

Structured Relevant Knowledge Extraction

Project Instructions

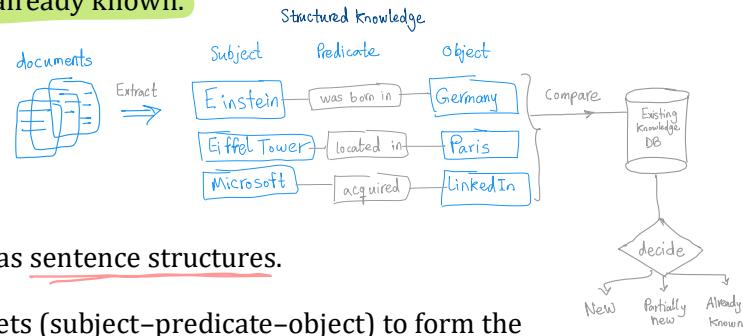
This project focuses on relevant knowledge extraction and integration to existing knowledge base. The goal is to extract structured knowledge (in the form of subject-predicate-object triplets) from documents, compare it with existing knowledge, and decide whether the information is new, partially new, or already known.

فَاعل - مُعْتَدَل - مُعَوِّل بِكَمْ

Project Workflow

1. Input Knowledge (Ground Truth)

- Start with introductory knowledge provided as sentence structures.
- Convert these sentences into structured triplets (subject–predicate–object) to form the initial knowledge base. Use NLP/LLM. Store sentence information as property of predicate.



2. Select Document Corpus

- Use a document (e.g., a PDF or text corpus) relevant to your chosen domain (e.g., environmental reports, technical standards, scientific papers).

- This document serves as the unstructured source of new knowledge.

3. Chunking Process

- Segment the document into smaller chunks (e.g., paragraphs, fixed-size windows, semantic units, etc.).

- Experiment with different chunking strategies and justify which works best for your corpus.
- Take care of coreference resolution.

1. What is Coreference Resolution?

Coreference resolution is the process of figuring out which words refer to the same entity in a text.

For example:

- Sentence: "Einstein was a physicist. He developed the theory of relativity."
- Without coreference resolution:
 - (Einstein, was, physicist)
 - (He, developed, theory of relativity)
- With coreference resolution:
 - (Einstein, was, physicist)
 - (Einstein, developed, theory of relativity)

So, it avoids ambiguity by making sure "He" is correctly linked to "Einstein".

→ This step is crucial before extracting subject–predicate–object triplets.

4. Triplet Extraction

- Use either traditional NLP pipelines (dependency parsing, OpenIE, spaCy) or LLM-based extraction (prompting) to extract triplets.
- Each chunk should yield candidate triplets.

5. Knowledge Comparison

- Compare extracted triplets against the initial ground truth.
- Categorize them into: Exists (already in the knowledge base), Partially new (some overlap with existing knowledge), New (completely new information). Keep log of decision makings.



Existing Knowledge DB

Neo4J

Graph DB Analytics

ChromaDB

①

②

③

④

⑤

⑥

⑦

⑧

⑨

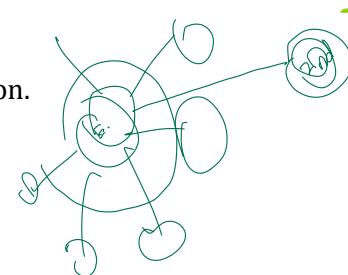
⑩

⑪

⑫

⑬

6. Knowledge Integration



4

5

6

7

8

9

10

11

12

13

14

15

16

17

18

19

20

21

22

23

24

25

26

27

28

29

30

31

32

33

34

35

36

37

38

39

40

41

42

43

44

45

46

47

48

49

50

51

52

53

54

55

56

57

58

59

60

61

62

63

64

65

66

67

68

69

70

71

72

73

74

75

76

77

78

79

80

81

82

83

84

85

86

87

88

89

90

91

92

93

94

95

96

97

98

99

100

101

102

103

104

105

106

107

108

109

110

111

112

113

114

115

116

117

118

119

120

121

122

123

124

125

126

127

128

129

130

131

132

133

134

135

136

137

138

139

140

141

142

143

144

145

146

147

148

149

150

151

152

153

154

155

156

157

158

159

160

161

162

163

164

165

166

167

168

169

170

171

172

173

174

175

176

177

178

179

180

181

182

183

184

185

186

187

188

189

190

191

192

193

194

195

196

197

198

199

200

201

202

203

204

205

206

207

208

209

210

211

212

213

214

215

216

217

218

219

220

221

222

223

224

225

226

227

228

229

230

231

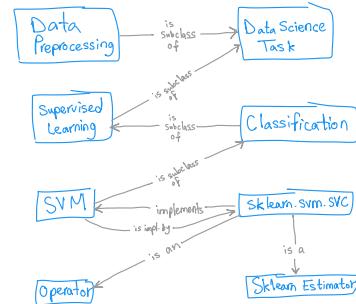


- Expand the knowledge base by integrating new or partially new triplets.
- Ensure consistency and avoid duplication.
- Take care of normalizing entity representation by considering singular, plural cases, etc.

Considerations for ML/AI/LLM Students

- **Ground Truth:** Choose introductory knowledge (sentence structures converted into triplets). This acts as your baseline knowledge graph. Example:

Data Preprocessing is subclass of *Data Science Task*.
Supervised Learning is subclass of *Data Science Task*.
Classification is subclass of *Supervised Learning*.
SVM is subclass of *Classification*.
sklearn.svm.SVC is an *Operator*.
sklearn.svm.SVC is a *Sklearn Estimator*.
sklearn.svm.SVC implements *SVM*.
SVM is implemented by *sklearn.svm.SVC*.



- Document Corpus: Choose a related document in your area of interest (e.g., AI, ML, LLM, etc). This will be chunked and processed for new knowledge extraction.
- Evaluation: Use qualitative analysis to judge relevancy.

Expand the knowledge base by starting from a limited but fundamental ground truth. Even if the initial base knowledge does not cover all possible information, use it as an anchor point to process related documents, extract structured triplets, and iteratively integrate new knowledge. This approach ensures that the knowledge graph grows systematically while maintaining alignment with the initial truth.