', 'ChoiceText']
data3.columns = ['Testname', 'ChoiceText']
data4.columns = ['Testname', 'ChoiceText']

[ ] data = pd.concat([data1,data2,data3,data4], axis=0)

[ ] data = data.reset_index(drop=True,inplace=False)
```



Model 1



Domain of Choice Categorization Model

3. Data Preprocessing

```
[12] data = data.dropna()

[10] data.loc[(data['Testname'] == "건축기사"), 'Testname'] = 0
     data.loc[(data['Testname'] == "대기환경기사"), 'Testname'] = 1
     data.loc[(data['Testname'] == "산업안전기사"), 'Testname'] = 2
     data.loc[(data['Testname'] == "소방설비기사"), 'Testname'] = 3
     data.loc[(data['Testname'] == "정보처리기사"), 'Testname'] = 4

[ ] def cleanText(readData):
    text = re.compile('^[^A-Za-z0-9가-힣]+')
    result = text.sub(' ', readData)
    return result

[ ] for i in range(len(data)):
    data.iloc[i]['ChoiceText'] = cleanText(data.iloc[i]['ChoiceText'])
```

Sample of Preprocessed Data

Testname	ChoiceText
0	쇼룸
1	공사비가 저렴하다
2	자형
3	폐가식
4	상층 침실 하층 침실
...	...
8841	순으로 반복
8842	키 분배가 용이하고 관리해야 할 키 개수가 적다
8843	년대까지 가장 많이 적용되었던 소프트웨어 개발 방법
8844	특정 하드웨어에 종속되어 특화된 업무를 서비스하기에 적합하다
8845	조건이 복합되어 있는 곳의 처리를 시각적으로 명확히 식별하는데 적합하다

Model 1



Domain of Choice Categorization Model

4. Split data

```
[20] x = data['ChoiceText']
y = data['Testname']

[21] from sklearn.model_selection import train_test_split
x_train, x_test, y_train, y_test = train_test_split(x,y,test_size=0.1,random_state=10)

[22] train_data = pd.DataFrame({'document':x_train,
                             'label':y_train})

[23] test_data = pd.DataFrame({'document':x_test,
                             'label':y_test})
```

5. Word Tokenization

```
[24] okt = Okt()

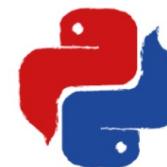
[25] X_train = []
for sentence in train_data['document']:
    temp_X = okt.morphs(sentence, stem=True) # Okt() 이용해서 토큰화
    X_train.append(temp_X)

[26] X_test = []
for sentence in test_data['document']:
    temp_X = okt.morphs(sentence, stem=True) # 토큰화
    X_test.append(temp_X)
```

- **Konlpy** : Python package for Korean natural language processing.
- This is a Python library used for Korean morpheme analysis.

konlpy/konlpy

Python package for Korean natural language processing.



Model 1



Domain of Choice Categorization Model

6. word vectorization

```
[46] tokenizer = Tokenizer()
     tokenizer.fit_on_texts(X_train)

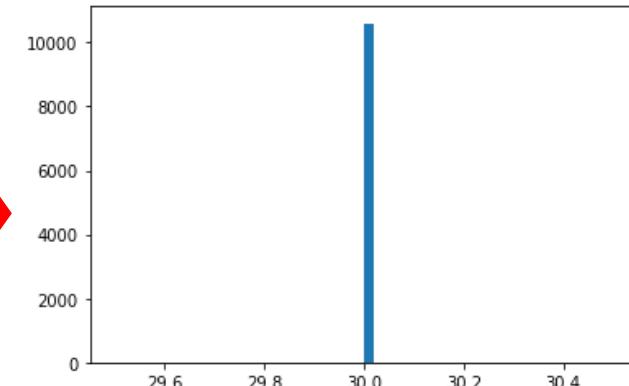
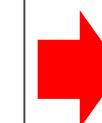
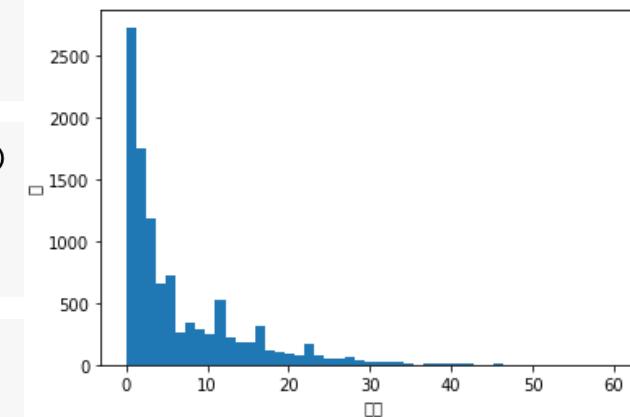
[47] total_count = len(tokenizer.word_index) # 단어의 수
     vocab_size = total_count
     print('단어 집합의 크기 :',vocab_size)

[48] tokenizer = Tokenizer(vocab_size, oov_token = 'OOV')
     tokenizer.fit_on_texts(X_train)
     X_train = tokenizer.texts_to_sequences(X_train)
     X_test = tokenizer.texts_to_sequences(X_test)

[49] y_train = np.array(train_data['label'])
     y_test = np.array(test_data['label'])

[ ] max_len = 50
    X_train = pad_sequences(X_train, maxlen = max_len)
    X_test = pad_sequences(X_test, maxlen = max_len)
```

Data length distribution



Model 1



Domain of Choice Categorization Model

7. Apply SMOTE

```
[ ] smote = SMOTE(random_state=0)
x_train_over,y_train_over = getset_descriptor: X_train.shape
[ ] y_train_over_fin = pd.get_dummies(y_train_over).values
```

Data balance

Label	Number_origin	Number_SMOTE
0	2690	2690
1	2531	2690
2	2155	2690
3	1868	2690
4	1354	2690

Model 1



Domain of Choice Categorization Model

8. LSTM Model

```
[36] from tensorflow.keras.layers import Embedding, Dense, LSTM, Dropout
    from tensorflow.keras.models import Sequential
    from tensorflow.keras.models import load_model
    from tensorflow.keras.callbacks import EarlyStopping, ModelCheckpoint

[37] model1 = Sequential()

    model1.add(Embedding(vocab_size, 30))
    model1.add(LSTM(128))
    model1.add(Dropout(0.2))
    model1.add(Dense(16))
    model1.add(Dropout(0.2))
    model1.add(Dense(5, activation='softmax'))
```

Model: "sequential_1"

Layer (type)	Output Shape	Param #
<hr/>		
embedding_1 (Embedding)	(None, None, 50)	447850
lstm_2 (LSTM)	(None, None, 128)	91648
dropout_3 (Dropout)	(None, None, 128)	0
lstm_3 (LSTM)	(None, 128)	131584
dropout_4 (Dropout)	(None, 128)	0
dense_2 (Dense)	(None, 32)	4128
dropout_5 (Dropout)	(None, 32)	0
dense_3 (Dense)	(None, 5)	165

Total params: 675,375

Trainable params: 675,375

Non-trainable params: 0



Model 1



Domain of Choice Categorization Model

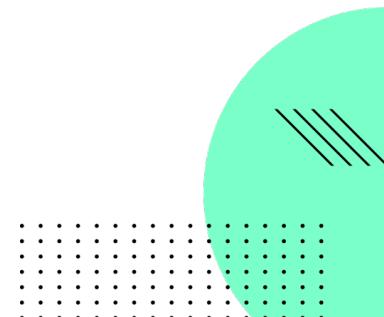
8. LSTM Model

```
[118] es = EarlyStopping(monitor='val_loss', mode='min', verbose=1, patience=4)
      mc = ModelCheckpoint('best_model.h5', monitor='val_acc', mode='max', verbose=1, save_best_only=True)

[119] modell.compile(optimizer='adam', loss='categorical_crossentropy', metrics=['acc'])

history = modell.fit(x_train_over, y_train_over_fin, epochs=10, callbacks=[es, mc], batch_size=60, validation_split=0.3)

Epoch 1/10
155/157 [=====>.] - ETA: 0s - loss: 1.5037 - acc: 0.3267
Epoch 1: val_acc improved from -inf to 0.22305, saving model to best_model.h5
157/157 [=====] - 8s 25ms/step - loss: 1.5019 - acc: 0.3287 - val_loss: 1.7642 - val_acc: 0.2230
Epoch 2/10
155/157 [=====>.] - ETA: 0s - loss: 0.9869 - acc: 0.6133
Epoch 2: val_acc improved from 0.22305 to 0.37918, saving model to best_model.h5
157/157 [=====] - 2s 13ms/step - loss: 0.9855 - acc: 0.6143 - val_loss: 1.9197 - val_acc: 0.3792
Epoch 3/10
155/157 [=====>.] - ETA: 0s - loss: 0.5701 - acc: 0.8085
Epoch 3: val_acc did not improve from 0.37918
157/157 [=====] - 2s 13ms/step - loss: 0.5685 - acc: 0.8090 - val_loss: 2.7367 - val_acc: 0.3532
Epoch 4/10
156/157 [=====>.] - ETA: 0s - loss: 0.3555 - acc: 0.8809
Epoch 4: val_acc improved from 0.37918 to 0.39281, saving model to best_model.h5
157/157 [=====] - 2s 13ms/step - loss: 0.3545 - acc: 0.8811 - val_loss: 2.9250 - val_acc: 0.3928
Epoch 5/10
155/157 [=====>.] - ETA: 0s - loss: 0.2658 - acc: 0.9070
Epoch 5: val_acc improved from 0.39281 to 0.39306, saving model to best_model.h5
157/157 [=====] - 2s 13ms/step - loss: 0.2662 - acc: 0.9071 - val_loss: 3.3972 - val_acc: 0.3931
Epoch 5: early stopping
```



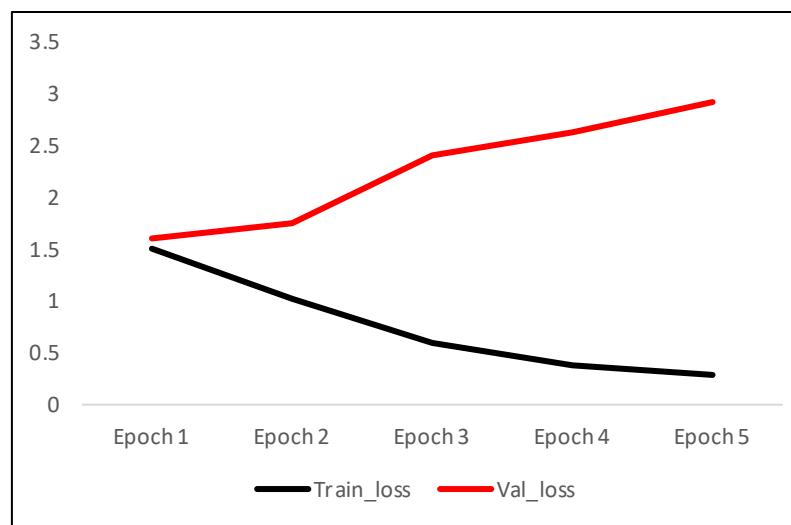
Model 1



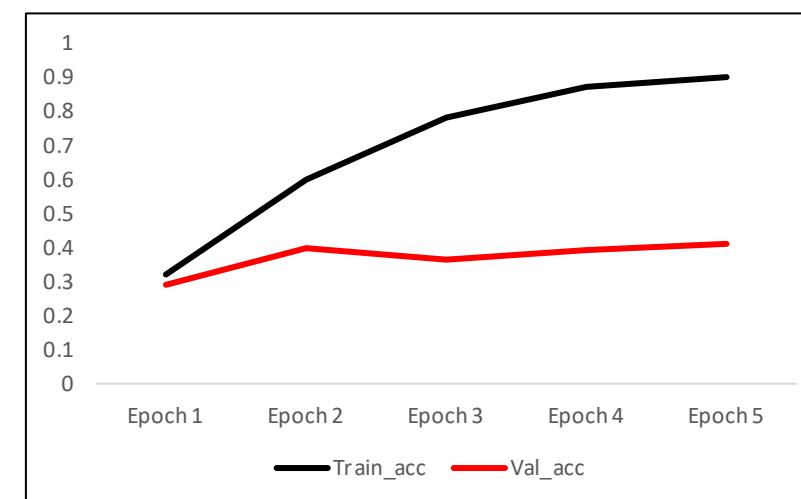
Domain of Choice Categorization Model

Result of model 1

Loss

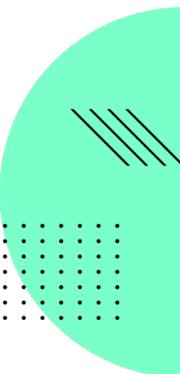


Acc



Test Acc

0.6749

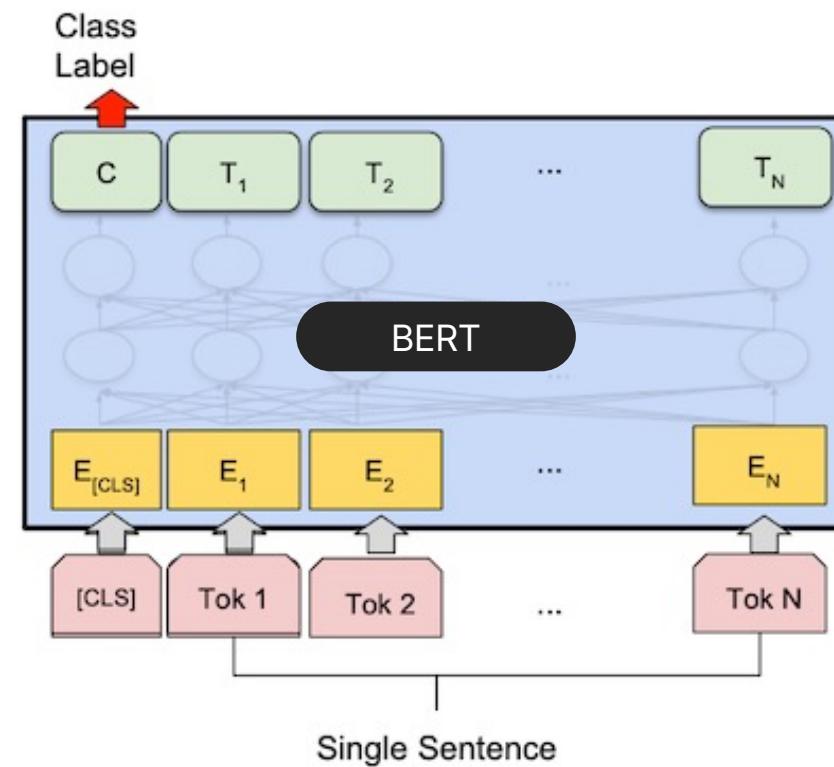


Model 2



Domain of Question Classification Model

- **BERT** : Pre-training of Deep Bidirectional Transformers for Language Understanding
- BERT is a language model that can improve the performance of a specific task through pre-training Embedding before it is done



Model 2



Domain of Question Classification Model

- **KoBERT** : Learn a large corpus consisting of millions of Korean sentences collected from Wikipedia, news, etc
- Data-based tokenization techniques are applied to reflect the characteristics of irregular language changes in the Korean language



TrainSet

Data source	Sentence	Token
Korean WIKI	5M	54M

Vocabulary

- Size : 8,002
- SentencePiece: Korean WIKI based Tokenizer
- Less number of parameters

Model 2



Domain of Question Classification Model

1. Install & Import required packages

```
[1] !pip install gluonnlp pandas tqdm
!pip install mxnet
!pip install sentencepiece==0.1.91
!pip install transformers==4.8.2
!pip install torch

[2] # 실행 후 필요시 런타임 재 실행
!pip install torch --upgrade

[3] !pip install git+https://git@github.com/SKTBrain/KoBERT.git@master

[4] !pip install 'git+https://github.com/SKTBrain/KoBERT.git#egg=kobert_tokenizer&subdirectory=kobert_hf'
```

```
▶ import pandas as pd
import re

from kobert.pytorch_kobert import get_pytorch_kobert_model
from kobert.utils import get_tokenizer

import torch
from torch import nn
import torch.nn.functional as F
import torch.optim as optim
from torch.utils.data import Dataset, DataLoader
import gluonnlp as nlp
import numpy as np
from tqdm import tqdm, tqdm_notebook
import pandas as pd

#transformers
from transformers import AdamW
from transformers.optimization import get_cosine_schedule_with_warmup
from transformers import BertModel
```

2. Import data

```
[12] data_raw = pd.read_excel("Dataset.xlsx", engine = "openpyxl")

[13] data = data_raw[['Testname', 'MainText']]
```

3. Data Preprocessing

```
[29] data.loc[(data['Testname'] == "건축기사"), 'Testname'] = 0
data.loc[(data['Testname'] == "대기환경기사"), 'Testname'] = 1
data.loc[(data['Testname'] == "산업안전기사"), 'Testname'] = 2
data.loc[(data['Testname'] == "소방설비기사"), 'Testname'] = 3
data.loc[(data['Testname'] == "정보처리기사"), 'Testname'] = 4

[30] for i in range(len(data)):
    if data.iloc[i]['MainText'][::2] in ['1.', '2.', '3.', '4.', '5.', '6.', '7.', '8.', '9.']:
        data.iloc[i]['MainText'] = data.iloc[i]['MainText'][3:]
    elif data.iloc[i]['MainText'][::4] in ['100.']:
        data.iloc[i]['MainText'] = data.iloc[i]['MainText'][5:]
    else:
        data.iloc[i]['MainText'] = data.iloc[i]['MainText'][4:]

[31] def cleanText(readData):
    text = re.compile('[^A-Za-z0-9가-힣]+')
    result = text.sub(' ', readData)
    return result

▶ for i in range(len(data)):
    data.iloc[i]['MainText'] = cleanText(data.iloc[i]['MainText'])
```

Model 2



Domain of Question Classification Model

4. Split data

```
[19] data_list = []
    for ques, label in zip(data['MainText'], data['Testname']):
        data = []
        data.append(ques)
        data.append(str(label))

        data_list.append(data)

[20] from sklearn.model_selection import train_test_split

dataset_train, dataset_test = train_test_split(data_list, test_size=0.2, shuffle=True, random_state=34)
```

5. Word embedding

```
[21] tokenizer = get_tokenizer()
    bertmodel, vocab = get_pytorch_kobert_model()

[22] class BERTDataset(Dataset):
    def __init__(self, dataset, sent_idx, label_idx, bert_tokenizer,vocab, max_len,
                 pad, pair):

        transform = nlp.data.BERTSentenceTransform(
            bert_tokenizer, max_seq_length=max_len,vocab=vocab, pad=pad, pair=pair)

        self.sentences = [transform([i[sent_idx]]) for i in dataset]
        self.labels = [np.int32(i[label_idx]) for i in dataset]

    def __getitem__(self, i):
        return (self.sentences[i] + (self.labels[i],))

    def __len__(self):
        return (len(self.labels))
```

```
[23] # Setting parameters
max_len = 64
batch_size = 64
warmup_ratio = 0.1
num_epochs = 20
max_grad_norm = 1
log_interval = 200
learning_rate = 5e-5
```

```
[24] tok = nlp.data.BERTSPTokenizer(tokenizer, vocab, lower=False)
    data_train = BERTDataset(dataset_train, 0, 1, tok, vocab, max_len, True, False)
    data_test = BERTDataset(dataset_test,0, 1, tok, vocab, max_len, True, False)

[25] train_dataloader = torch.utils.data.DataLoader(data_train, batch_size=batch_size, num_workers=5)
    test_dataloader = torch.utils.data.DataLoader(data_test, batch_size=batch_size, num_workers=5)
```

Model 2



Domain of Question Classification Model

6. Import pre-trained KoBERT

```
class BERTClassifier(nn.Module):
    def __init__(self,
                 bert,
                 hidden_size = 768,
                 num_classes=5,    ##클래스 수 조정##
                 dr_rate=None,
                 params=None):
        super(BERTClassifier, self).__init__()
        self.bert = bert
        self.dr_rate = dr_rate

        self.classifier = nn.Linear(hidden_size , num_classes)
        if dr_rate:
            self.dropout = nn.Dropout(p=dr_rate)

    def gen_attention_mask(self, token_ids, valid_length):
        attention_mask = torch.zeros_like(token_ids)
        for i, v in enumerate(valid_length):
            attention_mask[i][:v] = 1
        return attention_mask.float()

    def forward(self, token_ids, valid_length, segment_ids):
        attention_mask = self.gen_attention_mask(token_ids, valid_length)

        _, pooler = self.bert(input_ids = token_ids, token_type_ids = segment_ids.long(),
                              attention_mask = attention_mask.float().to(token_ids.device),return_dict=False)
        if self.dr_rate:
            out = self.dropout(pooler)
        return self.classifier(out)
```

```
#BERT 모델 불러오기
model = BERTclassifier(bertmodel, dr_rate=0.5).to(device)

#optimizer와 schedule 설정
no_decay = ['bias', 'LayerNorm.weight']
optimizer_grouped_parameters = [
    {'params': [p for n, p in model.named_parameters() if not any(nd in n for nd in no_decay)], 'weight_decay': 0.01},
    {'params': [p for n, p in model.named_parameters() if any(nd in n for nd in no_decay)], 'weight_decay': 0.0}
]

optimizer = AdamW(optimizer_grouped_parameters, lr=learning_rate)
loss_fn = nn.CrossEntropyLoss() # 다중분류를 위한 대표적인 loss func

t_total = len(train_dataloader) * num_epochs
warmup_step = int(t_total * warmup_ratio)

scheduler = get_cosine_schedule_with_warmup(optimizer, num_warmup_steps=warmup_step, num_training_steps=t_total)

#정확도 측정을 위한 함수 정의
def calc_accuracy(X,Y):
    max_vals, max_indices = torch.max(X, 1)
    train_acc = (max_indices == Y).sum().data.cpu().numpy()/max_indices.size()[0]
    return train_acc

train_dataloader
```

Model 2



Domain of Question Classification Model

7. Fine-tuning

```
[ ] train_history=[]
test_history=[]
loss_history=[]
for e in range(num_epochs):
    train_acc = 0.0
    test_acc = 0.0
    model.train()
    for batch_id, (token_ids, valid_length, segment_ids, label) in enumerate(tqdm_notebook(train_dataloader)):
        optimizer.zero_grad()
        token_ids = token_ids.long().to(device)
        segment_ids = segment_ids.long().to(device)
        valid_length= valid_length
        label = label.long().to(device)
        out = model(token_ids, valid_length, segment_ids)

        #print(label.shape,out.shape)
        loss = loss_fn(out, label)
        loss.backward()
        torch.nn.utils.clip_grad_norm_(model.parameters(), max_grad_norm)
        optimizer.step()
        scheduler.step() # Update learning rate schedule
        train_acc += calc_accuracy(out, label)
        if batch_id % log_interval == 0:
            print("epoch {} batch id {} loss {} train acc {}".format(e+1, batch_id+1, loss.data.cpu().numpy(),
                                                                    train_acc / (batch_id+1)))
            train_history.append(train_acc / (batch_id+1))
            loss_history.append(loss.data.cpu().numpy())
    print("epoch {} train acc {}".format(e+1, train_acc / (batch_id+1)))
#train_history.append(train_acc / (batch_id+1))
```

```
model.eval()
for batch_id, (token_ids, valid_length, segment_ids, label) in enumerate(tqdm_notebook(test_dataloader)):
    token_ids = token_ids.long().to(device)
    segment_ids = segment_ids.long().to(device)
    valid_length= valid_length
    label = label.long().to(device)
    out = model(token_ids, valid_length, segment_ids)
    test_acc += calc_accuracy(out, label)
print("epoch {} test acc {}".format(e+1, test_acc / (batch_id+1)))
test_history.append(test_acc / (batch_id+1))
```

Model 2



Domain of Question Classification Model

8. Result of Model2

```
100%          11/11 [00:19<00:00, 1.47s/it]
epoch 1 test acc 0.7501136363636364
100%          44/44 [03:20<00:00, 3.45s/it]
epoch 2 batch id 1 loss 0.9055153131484985 train acc 0.734375
epoch 2 train acc 0.8178267045454546
100%          11/11 [00:21<00:00, 1.46s/it]
epoch 2 test acc 0.8758522727272727
100%          44/44 [03:20<00:00, 3.54s/it]
epoch 3 batch id 1 loss 0.267633318901062 train acc 0.953125
epoch 3 train acc 0.9034090909090909
100%          11/11 [00:21<00:00, 1.49s/it]
epoch 3 test acc 0.8985795454545454
100%          44/44 [03:16<00:00, 3.33s/it]
epoch 4 batch id 1 loss 0.1269356608390808 train acc 0.953125
epoch 4 train acc 0.94921875
100%          11/11 [00:20<00:00, 1.42s/it]
epoch 4 test acc 0.9146022727272727
100%          44/44 [03:17<00:00, 3.68s/it]
epoch 5 batch id 1 loss 0.07583899050951004 train acc 1.0
epoch 5 train acc 0.9676846590909091
100%          11/11 [00:20<00:00, 1.45s/it]
epoch 5 test acc 0.9146022727272727
```

Test accuracy

About 0.92

Model 3



Similar Questions Extraction Model

- KLUE: Korean Language Understanding Evaluation
- KAKAO Dataset: KLUE STS (for Train), KorSTS (for Test)

KLUE

[Home](#) [Participants](#) [Paper](#) [Task](#) [Leaderboard](#) [Issue Report](#)

KLUE

Korean Language Understanding Evaluation

Korean Language Understanding Evaluation (KLUE) benchmark is a series of datasets to evaluate natural language understanding capability of Korean language models. KLUE consists of 8 diverse and representative tasks, which are accessible to anyone without any restrictions. With ethical considerations in mind, we deliberately design annotation guidelines to obtain unambiguous annotations for all datasets. Furthermore, we build an evaluation system and carefully choose evaluations metrics for every task, thus establishing fair comparison across Korean language models.

genre	filename	year	id	score	sentence1	sentence2
main-captions	MSRvid	2012test	1	5.000	비행기가 이륙하고 있다.	비행기가 이륙하고 있다.
main-captions	MSRvid	2012test	4	3.800	한 남자가 큰 플루트를 연주하고 있다.	남자가 플루트를 연주하고 있다.
main-captions	MSRvid	2012test	5	3.800	한 남자가 피자에 치즈를 뿌려놓고 있다.	한 남자가 구운 피자에 치즈 조각을 뿌려놓고 있다.
main-captions	MSRvid	2012test	6	2.600	세 남자가 체스를 하고 있다.	두 남자가 체스를 하고 있다.
main-captions	MSRvid	2012test	9	4.250	한 남자가 첼로를 연주하고 있다.	자리에 앉은 남자가 첼로를 연주하고 있다.
main-captions	MSRvid	2012test	11	4.250	몇몇 남자들이 싸우고 있다.	두 남자가 싸우고 있다.

Model 3



Similar Questions Extraction Model

1. Install & Import required packages

```
[ ] !pip install sentence-transformers datasets
[ ]
import pandas as pd
import numpy as np

import math
import logging
from datetime import datetime
import re

import torch
from torch.utils.data import DataLoader
from datasets import load_dataset
from sentence_transformers import SentenceTransformer, LoggingHandler, losses, models, util
from sentence_transformers.evaluation import EmbeddingSimilarityEvaluator
from sentence_transformers.readers import InputExample
```

2. Initialize logger for time check

```
[ ] logging.basicConfig(
    format="%(asctime)s - %(message)s",
    datefmt="%Y-%m-%d %H:%M:%S",
    level=logging.INFO,
    handlers=[LoggingHandler()],
)
```

3. Import pre-learning and embedding models

```
[ ] model_name = "klue roberta-base"
[ ]
embedding_model = models.Transformer(model_name)

[ ] pooler = models.Pooling(
    embedding_model.get_word_embedding_dimension(),
    pooling_mode_mean_tokens=True,
    pooling_mode_cls_token=False,
    pooling_mode_max_tokens=False,
)
[ ] model = SentenceTransformer(modules=[embedding_model, pooler])
```

Model 3



Similar Questions Extraction Model

4. Import dataset

```
[ ] datasets = load_dataset("klue", "sts")

[ ] testsets = load_dataset("kor_nlu", "sts")

[ ] train_samples = []
dev_samples = []
test_samples = []

# KLUE STS 내 훈련, 검증 데이터 예제 변환
for phase in ["train", "validation"]:
    examples = datasets[phase]

    for example in examples:
        score = float(example["labels"]["label"]) / 5.0 # 0.0 ~ 1.0 스케일로 유사도 정규화

        inp_example = InputExample(
            texts=[example["sentence1"], example["sentence2"]],
            label=score,
        )

        if phase == "validation":
            dev_samples.append(inp_example)
        else:
            train_samples.append(inp_example)
```

```
# KorSTS 내 테스트 데이터 예제 변환
for example in testsets["test"]:
    score = float(example["score"]) / 5.0

    if example["sentence1"] and example["sentence2"]:
        inp_example = InputExample(
            texts=[example["sentence1"], example["sentence2"]],
            label=score,
        )

    test_samples.append(inp_example)
```

Model 3



Similar Questions Extraction Model

5. Model learning

```
[ ] train_batch_size = 32  
  
[ ] train_dataloader = DataLoader(  
    train_samples,  
    shuffle=True,  
    batch_size=train_batch_size,  
)  
train_loss = losses.CosineSimilarityLoss(model=model)  
  
[ ] num_epochs = 30  
  
[ ] evaluator = EmbeddingSimilarityEvaluator.from_input_examples(  
    dev_samples,  
    name="sts-dev",  
)
```

```
[ ] warmup_steps = math.ceil(len(train_dataloader) * num_epochs)  
logging.info(f"Warmup-steps: {warmup_steps}")  
  
▶ model.fit(  
    train_objectives=[(train_dataloader, train_loss)],  
    evaluator=evaluator,  
    epochs=num_epochs,  
    evaluation_steps=1000,  
    warmup_steps=warmup_steps,  
)
```

Model 3



Similar Questions Extraction Model

6. Observation of the result

```
[ ] data = pd.read_excel("Dataset.xlsx", engine = "openpyxl")

[ ] def cleanText(readData):
    text = re.compile('[^A-Za-z0-9가-힣]+')
    result = text.sub('', readData)
    return result

[ ] for i in range(len(data)):
    data.iloc[i]['MainText'] = cleanText(data.iloc[i]['MainText'])

[ ] for i in range(len(data)):
    if data.iloc[i]['MainText'][0:2] in ['1.', '2.', '3.', '4.', '5.', '6.', '7.', '8.', '9.']:
        data.iloc[i]['MainText'] = data.iloc[i]['MainText'][3:]
    elif data.iloc[i]['MainText'][0:4] in ['100.']:
        data.iloc[i]['MainText'] = data.iloc[i]['MainText'][5:]
    else:
        data.iloc[i]['MainText'] = data.iloc[i]['MainText'][4:]

[ ] corpus = data['MainText'].values.tolist()
corpus_embeddings = model.encode(corpus, convert_to_tensor=True)
queries = data['MainText'].sample(n=7).values.tolist()
```

```
[ ] top_k = 5
for query in queries:
    query_embedding = model.encode(query, convert_to_tensor=True)
    cos_scores = util.pytorch_cos_sim(query_embedding, corpus_embeddings)[0]
    cos_scores = cos_scores.cpu()

    #We use np.argpartition, to only partially sort the top_k results
    top_results = np.argpartition(-cos_scores, range(top_k+1))[0:top_k+1]

    print("\n\n=====\n")
    print("Query:", query)
    print("\nTop 5 most similar sentences in corpus:")

    for idx in top_results[1:top_k+1]:
        print(corpus[idx].strip(), "(Score: %.4f)" % (cos_scores[idx]))
```

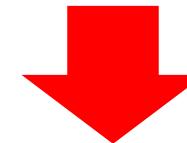
Model 3



Similar Questions Extraction Model

Query

산업안전보건법령상 안전보건표지의
종류 중 경고표지에 해당하지 않는 것은?

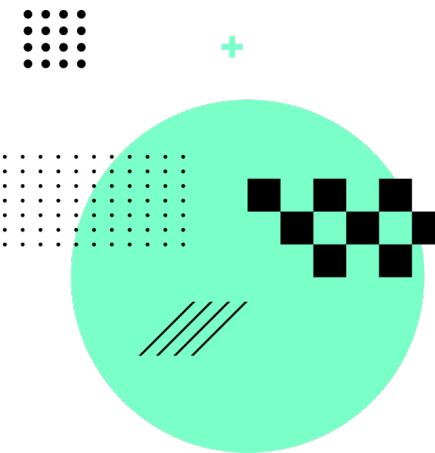


- 1) 산업안전보건법령상 안전보건표지의 종류와 형태 중 관계자 외 출입금지에 해당하지 않는 것은? (Score: 0.7140)
- 2) 다음 중 산업안전보건법령상 안전인증대상 방호장치에 해당하지 않는 것은? (Score: 0.6858)
- 3) 산업안전보건법령상 안전보건표지의 종류 중 경고표지의 기본모형형태이 다른 것은? (Score: 0.6751)
- 4) 산업안전보건법령상 위험물질의 종류를 구분할 때 다음 물질들이 해당하는 것은? (Score: 0.6625)
- 5) 산업안전보건법령상 중대재해의 범위에 해당하지 않는 것은? (Score: 0.6538)

Conclusion

Summary

- We built three NLP models for different purposes with Korean Certificate Exam Questions.
- As a result, we used both our own LSTM model and the latest developed BERT models
- And we used both our own crawled data and kakao open data.

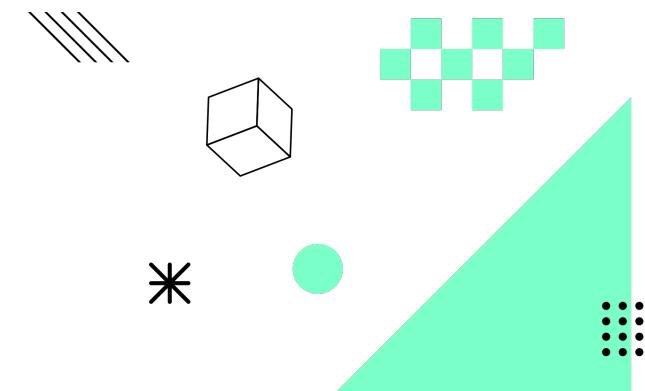


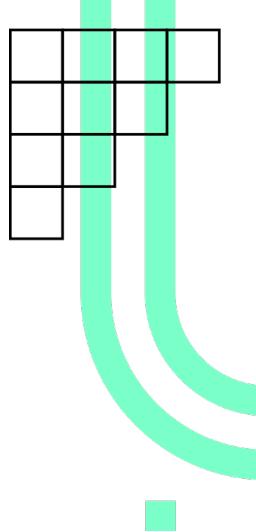
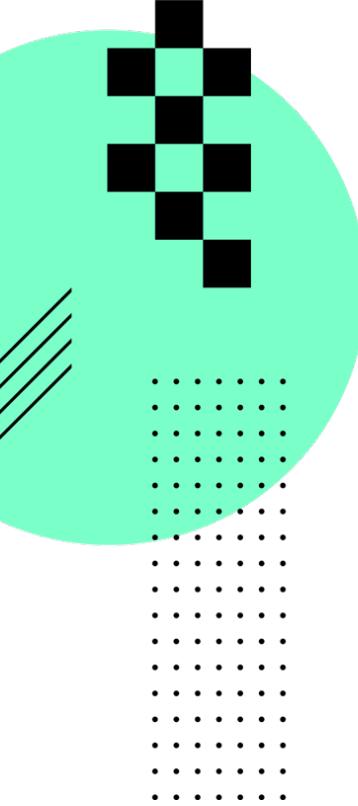
Contribution

- This study contributes to use NLP models to the new domain
- We expect this experience to help people in education industry and education policymakers.

References

- Myeong-Cheol Jwa, Jeong-Woo Jwa. (2022). Development of Tourism Information Named Entity Recognition Datasets for the Fine-tune KoBERT-CRF Model . International Journal of Internet, Broadcasting and Communication, 14(2), 55-62.
- Jwa, M.-C., & Jwa, J.-W. (2022). Development of Tourism Information Named Entity Recognition Datasets for the Fine-tune KoBERT-CRF Model. International Journal of Internet, Broadcasting and Communication, 14(2), 55–62. <https://doi.org/10.7236/IJIBC.2022.14.2.55>
- Hyunji Kim. (2021). A Study on Brand Image Analysis of Gaming Business Corporation using KoBERT and Twitter Data. Journal of Korea Game Society, 21(6), 75-85.
- Hannah Lee, Hyewon Bae, Kyuhong Shim, Wonyong Sung. (2022). Four Character Idiom Classification using KoBERT. 한국정보과학회 학술발표논문집, 2129-2131.
- KyuHwon Park, Young-Seob Jeong. (2021). Korean Daily Conversation Topics Classification Using KoBERT. 한국정보과학회 학술발표논문집, 1735-1737.
- A-Gyeong Kim, Young-Seob Jeong. (2021). Topic classification of domestic music using KoBERT. 한국정보과학회 학술발표논문집, 1738-1740.
- Park S, Moon J, Kim S, et al. KLUE: Korean Language Understanding Evaluation. arXivorg. Published online 2021. doi:10.48550/arXiv.2105.09680
- SKTBrain. KoBERT. GitHub. https://github.com/SKTBrain/KoBERT/tree/master/kobert_hf
- KoBERT. (2021, February 14). SKT Open Source. <https://sktelecom.github.io/project/kobert/>
- Text classification using RNN and CNN. Gist. <https://gist.github.com/Lucia-KIM/165b8f13c007f83b4762ab436ea95610>
- Huffon. Huffon/klue-transformers-tutorial. GitHub. <https://github.com/Huffon/klue-transformers-tutorial>
- KLUE-benchmark. Korean NLU Benchmark. GitHub. <https://github.com/KLUE-benchmark/KLUE>





Q & A

Deep Learning Analysis of Korean Certificate Exam Questions

DEEP LEARNING AND ITS APPLICATIONS TEAM 15

2017232044 KIM JINU
2017143004 CHOI HYEONGHO
2020147046 LEE HAKMIN