

Travail Encadré de Recherche

UNIVERSITÉ PARIS-SACLAY

FACULTY OF SCIENCES - MASTER IN DATA SCIENCE



Preparation of a challenge in Retrieval Augmented
Generation on technical texts.

Supervisor: Kim Gerdes
Academic Year 2023-2024
Pablo Mollá Chárlez

Contents

1	Project and Objectives	1
1.1	Purpose of the Report	1
2	Fundamentals of Patent Law	2
2.1	Definition of a Patent	2
2.2	Structure of a European Patent	2
2.3	Citations	3
2.3.1	Types of Citations	3
3	PatentMatch: A Dataset for Matching Patent Claims & Prior Art	4
3.1	Original Dataset & Parsing Script	4
3.2	Docker & ElasticSearch	5
3.3	Citing, Cited & Mapping Datasets	5
3.4	Q7 Dataset	6
3.5	Training and Test Datasets	7
4	Dataset RoadMap	8
5	Paper Research: SBERT	9
5.1	Overview of SBERT and BERT	9
5.2	SBERT's Efficient Improvements	9
5.3	Architectural Differences	9
5.4	Semantic Understanding and Training Techniques	10
6	Conclusion	10
7	Appendices	11
7.1	European Patent Example: Spinner Toy	12
7.2	0_parse.py	15
7.3	ElasticSearch	17
7.4	Q7-Dataset	18
7.5	Q7.py-Script	19

1 Project and Objectives

In early January 2024, I embarked on an engaging project with Qatent as part of my Master's in Data Science program under the guidance of Professor Kim Gerdes. This opportunity arose from a collaboration aimed at enhancing patent analytics through the use of data science techniques. Alongside three fellow students and PhD students, we formed a diverse team tasked with utilizing the PatentMatch dataset as a baseline. Through various modifications—including filtering, analysis, and data visualization—and creating new subdatasets, we developed a data challenge hosted on CodaBench for students in the Information Retrieval course. We applied machine learning algorithms such as BM25, TF-IDF, and transformer models (BERT, SBERT) to make predictions on citations, paragraphs, claims, and various features related to patent matters.

1.1 Purpose of the Report

The primary purpose of this report is to present the context, document the tasks performed by myself, the modifications made to the PatentMatch dataset, and the insights gained throughout the duration of the project. This report aims to showcase the practical applications of data science in patent law and

to illustrate the project's contributions to the field of patent analytics and to the Information Retrieval course.

2 Fundamentals of Patent Law

2.1 Definition of a Patent

Patents are like [official certificates](#) given by an institution that grant inventors the exclusive right to make, use, sell, and import an invention for a certain period of time, usually 20 years. Think of them as protective shields for inventions. These inventions can be new gadgets, chemical formulas, processes, or any new and useful improvement of an existing invention. There are multiple reasons to why patents are important and interesting:

- protection for inventors, encourages innovation, economic growth, revenue generation, sharing knowledge.

The [duration of patent](#) protection varies by region. In general, a patent typically lasts for 20 years from the filing date of the application. For instance, the European Patent Office (EPO) and the United States Patent and Trademark Office (USPTO) patents have a standard term of 20 years from the date of filing the application. It is essential to understand that to keep a patent in force, maintenance fees may be required, and these fees can vary between jurisdictions. It's also worth noting that the [scope of patent protection](#) is regional. A patent granted by the EPO is valid in European member states. When seeking patent protection in multiple regions, an applicant can file an international patent application under the World Intellectual Property Office (WIPO).



2.2 Structure of a European Patent

A European patent is a legal document protecting an invention in multiple European countries and is administered by the European Patent Office. The typical structure of a European patent document includes the following components: ¹

- **Date of Publication:** The date on which the patent document is officially published by the European Patent Office.
- **Application Number:** A unique identifier assigned to the patent application when it is filed.
- **Date of Filing:** The official date on which the patent application was submitted to the EPO.
- **Designated Contracting States:** The countries in which patent protection is sought, selected from the member states of the European Patent Convention (EPC).
- **International Patent Classification (IPC):** Codes assigned to the patent, indicating the technical fields to which the invention relates.
- **Applicant & Inventor:** The names of the applicant (the entity applying for the patent) and the inventor(s) (the person(s) who created the invention).

¹See the Appendix (5.1) for an example of a European patent.

- **Abstract:** A brief summary of the invention and its potential applications.
- **Description:**
 - *Field of Invention:* The technical field to which the invention belongs.
 - *Background Art:* Prior art and existing technologies relevant to the invention, highlighting the problems addressed.
 - *Summary of Invention:* An overview of what the invention is and how it addresses the problems identified in the background.
 - *Description of Drawings:* Explanation of the figures and drawings included in the patent document.
 - *Detailed Description:* A comprehensive, detailed explanation of the invention, often supported by examples and embodiments.
- **Claims:** The most crucial part of the patent, defining the scope of protection conferred by the patent. Each claim specifies particular technical features of the invention.
- **Figures:** Drawings and diagrams that illustrate the invention, essential for understanding the invention.
- **International Search Report:**
 - *Documents considered to be relevant:* Lists prior art documents that were taken into consideration during the search.
 - *References Cited in the Description:* Additional documents cited by the inventor or applicant that are relevant to the patentability of the invention.

This structure ensures that the patent document is comprehensive and provides all necessary details to understand the invention fully, assess its novelty, and determine the scope of protection it seeks.

2.3 Citations

When filing a patent, it's strategic to reference relevant prior art to show that you're aware of the existing technologies in the field. Putting too much focus on existing inventions (prior art) could unintentionally make your own invention seem less unique or innovative, or give the impression that it's just a minor variation.

The aim should be **to carefully reference the prior art that your invention improves upon or is distinct from**. These **citations assess the novelty, inventive step, and patentability of the invention**. Here we discuss three specific types of citations often encountered in patent documents: X, A, and Y citations.

2.3.1 Types of Citations

- X Definition** An X citation indicates highly relevant prior art that challenges the novelty or non-obviousness of the claimed invention.
- A Definition:** An A citation refers to documents that provide relevant technological background but do not directly relate to the novelty of the invention.
- Y Definition:** Y citations are relevant to the novelty or inventive step but do not conclusively challenge the patentability.

In the realm of patent analytics and research, the utilization of comprehensive datasets is crucial for developing tools and methods that enhance our understanding and processing of patent documents.

Patent examiners need to solve a complex information retrieval task when they assess the novelty and inventive step of claims made in a patent application. Given a claim, they search for prior art, which comprises all relevant publicly available information. This time-consuming task requires a deep understanding of the respective technical domain and the patent-domain-specific language.

In response to the challenge of identifying relevant historical inventions, a specialized dataset named PatentMatch was developed in 2020 to aid in the computerized identification of prior art. This dataset was crafted to support supervised machine learning approaches, allowing for the more refined training of algorithms in this domain.

3 PatentMatch: A Dataset for Matching Patent Claims & Prior Art

The PatentMatch dataset, as detailed on its GitHub repository, is a structured collection of patent texts and associated metadata, designed to support the matching of patent claims with prior art, with entries dating back to 1978.

This compilation, structured originally in XML format, includes matched sets comprising patent application claims and associated textual sections of varying relevance from referenced patents processed by the European Patent Office. A key addition post-2012 is the inclusion of search reports that assist in evaluating the uniqueness and innovative progress of the patent applications.

These sets have been meticulously annotated by the expert patent examiners at the EPO, designating each pair with a tag/category that reflects the semantic relationship between the claim and the text excerpt. The tags A, X and Y, as previously mentioned, serve to indicate the potential impact of the text on the originality of the patent claim in question.

3.1 Original Dataset & Parsing Script

The script 0_parse.py is developed to read XML files sourced from the EPO’s bulk data sets designed for text analytics, which can be found at the EPO’s data website. Its primary function is to meticulously extract and structure all relevant information from these XML files.

In my role, it was imperative to gain a thorough understanding of how this extraction process was carried out by the original script. I encountered the need to modify the baseline script to rectify inconsistencies and ensure the extraction process aligned with our specific project objectives. This involved filtering out erroneous data and selectively parsing data that was pertinent to our study — specifically, descriptions in English and data falling within the designated timeframe, which included citations and patent applications from 2010 to the year 2020.

To improve data quality, I performed cleansing operations on the fields and employed regular expressions, which proved essential for retrieving certain pieces of information that the initial functions had overlooked. The ultimate challenge was to successfully apply this refined parsing process to the entirety of the 28 out of 40 XML files (each approximately 5GB in size), to effectively handle and process the complete filtered dataset of 73 out of 200GB. This comprehensive processing² was a pivotal step in preparing the data for subsequent stages of our project.

²See the Appendix (7.2) for the multiple functions.

3.2 Docker & ElasticSearch

After completing the initial parsing of the data using the `0_parse.py` script, we recognized the necessity to enhance our data management system. To achieve this, we decided to utilize **Docker**.

- **Docker** is a popular **open-source platform** that simplifies the creation, deployment, and running of applications using containerization technology.
- Docker containers wrap software in a complete filesystem that **contains everything needed to run**: code, runtime, system tools, system libraries – anything that can be installed on a server. This guarantees that the **software will always run the same**, regardless of its environment.

One of our team members took the initiative to use Docker on the ‘**calcul**’ server offered by the **Laboratoire Interdisciplinaire des Sciences Numériques** (LISN) of the Université Paris-Saclay, provided by our professor. This strategic move was essential for managing the extensive datasets effectively.

Using Docker, we downloaded the **Elasticsearch’s official image** to create the corresponding container and handle the large volumes of text data extracted by our parsing script.

- **Elasticsearch**, a powerful **open-source search** and **analytics engine**, is well-suited for navigating vast amounts of textual data.
- **ElasticVue** ³, a **web-based client that interfaces with Elasticsearch** to provide a more accessible view of the indexed data, enhancing our ability to query and manipulate the information effectively.

The dataset retrieved with `0_parse.py` was organized into **2 distinct subdatasets** which represent 1428046 patent applications ($\approx 72.9\text{GB}$) and 1287385 patent citations ($\approx 310\text{MB}$).

1. The first segment, denoted ‘`ep_patent_applications`’, encompasses all patent applications, detailing elements like the abstract, title, claims, and others, including their citation identifiers.
2. On the other hand, the second segment, named ‘`ep_patents_citation`’ houses each citation identifier alongside its relevant citation data. These two segments are **interlinked** by the shared citation identifiers, allowing for integrated analysis between patent submissions and their references.

This approach not only streamlined our data processing workflow but also enhanced the robustness and scalability of our data management system, ensuring efficient handling and retrieval of information as needed for our project’s success.

3.3 Citing, Cited & Mapping Datasets

One of my fellow comrades was in charge of further **segmenting** the ‘`ep_patent_applications`’ and ‘`ep_patent_citations`’ datasets from **ElasticSearch**, extracting and organizing them into four more specialized **JSON datasets**.

These datasets are crafted by extracting content information from the filtered and indexed data previously stored in Elasticsearch. The structuring into **JSON format** ensures that each piece of content is meticulously referenced by IDs, facilitating straightforward access and manipulation.

It’s important to emphasize that within these datasets, we selectively focused on citing documents within the “**A**” and “**G**” classifications of the International Patent Classification (IPC) system. The “**A**” classification considers a spectrum of patents belonging to **Human Necessities**. On the other hand, the “**G**” classification covers **Physics**, incorporating patents that frequently involve cutting-edge technologies and intricate physical principles. Furthermore, we honed in on patents designated as “**A**”, “**X**”, or “**Y**”

³See the Appendix (7.3) for a ElasticVue preview of the collected data.

within the search reports crafted by the patent examiners, ensuring a targeted approach for our analysis.

The aforementioned datasets are the following:

- [content_citing.json](#) (10,779 documents): This dataset comprises the content information of the patents that are citing other patents. It focuses on the textual content of the citing patents.
- [mapping_citing_cited.json](#) (10,779 documents): This dataset creates mappings between the citing patents and the patents they cite, establishing a direct reference system between different documents.
- [content_cited.json](#) (12,195 documents): This dataset contains the content of the patents that have been cited by other patents, capturing the essential information of the patents referenced in the citations.
- [content_uncited.json](#) (10,000 documents): This includes patents that have not been cited by others in the dataset, providing a baseline of patents for comparative purposes.

With these datasets, we were equipped to begin the necessary **preprocessing steps** to more directly analyze and understand the relationships within the data, **task in which a colleague and I were solely responsible for**.

Specifically, we aimed to identify and extract relevant paragraphs and claims from the cited patents, focusing on those references made by citing patents. This process is crucial for our primary objective: predicting the relationships and references between patents based on their content and citation patterns. To accomplish such task, we created the [Q7.py script](#) and its corresponding Q7.json dataset.

3.4 Q7 Dataset

The script [Q7.py](#) was created with the objective of synthesizing data from three out of the four previous JSON datasets: ‘content_citing.json’, ‘mapping_citing_cited.json’, and ‘content_cited.json’. The goal was to perform a refined intersection of this information into a new JSON structure that clearly delineates all pertinent details from individual patents.⁴

In assembling this consolidated dataset, we aimed to capture specific fields: the citing patent ID (‘ID_Citing’), the category of the citing patent (‘Category_Citing’), the text of its claims (‘Claims_Text’), the cited patent ID (‘ID_Cited’), the references cited within the search report (‘References_Cited’), and the cited content (‘Content_Cited’).

Our task was multifaceted and complex, particularly when it came to extracting and cleanly formatting the content from the citations as noted by the examiners in the [European Search Reports](#). These examiners each had their **unique style** of noting down the interrelation between the citing patents and their citations.

For instance, a patent’s information such as its abstract, description, figures, and claims content needed to be pulled and cleansed from our datasets⁵. We then had to carefully link this content with the relevant portions from the cited patents, as determined by the examiner.

This process involved extracting elements such as the abstract, specific paragraphs, and detailed descriptions of cited figures, a task that required a versatile approach due to the diversity of referencing styles employed by the examiners.

⁴See the Appendix (7.4) for an illustration of the expected Q7.json Dataset.

⁵See Appendix (7.4) for a content patent instance.

Regular expressions played a key role in our extraction process, as we faced a multitude of referencing formats within the citations. Our challenge was to design regular expressions that were broad enough to match any potential structure yet precise enough to retrieve only the relevant. For example, a citation might reference an entire document, specific claims, a range of claims, particular paragraphs, or even just figures and tables such as:

- The whole document; claims 5, 7, 8; claim 6; figures 1, 6, 7, 8, 9.
- the whole document; claims 7-10; claim 6; figures 1-6, 7, 8, 9.
- In particular, paragraphs 0013-0018, 0034, 0037-0039, 0045-0049; figures 2, 4.
- whole document; paragraph [0092] - paragraph [0095]; claim 4; example 6; tables 1, 2.
- Figures 2, 3, 3B, 3C, 4, 6, 7; figures 1-9; paragraph [0001] - paragraph [0009]; paragraph [0017] - paragraph [0026].

This thorough and detailed extraction process lead us to develop numerous functions ⁶ to tackle each situation and allowed us to generate a dataset perfectly suited for our ultimate goal: to directly ascertain which paragraphs and claims from a cited patent are being referenced by a citing patent, thereby facilitating accurate predictions of patent citations.

3.5 Training and Test Datasets

Once the Q7.json dataset was compiled and finalized, the next phase of our project involved dividing this comprehensive dataset into 4 more targeted subsets. This process was essential for **creating distinct training and testing** sets tailored for the Codabench challenge. Here’s a breakdown of how the dataset was segmented:

- **cleaned_content_citing.json** - This dataset, containing 7,831 documents, was curated to include only the cleaned content from the citing patents. This meant removing any irrelevant data, ensuring the dataset was primed for analysis.
- **citation_Train.json** - With 8,860 documents, this file became the training dataset. It includes a selection of patent data specifically chosen to train the machine learning models that participants would develop and refine during the challenge.
- **citation_Test.json** - Comprising 1,000 documents, this smaller subset serves as the testing dataset. It is utilized to evaluate the performance of the models trained using the `citation_Train.json`, providing a benchmark for the algorithms developed.
- **cleaned_content_uncited.json** - This dataset is somewhat unique as it contains a mix of 8,834 documents, plus an additional 8,000, all of which represent patents that weren’t cited by others in the dataset. It serves a crucial role in the challenge, used as negative examples in training machine learning models.

The creation of these four datasets marked the **completion of the preparation phase for the Codabench challenge**. With **training and testing datasets ready**, participants could engage in the challenge with datasets designed to simulate real-world patent analysis scenarios, ultimately testing their models’ abilities to predict and understand patent citations. This structured approach to dataset preparation was pivotal in ensuring that the Codabench challenge would be both comprehensive and representative of actual tasks faced by those working in the field of patent analytics.

⁶See Appendix (7.5) for detailed functions.

4 Dataset RoadMap

In the culminating steps of our project, we refined a significant corpus of patent data into a suite of datasets, each designed with a focused objective to advance our Information Retrieval capabilities for the Codabench challenge. Beginning with the comprehensive **raw data from EPO patent filings**, we applied a meticulous parsing technique, employing a script to transform this vast information into a more manageable format.

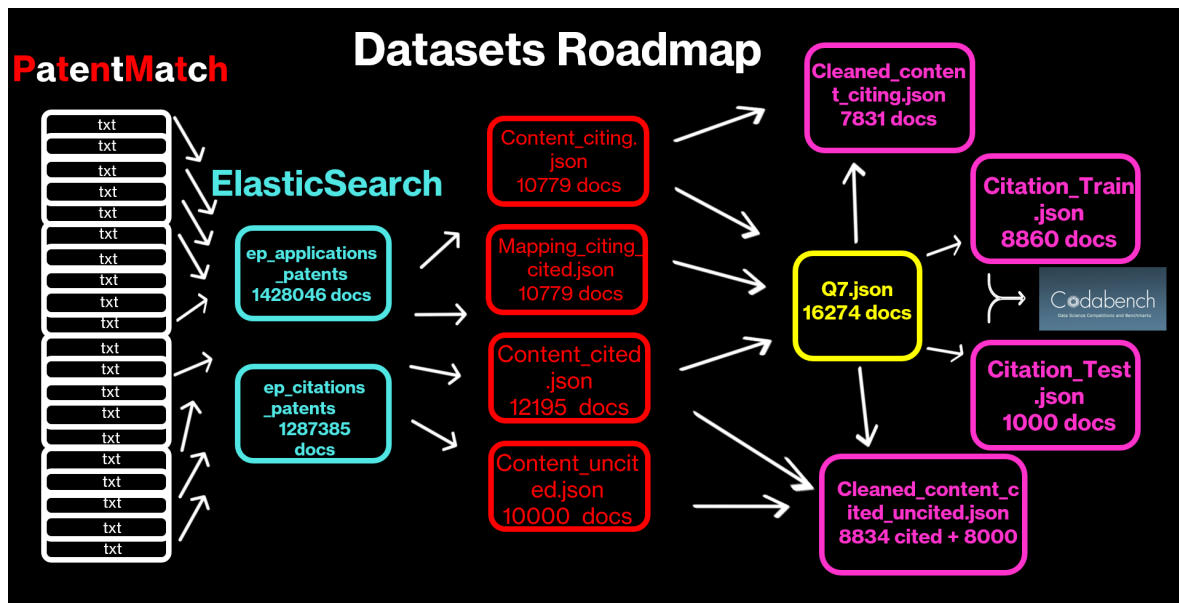
From the outset, **our data journey was steered by ElasticSearch**, serving as the engine for organizing over a million documents into **two primary collections**: one for **patent applications** and the other for **patent citations**.

These initial datasets were then expertly distilled, thanks to a dedicated script, **into four datasets** with a collaborative effort (content_citing.json, mapping_citing_cited.json, content_cited.json and content_uncited.json). These datasets—ranging from content of citing patents to mappings of citations and uncited content, were **meticulously filtered** ("A" and "G" IPC classifications and "A", "X" or "Y" search report citations) referenced by unique identifiers, ensuring precision in our analysis and bridging both cited and citing.

As we synthesized the data, **the challenge** laid not just in searching through the IDs and categories but in extracting and formatting the content from a multitude of examiner citations. The **diversity of these citations** demanded robust regular expressions, among other techniques, to capture the necessary details accurately.

Eventually, this granular approach to data curation culminated in the **final assembly** of **training and testing datasets**. The 'Citation_Train.json' and 'Citation_Test.json' datasets emerged from this process, marking the end of our dataset preparation and the beginning of a new chapter of challenge preparation. This meticulous and structured approach to dataset creation has been pivotal, ensuring that our contributions to the **Codabench challenge** are as comprehensive and authentic as the real-world challenges faced by professionals in patent analytics.

For a detailed look at the transformation from raw data to the finalized datasets used in the Codabench challenge, please refer to the accompanying visual roadmap provided as follows.



5 Paper Research: SBERT

In the context of my TER, I also did some research on transformer models. I delved into the nuances of **Sentence-BERT** (SBERT), a significant modification of the well-known BERT model, as outlined in the original paper from 2019 titled Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks and presented to my comrades the key concepts. The main distinctions and benefits of SBERT over BERT, focusing on its architecture and application in natural language processing tasks are overviewed:

5.1 Overview of SBERT and BERT

SBERT is an adaptation of the original BERT (Bidirectional Encoder Representations from Transformers) designed to enhance the efficiency and accuracy of generating sentence embeddings. While BERT has set benchmarks in various NLP tasks, its architecture inherently faces challenges concerning efficiency, especially in tasks requiring sentence similarity assessments.

5.2 SBERT's Efficient Improvements

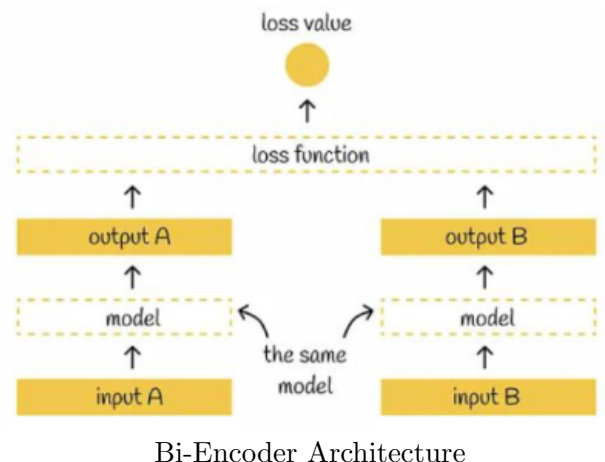
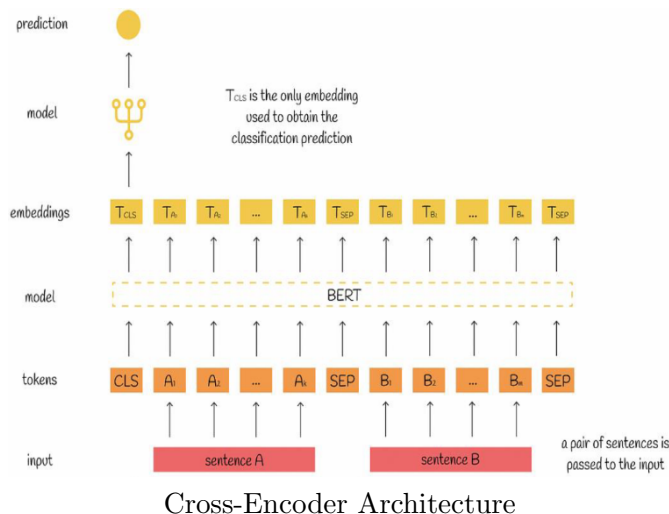
SBERT addresses these challenges by modifying the BERT architecture to produce **sentence embeddings** more efficiently. Traditional **BERT models** process sentences in pairs and **do not support the reuse of pre-computed sentence embeddings**. This approach results in a quadratic computation increase with the number of sentences, making it computationally expensive for large datasets.

In contrast, SBERT allows **embeddings to be pre-computed** and **reused**, significantly reducing computation time while maintaining the accuracy of the embeddings. This is achieved by adapting the model to a **siamese network structure**, which processes each sentence independently, thus supporting tasks like semantic similarity searches more efficiently.

5.3 Architectural Differences

The fundamental architectural change between **BERT** and **SBERT** is the move from a **cross-encoder** to a **bi-encoder** setup. BERT's **cross-encoder evaluates sentence pairs collectively**, using the transformer architecture to output a similarity score, which, while accurate, demands substantial computational resources. This setup does not produce standalone sentence embeddings, which limits its applicability for certain types of semantic analyses.

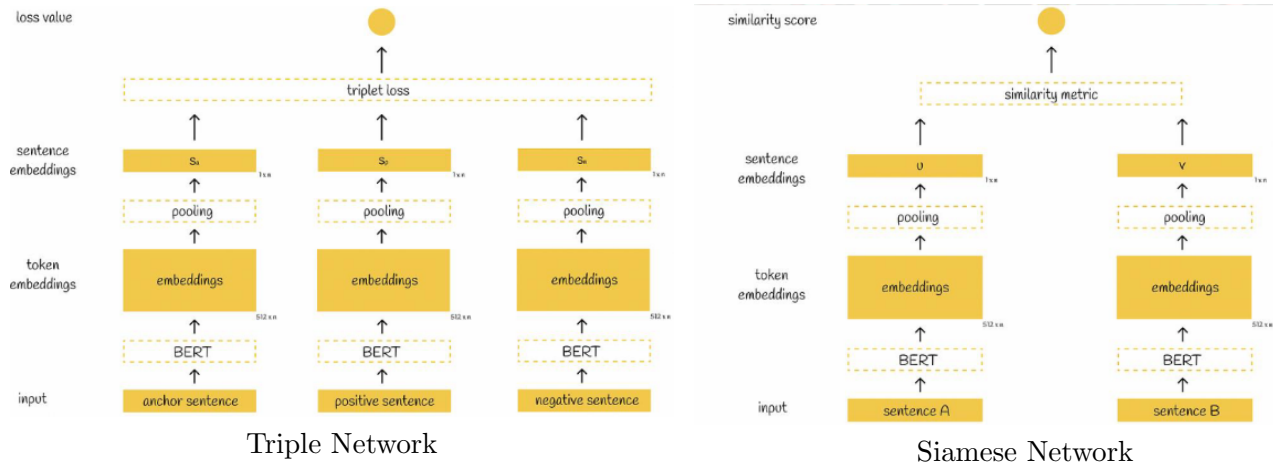
On the other hand, SBERT employs a **bi-encoder structure** where sentences are **processed independently**, generating distinct embeddings for each. These embeddings can then be compared using cosine similarity measures, making SBERT much faster and more versatile for applications involving large-scale semantic comparisons.



5.4 Semantic Understanding and Training Techniques

SBERT enhances the understanding of semantic meanings in sentences through a specialized training approach. It utilizes siamese and triplet network architectures within its training process, employing a technique known as the triplet loss function. This method involves training on sentence triplets: an anchor sentence, a positive sentence (semantically similar to the anchor), and a negative sentence (semantically different). The model learns to pull the anchor and positive sentences closer in the embedding space while pushing the negative sentence farther away.

This training strategy enables SBERT to achieve a nuanced understanding of sentence semantics, beneficial for tasks requiring precise interpretation of sentence meanings, such as matching questions to answers or identifying similar sentences across documents.



6 Conclusion

In conclusion, this report has set down the meticulous process of developing a structured and analytical approach to patent data, culminating in the creation of comprehensive datasets for the Codabench challenge. Our journey began with parsing the vast XML formatted patent data, leveraging the prowess of Elasticsearch managed through Docker on the LISN's "calcul" server. A collaborative effort led to the successful preprocessing and extraction of relevant patent information, overcoming challenges posed by diverse citation references.

Subsequent to the assembly of the Q7.json dataset, our team partitioned this data into distinct subsets, creating clean, targeted datasets for training and testing, essential for the machine learning models' development. The resulting datasets not only facilitated a deep dive into the nuances of patent citations but also laid a strong foundation for the Codabench challenge participants to innovate and test their predictive models.

My journey through the complex landscape of patent data has been profoundly educational and immensely enriching. The process of developing structured datasets for the Codabench challenge has endowed me with invaluable insights into patents, data mining techniques, and the application of NLP in machine learning models, particularly BM25, TF-IDF, BERT and SBERT transformers.

The hands-on experience with data preprocessing, information extraction, and the challenges of dealing with complex patent references has sharpened my skills and deepened my understanding of natural language processing and its pivotal role in Information Retrieval.

I am immensely grateful for the collaboration and assistance provided by my peers, whose expertise and shared effort made this project research possible. Their support was instrumental in overcoming the intricate challenges we faced. I am equally thankful for the patience and passionate guidance offered

by Professor Kim, whose enthusiasm and deep knowledge in the field were not only inspiring but also illuminating.

7 Appendices

The comprehensive material within the appendix of this report offers just a glimpse of the extensive information that has been compiled. For those who wish to delve into the full breadth of the research and findings, I invite you to visit my personal GitHub repository, where the complete scripts and documentation are accessible.

Throughout my research, I have engaged with a variety of scholarly materials to solidify my understanding of patent law and information retrieval techniques.

While some documents, particularly those central to my research, were read with meticulous attention to detail, others were used to grasp the general concepts and methodologies. This approach allowed me to efficiently allocate my time to the most pertinent subjects while maintaining a broad awareness of the field as a whole.

References

- [1] WORLD INTELLECTUAL PROPERTY OFFICE (WIPO), *Patent Draft Manual*, URL : http://www.ub.edu/centredopatents/pdf/material_referencia/WIPO_Patent_Drafting_Manual_2022.pdf, 2022.
- [2] WORLD INTELLECTUAL PROPERTY OFFICE (WIPO), *Patent Claims Format & Types of Claims*, URL : https://www.wipo.int/edocs/mdocs/aspac/en/wipo_ip_phl_16/wipo_ip_phl_16_t5.pdf, 2022.
- [3] EUROPEAN PATENT OFFICE - ESPACENET, *Patent Search Engine*, URL : <https://worldwide.espacenet.com/patent/>, 2024.
- [4] ARTICLE ON LLAMAINDEX.AI, *Running Mixtral 8x7 locally with LlamaIndex and Ollama*, URL : <https://www.llamaindex.ai/blog/running-mixtral-8x7-locally-with-llamaindex-e6cebeabe0ab>, 2023.
- [5] YUNFAN GAO, YUN XIONG, *Retrieval-Augmented Generation for Large Language Models: A Survey*, URL : <https://arxiv.org/pdf/2312.10997v4.pdf>, 2024.
- [6] TREY GRAINGER, DOUG TURNBULL, MAX IRWIN, *AI-Powered Search*, URL : <https://www.manning.com/books/ai-powered-search>, 2024.

7.1 European Patent Example: Spinner Toy

The original document can be found on the official website of the EPO:

- Espacenet Patent Search and Spinner Toy - Patent.

EP 3 939 678 A1

1

2

3

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

232

233

234

235

236

237

238

239

240

241

242

243

244

245

246

247

248

249

250

251

252

253

254

255

256

257

258

259

260

261

262

263

264

265

266

267

268

269

270

271

272

273

274

275

276

277

278

279

280

281

282

283

284

285

286

287

288

289

290

291

292

293

294

295

296

297

298

299

300

301

302

303

304

305

306

307

308

309

310

311

312

313

314

315

316

317

318

319

320

321

322

323

324

325

326

327

328

329

330

331

332

333

334

335

336

337

338

339

340

341

342

343

344

345

346

347

348

349

350

351

352

353

354

355

356

357

358

359

360

361

362

363

364

365

366

367

368

369

370

371

372

373

374

375

376

377

378

379

380

381

382

383

384

385

386

387

388

389

390

391

392

393

394

395

396

397

398

399

400

401

402

403

404

405

406

407

408

409

410

411

412

413

414

415

416

417

418

419

420

421

422

423

424

425

426

427

428

429

430

431

432

433

434

435

436

437

438

439

440

441

442

443

444

445

446

447

448

449

450

451

452

453

454

455

456

457

458

459

460

461

462

463

464

465

466

467

468

469

470

471

472

473

474

475

476

477

478

479

480

481

482

483

484

485

486

487

488

489

490

491

492

493

494

495

496

497

498

499

500

501

502

503

504

505

506

507

508

509

510

511

512

513

514

515

516

517

518

519

520

521

522

523

524

525

526

527

528

529

530

531

532

533

534

535

536

537

538

539

540

541

542

543

544

545

546

547

548

549

550

551

552

553

554

555

556

557

558

559

560

561

562

563

564

565

566

567

568

569

570

571

572

573

574

575

576

577

578

579

580

581

582

583

584

585

586

587

588

589

590

591

592

593

594

595

596

597

598

599

600

601

602

603

604

605

606

607

608

609

610

611

612

613

614

615

616

617

618

619

620

621

622

623

624

625

626

627

628

629

630

631

632

633

634

635

636

637

638

639

640

641

642

643

644

645

646

647

648

649

650

651

652

653

654

655

656

657

658

659

660

661

662

663

664

665

666

667

668

669

670

671

672

673

674

675

676

677

678

679

680

681

682

683

684

685

686

687

688

689

690

691

692

693

694

695

696

697

698

699

700

701

702

703

704

705

706

707

708

709

710

711

712

713

714

715

716

717

718

719

720

721

722

723

724

725

726

727

728

729

730

731

732

733

734

735

736

737

738

739

740

741

742

743

744

745

746

747

748

749

750

751

752

753

754

755

756

757

758

759

760

761

762

763

764

765

766

767

768

769

770

771

772

773

774

775

776

777

778

779

780

781

782

783

784

785

786

787

788

789

790

791

792

793

794

795

796

797

798

799

800

801

802

803

804

805

806

807

808

809

810

811

812

813

814

815

816

817

818

819

820

821

822

823

824

825

826

827

828

829

830

831

832

833

834

835

836

837

838

839

840

841

842

843

844

845

846

847

848

849

850

851

852

853

854

855

856

857

858

859

860

861

862

863

864

865

866

867

868

869

870

871

872

873

874

875

876

877

878

879

880

881

882

883

884

885

886

887

888

889

890

891

892

893

894

895

896

897

898

899

900

901

902

903

904

905

906

907

908

909

910

911

912

913

914

915

916

917

918

919

920

921

922

923

924

925

926

927

928

929

930

931

932

933

934

935

936

937

938

939

940

941

942

943

944

945

946

947

948

949

950

951

952

953

954

955

956

957

958

959

960

961

962

963

964

965

966

967

968

969

970

971

972

973

974

975

976

977

978

979

980

981

982

983

984

985

986

987

988

989

990

991

992

993

994

995

996

997

998

999

1000

1001

1002

1003

1004

1005

1006

1007

1008

1009

1010

1011

1012

1013

1014

1015

1016

1017

1018

1019

1020

1021

1022

1023

1024

1025

1026

1027

1028

1029

1030

1031

1032

1033

1034

1035

1036

1037

1038

1039

1040

1041

1042

1043

1044

1045

1046

1047

1048

1049

1050

1051

1052

1053

1054

1055

1056

1057

1058

1059

1060

1061

1062

1063

1064

1065

1066

1067

1068

1069

1070

1071

1072

1073

1074

1075

1076

1077

1078

1079

1080

1081

1082

1083

1084

1085

1086

1087

1088

1089

1090

1091

1092

1093

1094

1095

1096

1097

1098

1099

1100

1101

1102

1103

1104

1105

1106

1107

1108

1109

1110

1111

1112

1113

1114

1115

1116

1117

1118

1119

1120

1121

1122

1123

1124

1125

1126

1127

1128

1129

1130

1131

1132

1133

1134

1135

1136

1137

1138

1139

1140

1141

1142

1143

1144

1145

1146

1147

1148

1149

1150

1151

1152

1153

1154

1155

1156

1157

1158

1159

1160

1161

1162

1163

1164

1165

1166

1167

1168

1169

1170

1171

1172

1173

1174

1175

1176

1177

1178

1179

1180

1181

1182

1183

1184

1185

1186

1187

1188

1189

1190

1191

1192

1193

1194

1195

1196

1197

1198

1199

1200

1201

1202

1203

1204

1205

1206

1207

1208

1209

1210

1211

1212

1213

1214

1215

1216

1217

1218

1219

1220

1221

1222

1223

1224

1225

1226

1227

1228

1229

1230

1231

1232

1233

1234

1235

1236

1237

1238

1239

1240

1241

1242

1243

1244

1245

1246

1247

1248

1249

1250

1251

1252

1253

1254

1255

1256

1257

1258

1259

1260

1261

1262

1263

1264

1265

1266

1267

1268

1269

1270

1271

1272

1273

1274

1275

1276

1277

1278

1279

1280

1281

1282

1283

1284

1285

1286

1287

1288

1289

1290

1291

1292

1293

1294

1295

1296

1297

1298

1299

1300

1301

1302

1303

1304

1305

1306

1307

1308

1309

1310

1311

1312

1313

1314

1315

1316

1317

1318

1319

1320

1321

1322

1323

1324

1325

1326

1327

1328

1329

1330

1331

1332

1333

1334

1335

1336

1337

1338

1339

1340

1341

1342

1343

1344

1345

1346

1347

1348

1349

1350

1351

1352

1353

1354

1355

1356

1357

1358

1359

1360

1361

1362

1363

1364

1365

1366

1367

1368

1369

1370

1371

1372

1373

1374

1375

1376

1377

1378

1379

1380

1381

1382

1383

1384

1385

1386

1387

1388

1389

1390

1391

1392

1393

1394

1395

1396

1397

1398

1399

1400

1401

1402

1403

1404

1405

1406

1407

1408

1409

1410

1411

1412

1413

1414

1415

1416

1417

1418

1419

1420

1421

1422

1423

1424

1425

1426

1427

1428

1429

1430

1431

1432

1433

1434

1435

1436

1437

1438

1439

1440

1441

1442

1443

1444

1445

1446

1447

1448

1449

1450

1451

1452

1453

1454

1455

1456

1457

1458

1459

1460

1461

1462

1463

1464

1465

1466

1467

1468

1469

1470

1471

1472

1473

1474

1475

1476

1477

1478

1479</

3. The spinner toy according to claim 1, characterized in that the rotating element (1) has the shape of a volumetric body, which is configured to rotate about a vertical axis passing through the center of gravity of the volumetric body.
4. The spinner toy according to claim 1, characterized in that at least two screw grooves (12) of a given profile are made on the inner surface of the body (5) of the hollow sleeve (11), while their beginning and end are made turning into circular single turn grooves (30).
5. A method of rotating a spinner toy according to claim 1 of the formula, comprising imparting rotation to a rotating element (1) with a centrally installed means of rotation (2), characterized in that the rotation movement to the rotating element (1) is imparted without interrupting the use of the spinner toy by the user by continuously maintaining its rotation by means of mechanical transformation of the reciprocating motion of the pusher (19) of the means of rotation (2) in the form of a ball bearing and screw converter (10) into unidirectional rotational motion without slowing down the rotating element (1).
6. The method of rotation of the spinner toy according to claim 5, characterized in that the reciprocating motion of the pusher (19) of the ball bearing and screw converter (10) is performed by periodically pressing the spring-loaded top (6) button cover of the ball bearing and screw converter (10), followed by the return of the pusher (19) by the spring (29) to the initial upper position, while simultaneously holding the spinner toy with the rotating element (1) in the play position between the user's fingers.

EP 3 939 678 A1

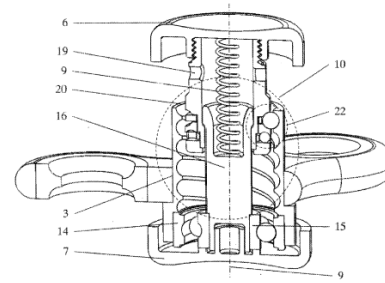


Fig. 1

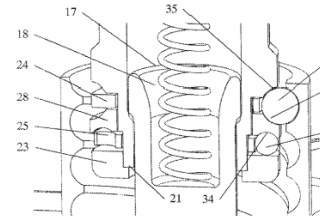


Fig. 2

7

EP 3 939 678 A1

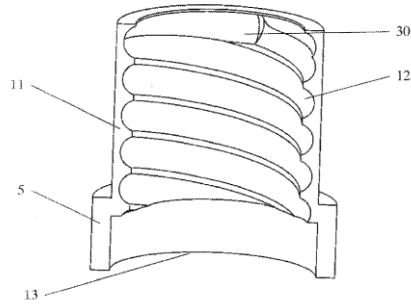


Fig. 3

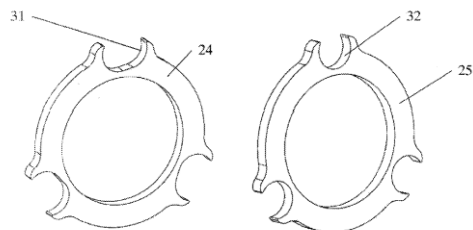


Fig. 4

Fig. 5

EP 3 939 678 A1

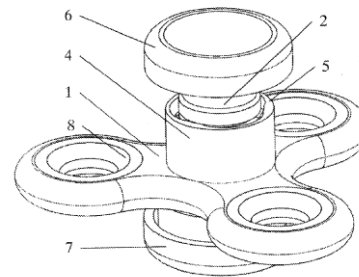


Fig. 6

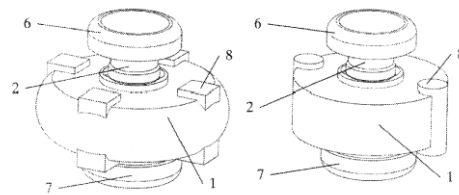


Fig. 7

Fig. 8

9

EP 3 939 678 A1

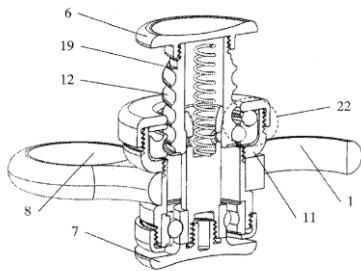


Fig. 9

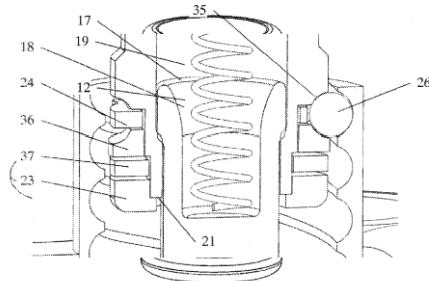


Fig. 10

10

INTERNATIONAL SEARCH REPORT

International application No.
PCT/BY 2020/000003

A. CLASSIFICATION OF SUBJECT MATTER A63F 9/16 (2006.01) A63H1/00 (2019.01) According to International Patent Classification (IPC) or to both national classification and IPC		
B. FIELDS SEARCHED Minimum documentation searched (classification system followed by classification symbols) A63F Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched Electronic data base consulted during the international search (name of data base and, where practicable, search terms used) Espacenet, PatSearch, PAJ, WIPO, USPTO, RUPTO		
C. DOCUMENTS CONSIDERED TO BE RELEVANT		
Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
A	US 5135425 A (ANDREWS MELVIN R; ANDREWS ROGER W) 04.06.1992	1-6
A	US 9914063 B1 (MCCOSKERY MICHAEL SCOTT et al.) 13.03.2018	1-6
A	US 2017326468 A1 (KINMONT JR RICHARD C et al.) 16.11.2017	1-6
A	CN 107638699 A (DONGGUAN DAVANTECH CO LTD) 30.01.2018	1-6
<input type="checkbox"/> Further documents are listed in the continuation of Box C. <input type="checkbox"/> See patent family annex.		
* Special categories of cited documents: "A" document defining the general state of the art which is not considered to be of particular relevance but published on or after the international filing date "E" earlier application or patent but published on or after the international filing date "L" document which may have priority claims or which is cited to establish the publication date of another claim or other special reason (as specified) "O" document referring to an oral disclosure, use, exhibition or other means "P" document published prior to the international filing date but later than the priority date claimed "T" later document published after the international filing date or priority date and not in conflict with the application but cited to underscore the principle or theory underlying the invention "X" document of particular relevance; the claimed invention cannot be considered novel or inventive in view of the document cited above "Y" document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other cited documents, such combination being obvious to a person skilled in the art "Z" document number of the same patent family		
Date of the actual completion of the international search 14 May 2020 (14.05.2020)		Date of mailing of the international search report 21 May 2020 (21.05.2020)
Name and mailing address of the ISA/ Facsimile No.		Authorized officer Telephone No.

from PCT/ISA/210 (second sheet) July 1998)

11

EP 3 939 678 A1

REFERENCES CITED IN THE DESCRIPTION

This list of references cited by the applicant is for the reader's convenience only. It does not form part of the European patent document. Even though great care has been taken in compiling the references, errors or omissions cannot be excluded and the EPO disclaims all liability in this regard.

Patent documents cited in the description

- | | |
|-----------------------|------------------------|
| • TW M551522 [0028] | • JP 07092017 B [0028] |
| • TW 11112017 [0028] | • CN 107638699 [0028] |
| • CN 107754323 [0028] | • US 9895620 B [0028] |
| • CN 06032018 [0028] | • US 9914063 B [0028] |
| • JP 3212430 B [0028] | • US 5135425 A [0028] |

7.2 0_parse.py

Some important functions modified to index all the dataset in ElasticSearch.

```
def from_citationIDs_to_application_number(line):
    pattern_patcit = re.compile('patcit dnum="[A-Z]{2}[A-Z\d]+'')
    result_patents = [elem[13:-1] for elem in pattern_patcit.findall(line)]
    return result_patents

def extract_citationIDs(application_identifier, line):
    words = line.split("\t")[7].split(" ")
    indices = [i for i, x in enumerate(words) if "sr-cit" in x]
    return [application_identifier + "_" + words[i][words[i].find("sr-cit")
        +6:words[i].find("sr-cit")+10] for i in indices]

def main(file):
    f = open(file, "r", encoding="utf8", errors='ignore')
    lines = f.readlines()
    records = {}
    citations = {}
    pattern_en = re.compile("\t20(1[0-9]|20)-\d{2}-\d{2}\ten\t")
    flag_d = False
    flag_c = False
    flag_t = False
    flag_a = False
    flag_am = False
    flag_ams = False
    temp_record = {}
    for line in lines:
        if re.search(pattern_en, line):

            if "\tTITLE\t" in line:
                if flag_c:
                    flag_a = False
                    flag_d = False
                    flag_c = False
                    application_identifier = line.split("EP\t")[1].split("\ten\t")[0].replace("\t", "")
                    application_title = line.split("\tTITLE\t")[1][2:]
                    flag_t = True
                    temp_record = {}
                else:
                    application_identifier = line.split("EP\t")[1].split("\ten\t")[0].replace("\t", "")
                    application_title = line.split("\tTITLE\t")[1][2:]
                    flag_t = True

            if "\tABSTR\t" in line and flag_t:
                application_abstract = line.split("\tABSTR\t")[1]
                flag_a = True

            if "\tDESCR\t" in line and flag_t:
                application_number = line.split("EP\t")[1].split("\t")[0]
                application_category = line.split("EP\t")[1].split("\t")[1]
                application_date = line.split("EP\t")[1].split("\t")[2]
```

Continuation of the function on next page


```

if not flag_a:
    temp_record = {
        application_identifier : {
            "application_number": application_number,
            "application_category": application_category,
            "application_date": application_date,
            "title": application_title,
            "description": line.split("\tDESCR\t")[1][7:]}
    flag_d = True
    if application_date == "":
        #print("Skipping entry, missing date: " + application_identifier)
        continue
elif flag_a:
    temp_record = {
        application_identifier : {
            "application_number": application_number,
            "application_category": application_category,
            "application_date": application_date,
            "title": application_title,
            "abstract": application_abstract,
            "description": line.split("\tDESCR\t")[1][7:]}
    flag_d = True
    if application_date == "":
        #print("Skipping entry, missing date: " + application_identifier)
        continue
if "\tCLAIM\t" in line and flag_d and application_identifier in temp_record:
    flag_c = True
    temp_record[application_identifier]["claims"] = line.split("\tCLAIM\t")[1][7:]
if "\tAMEND\t" in line and flag_c and application_identifier in temp_record:
    application_amend = line.split("\tAMEND\t")[1]
    temp_record[application_identifier]["amended_claims"] = application_amend
    #flag_am = True # need to reset
if "\tACSTM\t" in line and application_identifier in temp_record:
    application_amended_statements = line.split("\tACSTM\t")[1]
    temp_record[application_identifier]["amended_claims_statements"] = application_amended_statements
    #flag_ams = True
if "\tSRPRT\t" in line and application_identifier in temp_record:
    # if extract_classifications(line):
    # and extract_citationIDs(application_identifier, line):
    temp_record[application_identifier]["citation_ipcr_classification"] = extract_classifications(
        line)
    temp_record[application_identifier]["citation_ids"] = extract_citationIDs(
        application_identifier, line)
    temp_record[application_identifier]["citation_application_number"] =
    from_citationIDs_to_application_number(line)
    application_number_and_category = application_number + application_category
    for citation_id in temp_record[application_identifier]["citation_ids"]:
        #print("evaluate citation id: "+citation_id)
        citations[citation_id] = extract_citation_entry(
            citation_id,
            line.split("\tSRPRT\t")[1],
            application_number_and_category)
if "\tPDFEP\t" in line and application_identifier in temp_record:
    application_url = line.split("\tPDFEP\t")[1]
    temp_record[application_identifier]["publication_url"] = application_url
if application_identifier in temp_record: # and "citation_ids"
    records.update(temp_record)

print(f"File {file} finished")
upload(records, INDEX_APPL, "patent_eu")
upload(citations, INDEX_CIT, "citation_eu")

```

7.3 ElasticSearch

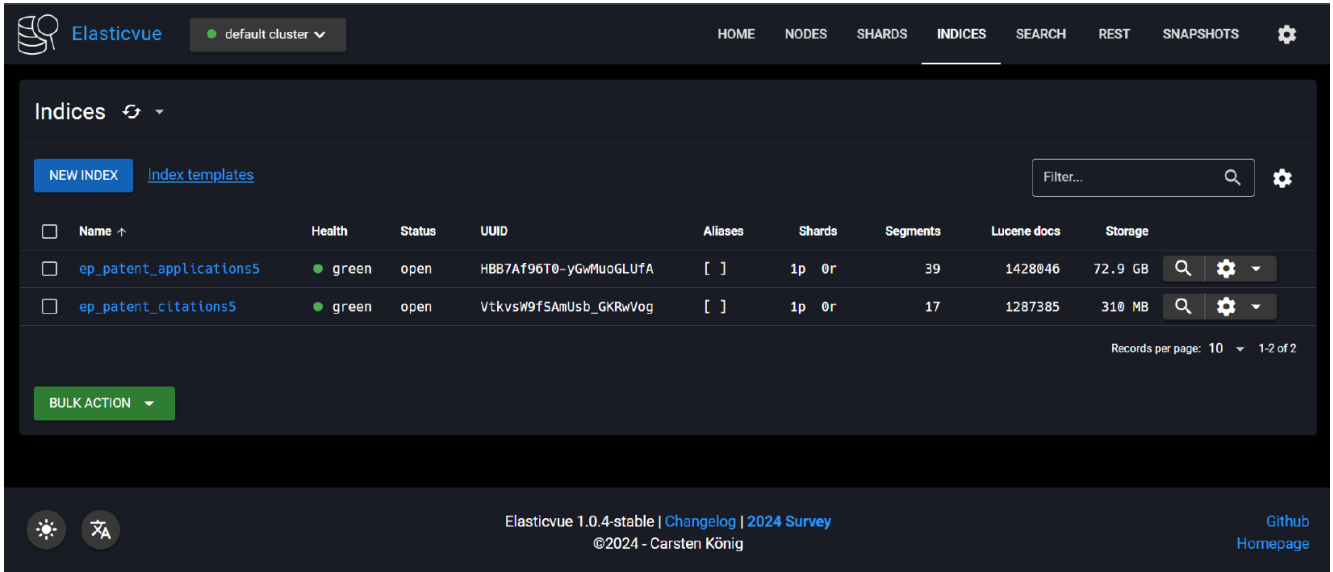


Figure 1: ElasticVue visualization of indexed datasets ep_patent_applications & ep_patent_citations.

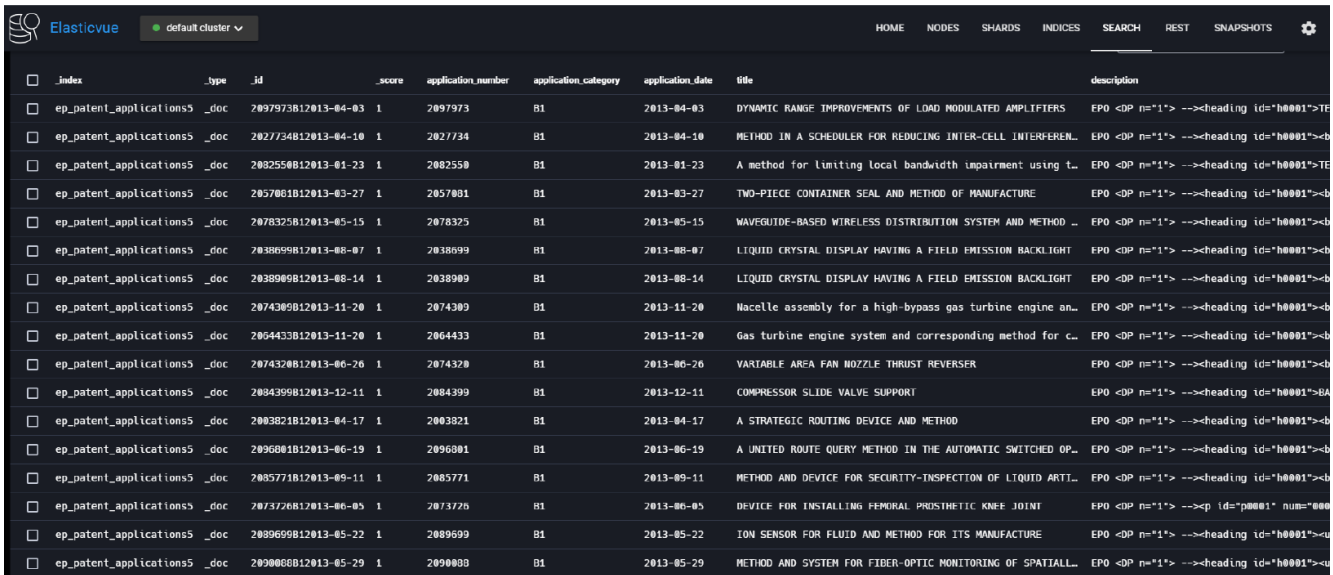


Figure 2: Visualization of ep_patent_applications dataset.

7.4 Q7-Dataset

```

1 DATASETS.JSON | Qmlto -f QT_DRAFT_S09aen.j | Q
2
3 {
4   "ID_Citing": "3672283A1",
5   "Category_Cited": "X",
6   "Claims_Text": {
7     "c-en-0001": "Method for carrying out a health check of at least two cameras (12a-12d) comprising the steps of: - initiating (502) a health check process; - checking (506), in
8       "c-en-0002": "Method according to claim 1, wherein the selecting (508) further comprises selecting the motion event based on that the motion event was stored, registered or
9       "c-en-0004": "Method according to any of the preceding claims, wherein in response to that the checking step (506) indicates that no motion event has been stored in the database
10      "c-en-0005": "Method according to any of the preceding claims, further comprising the step of: composing (512) a single video sequence including the retrieved video clips, of
11      "c-en-0006": "Method according to claim 5 further comprising a step of transmitting (514) the composed video sequence to a communication network (30) or making the composed vid
12      "c-en-0007": "Method according to any of the preceding claims wherein the stored motion event is generated by one of the cameras (12a-12e) as a result of motion having been det
13      "c-en-0008": "Method according to any of the preceding claims wherein each video clip has a predetermined length of time ranging from 1 seconds to 10 seconds,"
14      "c-en-0010": "A camera system (10) having an integrated health checker, the camera system (10) comprising: at least two cameras (12a-12e) connected to a communications netwo
15      "c-en-0011": "The camera system (10) according to claim 10 wherein the retriever (20) is further arranged to retrieve a snap shot or a live video clip captured by one of the at l
16      "c-en-0012": "The camera system (10) according to claim 10 wherein the retriever (20) is further comprised a connector (24) coupled to the retrieved video clip and corresponding camera identifi
17      "c-en-0013": "The camera system (10) according to claim 10 or 12 further comprising: a transmitter (26) configured to forward the composed video sequence to the communicatio
18    },
19    "ID_Cited": "Z709350A1",
20    "References_Cited": "** abstract ** paragraphs [0005] - [0073] ** figure 6 **,
21    "Content_Cited": {
22      "pde1": "\t\tThere is provided a method for configuring a set of image capturing sections of a camera for a first scene condition type currently viewed by the camera. The method
23        "pde5": "In some cases there may be a large number of test images 408-a to select from. In order to simplify the selection in such cases the selection may be performed as a se
24        "pde6": "Fig. 6 illustrates a plurality of test images 608. The plurality of test images may be divided into groups, for example by the selection receiving unit 312. The numbe
25        "pde7": "The selection receiving unit 310 may then receive input relating to one of the groups 602, 604, 606, 608 which has been selected. The representative test image may se
26        "pde8": "The selected group 606 in turn comprises a plurality of sub-groups 608-a-n of test images. Each sub-group 608-a-n may be a group of several test images. If so, the sele
27        "pde9": "Eventually, after repeating the above procedure until the selected sub-group comprises a single test image, the selection receiving unit 312 thus receives input rel
28        "pde10": "The above sequential procedure for selecting test images is efficient and reduces the time and effort of selecting among the test images."
29        "pde11": "In step 510 the set of image capturing settings 404 corresponding to the selected test image 408 is stored in the second storing unit 314 as the configured set o
30        "pde12": "The above method has been described with respect to a first scene condition type 408. However, it is to be understood that the method may be applied repeatedly in ord
31        "pde13": "Moreover, the method may be applied as soon as a new scene condition type is detected, i.e., as soon as a scene condition type for which no configured set of image ca
32        "pde14": "The above, as well as additional objects, features and advantages of the present invention, will be better understood through the following illustrative and non-limit
33      }
34    }
35  }
36
37   "ID_Citing": "3672283A1",
38   "Category_Cited": "X",
39   "Claims_Text": {
40     "c-en-0001": "A method for improving the spatial hearing perception of a binaural hearing aid (20), said binaural hearing aid (20) comprising a first local unit (21) and a seco
41     "c-en-0002": "The method according to claim 1, wherein the first reference signal (32) is derived from a set of first input signals (26, 62), each of which is generated from the
42     "c-en-0004": "The method according to claim 3, wherein in dependence of the first coherence parameter (38), a first magnitude threshold value is derived, and wherein the first m
43     "c-en-0005": "The method according to claim 4, wherein the first coherence parameter (38) is calculated for a plurality of subsequent time-frequency bins, and wherein the first a
44     "c-en-0006": "The method according to one of the preceding claims, wherein a first phase threshold value is compared to a phase of the first coherence parameter, and wherein at t
45     "c-en-0007": "The method according to one of the preceding claims, wherein the first reference signal (32) is generated from the first input signal (26), wherein the second refe
46     "c-en-0008": "The method according to one of the claims 1 to 6, wherein in the first local unit (21), a first supplementary input signal (62) is generated from the environment
47   }

```

Figure 3: Visualization of Q7’s content dataset.

ep_patent_applications5 / _doc / 2097973B12013-04-03 ↻

_index	ep_patent_applications5	_id	2097973B12013-04-03	_version	1	_primary_term	1	_seq_no	173793
1	{								
2	"application_number": "2097973",								
3	"application_category": "B1",								
4	"application_date": "2013-04-03",								
5	"title": "DYNAMIC RANGE IMPROVEMENTS OF LOAD MODULATED AMPLIFIERS",								
6	"description": "EPO <DP n=\> --><heading id=\>TECHNICAL FIELD OF THE INVENTION</heading><p id=\>num=\>The present invention relates generally to power amplifiers and amplifying methods and more specifically to high efficiency power amplifiers.</p><heading id=\>DESCRIPTION OF RELATED ART</heading><p id=\>num=\>In radio transmitters for broadcast, cellular and satellite systems the power amplifier in the transmitter has to be very linear in addition to being able to simultaneously amplify many radio channels (i.e. frequencies) spread across a wide bandwidth. High linearity is required since nonlinear amplifiers would cause leakage of interfering signal energy between channels and distortion within each channel.</p><p id=\>num=\>In the wireless communication industry a premium is placed on the ability to amplify wide bandwidth signals, e.g. spread spectrum signals, in highly efficient manner. To limit the size of the DC power supply and cooling equipment in a radio base station, it is essential to keep a high overall power efficiency. Various attempts have been made to address this problem, however it remains difficult to design a high efficiency power amplifier system that is at the same time also able to linearly amplify wide bandwidth signals.</p><p id=\>num=\>One such system is the feed forward amplification system, in which the output signal from the amplifier stage is compared to the input signal to be able to determine the difference signal. To outbalance the non-desired distortion components due to non-linear amplification, said difference-signal is amplified to a suitable amount and added in reversed phase to the output signal from the power amplifier.<!-- EPO <DP n=\> --></p><p id=\>num=\>In a pre-distortion amplifier system however, the input signal first passes a non-linear pre-distorter, which is an inverse non-linear function to the transfer function of the power stage. Thus, the pre-distortion of the input signal will automatically compensate for the distortion generated by the power stage. In an adaptive pre-distortion system, correction values are stored in a look-up table arrangement of which the output is due to the incoming vector amplitude and the output signals amplitude and phase.</p><p id=\>num=\>Another known method and system for power amplifying is EER (Envelope Elimination and Restoration). The method is to determine the envelope of the input signal and regenerate it on the output by means of a modulating step. The amplifier operates with constant gain and output amplitude.</p><p id=\>num=\>Further, another known system and method is called Envelope Tracking. By this, the output signal is compared to the input drive signal for generating an amplifier control signal. Said signal controls the gain of the system and compensates for deviations. The system comprises								

Figure 4: Visualization of raw content information from the dataset ep_patent_applications.

7.5 Q7.py-Script

Some important functions created to extract the paragraphs, claims or figures content.

```
def extract_and_expand_numbers(text):
    pattern = re.compile(r"\\(\\d{4})\\(?:\\s*\\s*paragraph\\s*\\(\\d{4})\\|\\s*\\s*\\(\\d{4})\\)?)")
    matches = pattern.findall(text)
    numbers = []
    for match in matches:
        start = match[0]
        end = match[1] if match[1] else match[2]
        if start and end:
            start_num, end_num = int(start), int(end)
            numbers.extend(range(start_num, end_num + 1))
        elif start:
            numbers.append(int(start))
    return sorted(set(numbers))

def clean_text(html_text):
    clean_text = re.sub(r'<[^>]+>', '', text)

def extract_paragraphs_by_ids(text, paragraph_ids=None):
    extracted_paragraphs = {}
    # Update to match any paragraph ID if None provided
    if paragraph_ids:
        for pid in paragraph_ids:
            pattern = re.compile(r'<p id="p{:04d}" num="\\d+">(.*?)</p>'.format(int(pid)), re.DOTALL)
            match = pattern.search(text)
            if match:
                paragraph_text = match.group(1)
                paragraph_text = re.sub(r'<figref idref="^[^"]*>(.*?)</figref>', r'\\1', paragraph_text)
                paragraph_text = re.sub(r'<[^>]+>', '', paragraph_text)
                extracted_paragraphs[f"p{pid:04d}"] = paragraph_text
    else:
        paragraph_pattern = re.compile(r'<p id="p(\\d+)">[^>]*>(.*?)</p>', re.DOTALL)
        matches = paragraph_pattern.findall(text)
        for pid, pcontent in matches:
            # Clean up <figref> tags and other HTML tags from paragraphs
            pcontent_clean = re.sub(r'<figref idref="^[^"]*>(.*?)</figref>', r'\\1', pcontent)
            pcontent_clean = re.sub(r'<[^>]+>', '', pcontent_clean).strip()
            extracted_paragraphs[f"p{int(pid):04d}"] = pcontent_clean
    return extracted_paragraphs

def split_text_with_complex_pattern(text):
    pattern = re.compile(
        r'\\.(?!\\s*(Fig|Figs|FIG|FIGS)\\.\\s*\\d+([-_,]\\s*\\d+)*\\s+|</?\\s*(li|ul|p|!-)>\\s*(?=[A-Z])'
    )

    # Split the text using the defined pattern
    lines = pattern.split(text)
    #print("Lines:", lines)
    # Ensure that we filter out any None before processing
    lines = [line for line in lines if line is not None]

    return lines
```

```

def extract_claims_from_paragraphs(text):
    claim_numbers = []
    # Check if "claims" is mentioned without specific numbers
    if re.search(r'\bclaims?;\bclaims?$', text, re.IGNORECASE):
        return [0]
    # Pattern to find claims and their ranges, correcting for spaces around dashes
    claims_pattern = re.compile(r'\bclaims?\s*([0-9,-]+)', re.IGNORECASE)
    text = re.sub(r'(\d+)\s*-\s*(\d+)', r'\1-\2', text) # Remove spaces around dashes in ranges
    text = re.sub(r'\s*-\s*', '-', text) # Remove spaces around dashes in ranges
    #print(text)
    claims_matches = claims_pattern.findall(text)

    for match in claims_matches:
        claim_list = match.split(',')
        for claim_range in claim_list:
            claim_range = claim_range.strip()
            if '-' in claim_range:
                parts = claim_range.split('-')
                # Ensure both parts are valid before proceeding
                if len(parts) == 2 and parts[0].isdigit() and parts[1].isdigit():
                    start_claim, end_claim = map(int, parts)
                    claim_numbers.extend(range(start_claim, end_claim + 1))
            elif claim_range.isdigit():
                claim_numbers.append(int(claim_range))

    return sorted(set(claim_numbers))

def extract_lines_with_figure_references(text):
    # Remove HTML tags but keep their content, especially for <figref>
    cleaned_text = re.sub(r'<figref idref="\w+\>', '', text)

    # Some cleaning
    cleaned_text = cleaned_text.replace('</figref>', '')

    # Lines split correctly
    well_split = split_text_with_complex_pattern(cleaned_text)

    # Keywords to filter
    keywords = ["figure", "figures", "Figure", "Figures", "FIG.", "FIGS.", "Fig.", "Figs."]

    figure_sentences = [line for line in well_split if any(keyword in line for keyword in keywords)]

    lista = []
    for text in figure_sentences:
        # Case 1: Removing <figref> including brackets and extracting the figure number or range directly.
        text = re.sub(r'[\s*<figref idref="[^"]+\>(FIGS?. [0-9a-zA-Z\(\)-]+\</figref>\s*)', r'\1', text)

        # Case 2: Direct replacement of <figref> tags with their content.
        text = re.sub(r'<figref idref="[^"]+\>(FIGS?. [0-9a-zA-Z\(\)-]+\</figref>', r'\1', text)

        # Case 3: Similar to case 2 but specifically targeting the pattern with brackets at the start.
        text = re.sub(r'[\s*<figref idref="[^"]+\>(Fig. [0-9]+\s*)?\s*\1</figref>', r'\1', text)

        # Clean remaining HTML tags but keep their content
        text = re.sub(r'<[>]+\>', '', text)
        lista.append(text)

    return lista

```