HEIG-VD — MAC

Practical Work 3 – Lab Report

Loïc Herman

19 avril 2024

Deliverable 1

```
PUT /cacm_standard
{
  "settings": {
    "analysis": {
      "analyzer": {
        "default": {
          "type": "standard"
        }
      }
    }
  },
  "mappings": {
    "properties": {
      "id": {
        "type": "keyword",
        "store": true,
        "index": false
      },
      "author": {
        "type": "keyword"
      },
      "title": {
        "type": "text",
        "fielddata": true
      },
      "date": {
        "type": "date"
      },
      "summary": {
        "type": "text",
        "fielddata": true,
        "index_options": "offsets"
      }
    }
  }
}

POST _reindex
{
  "source": {
    "index": "cacm_dynamic"
  },
  "dest": {
    "index": "cacm_standard"
  }
}
```

Listing 1 – Requests for deliverable 1

Deliverable 2

```
PUT /cacm_termvector
{
  "mappings": {
    "properties": {
      "id": {
        "type": "keyword",
        "store": true,
        "index": false
      },
      "author": {
        "type": "keyword"
      },
      "title": {
        "type": "text",
        "fielddata": true
      },
      "date": {
        "type": "date"
      },
      "summary": {
        "type": "text",
        "fielddata": true,
        "index_options": "offsets",
        "term_vector": "with_offsets"
      }
    }
  }
}

POST _reindex
{
  "source": {
    "index": "cacm_dynamic"
  },
  "dest": {
    "index": "cacm_termvector"
  }
}
```

Listing 2 – Requests for deliverable 2

Deliverable 3

```
GET /cacm_termvector/_termvectors/GUQtV44BSj7Uev35U4zP?fields=summary
```

Listing 3 – Requests for deliverable 3

```
{
  "_index": "cacm_termvector",
  "_id": "GUQtV44BSj7Uev35U4zP",
  "_version": 1,
  "found": true,
  "took": 3,
  "term_vectors": {
    "summary": {
      "field_statistics": {
        "sum_doc_freq": 97730,
        "doc_count": 1585,
        "sum_ttf": 150220
      },
      [truncated]
    }
  }
}
```

Listing 4 – Response of deliverable 3

Deliverable 4

A « term vector » in Elasticsearch is a detailed breakdown of the terms present in a document, including their frequency and positions. It provides information like term frequency (TF), inverse document frequency (IDF), term positions, and field statistics. This data helps in tasks such as search relevance, scoring, and text analytics by offering insights into the content and structure of documents.

Deliverable 5

```
GET /cacm_termvector,cacm_standard/_stats/store
```

Listing 5 – Requests for deliverable 5


```
{
  "indices": {
    "cacm_standard": {
      "uuid": "TFcr2URyR62LM6OVpjDl3g",
      "total": {
        "store": {
          "size_in_bytes": 1894157,
          "total_data_set_size_in_bytes": 1894157,
          "reserved_in_bytes": 0
        }
      }
    },
    "cacm_termvector": {
      "uuid": "f-maSwi9TeW4Bh1OD_m6rA",
      "total": {
        "store": {
          "size_in_bytes": 2636782,
          "total_data_set_size_in_bytes": 2636782,
          "reserved_in_bytes": 0
        }
      }
    }
  }
}
```


Listing 6 – Response of deliverable 5 (partly truncated for size)

The size of the `cacm_standard` index is **1.89 MB**, while the `cacm_termvector` is higher at **2.63 MB**.

The difference in size between the two indices can be attributed to the inclusion of term vectors in the `cacm_termvector` index. Term vectors store additional information about the terms present in the documents, such as their frequency and positions, which allows for more advanced text processing and analysis.

Therefore, the larger size of the `cacm_termvector` index indicates that it contains more detailed information about the textual content of the documents compared to the `cacm_standard` index. This extra information comes at the cost of increased storage space. We note however that this impact is mostly limited since most of the documents do not have a `summary` field, which was the only field we used the analysis on.

Deliverable 6

```
GET /cacm_standard/_search
{
  "size": 0,
  "aggregations": {
    "authors": {
      "terms": {
        "field": "author",
        "size": 1
      }
    }
  }
}
```

Listing 7 – Requests for deliverable 6


```
{
  "took": 0,
  "timed_out": false,
  "_shards": {
    "total": 1,
    "successful": 1,
    "skipped": 0,
    "failed": 0
  },
  "hits": {
    "total": {
      "value": 3202,
      "relation": "eq"
    },
    "max_score": null,
    "hits": []
  },
  "aggregations": {
    "authors": {
      "doc_count_error_upper_bound": 0,
      "sum_other_doc_count": 4268,
      "buckets": [
        {
          "key": "Thacher Jr., H. C.",
          "doc_count": 38
        }
      ]
    }
  }
}
```


Listing 8 – Response of deliverable 6

As such, **Thacher Jr., H. C.** has the most publications, appearing as the author in **38** documents.

Deliverable 7

```
GET /cacm_standard/_search
{
  "size": 0,
  "aggregations": {
    "titles": {
      "terms": {
        "field": "title",
        "size": 10
      }
    }
  }
}
```

Listing 9 – Requests for deliverable 7

	key	doc_count	frequency
1	of	1138	35.54%
2	algorithm	975	30.04%
3	a	895	27.95%
4	for	714	22.30%
5	the	645	20.14%
6	and	434	13.55%
7	in	416	12.99%
8	on	340	10.62%
9	an	275	8.59%
10	computer	275	8.59%

Table 1 – Results for deliverable 7. Total hits : **3202**.

Deliverable 8

8.a)

```
PUT /cacm_whitespace
{
  "settings": {
    "analysis": {
      "analyzer": {
        "default": {
          "type": "whitespace"
        }
      }
    }
  },
  "mappings": {
    "properties": {
      "id": { "type": "keyword", "store": true, "index": false },
      "author": { "type": "keyword" },
      "title": { "type": "text", "fielddata": true },
      "date": { "type": "date" },
      "summary": { "type": "text", "fielddata": true, "index_options": "offsets" }
    }
  }
}

POST _reindex
{
  "source": {
    "index": "cacm_dynamic"
  },
  "dest": {
    "index": "cacm_whitespace"
  }
}
```

Listing 10 – Creation of the `whitespace` analyzer-based index

8.b)

```
PUT /cacm_english
{
  "settings": {
    "analysis": {
      "analyzer": {
        "default": {
          "type": "english"
        }
      }
    }
  },
  "mappings": {
    "properties": {
      "id": { "type": "keyword", "store": true, "index": false },
      "author": { "type": "keyword" },
      "title": { "type": "text", "fielddata": true },
      "date": { "type": "date" },
      "summary": { "type": "text", "fielddata": true, "index_options": "offsets" }
    }
  }
}

POST _reindex
{
  "source": {
    "index": "cacm_dynamic"
  },
  "dest": {
    "index": "cacm_english"
  }
}
```

Listing 11 – Creation of the `english` analyzer-based index

8.c)

```
PUT /cacm_custom_lowercase_shingles_unigrams
{
  "settings": {
    "analysis": {
      "filter": {
        "shingles": {
          "type": "shingle",
          "min_shingle_size": 2,
          "max_shingle_size": 2,
          "output_unigrams": true
        }
      },
      "analyzer": {
        "default": {
          "type": "custom",
          "tokenizer": "standard",
          "filter": [ "lowercase", "shingles" ]
        }
      }
    }
  },
  "mappings": {
    "properties": {
      "id": { "type": "keyword", "store": true, "index": false },
      "author": { "type": "keyword" },
      "title": { "type": "text", "fielddata": true },
      "date": { "type": "date" },
      "summary": { "type": "text", "fielddata": true, "index_options": "offsets" }
    }
  }
}

POST _reindex
{
  "source": {
    "index": "cacm_dynamic"
  },
  "dest": {
    "index": "cacm_custom_lowercase_shingles_unigrams"
  }
}
```

Listing 12 – Creation of the index with a custom analyzer outputting shingles with a `lowercase` filter

8.d)

```
PUT /cacm_custom_lowercase_shingles
{
  "settings": {
    "analysis": {
      "filter": {
        "shingles": {
          "type": "shingle",
          "min_shingle_size": 3,
          "max_shingle_size": 3,
          "output_unigrams": false
        }
      },
      "analyzer": {
        "default": {
          "type": "custom",
          "tokenizer": "standard",
          "filter": [ "lowercase", "shingles" ]
        }
      }
    }
  },
  "mappings": {
    "properties": {
      "id": { "type": "keyword", "store": true, "index": false },
      "author": { "type": "keyword" },
      "title": { "type": "text", "fielddata": true },
      "date": { "type": "date" },
      "summary": { "type": "text", "fielddata": true, "index_options": "offsets" }
    }
  }
}

POST _reindex
{
  "source": {
    "index": "cacm_dynamic"
  },
  "dest": {
    "index": "cacm_custom_lowercase_shingles"
  }
}
```

Listing 13 – Creation of the index with a custom analyzer outputting shingles of size 3

8.e)

```
PUT /cacm_stop
{
  "settings": {
    "analysis": {
      "filter": {
        "stop": {
          "type": "stop",
          "stopwords_path": "data/common_words.txt"
        }
      },
      "analyzer": {
        "default": {
          "type": "custom",
          "tokenizer": "standard",
          "filter": [ "stop" ]
        }
      }
    }
  },
  "mappings": {
    "properties": {
      "id": { "type": "keyword", "store": true, "index": false },
      "author": { "type": "keyword" },
      "title": { "type": "text", "fielddata": true },
      "date": { "type": "date" },
      "summary": { "type": "text", "fielddata": true, "index_options": "offsets" }
    }
  }
}

POST _reindex
{
  "source": {
    "index": "cacm_dynamic"
  },
  "dest": {
    "index": "cacm_stop"
  }
}
```

Listing 14 – Creation of the index with a standard analyzer using a stop filter with common words

Deliverable 9

Whitespace analyzer

The whitespace analyzer breaks text into terms whenever it encounters any whitespace characters. It does not lower case the terms, nor does it remove any punctuation. This makes it quite simplistic and fast, suitable for scenarios where text is pre-processed or when exact matches, including punctuation and case, are necessary.

English analyzer

The english analyzer is designed specifically for the English language, providing several layers of processing to enhance text search. It includes tokenization, stemming, and the removal of English stop words. Stemming reduces words to their root form, which can significantly improve search capabilities by focusing on the meaning rather than the exact word forms. This analyzer is ideal for processing natural language data, enhancing search flexibility and relevance in English text databases.

Custom analyzer with lowercase filter and shingles of size 1 and 2

This custom analyzer builds upon the standard analyzer by applying a shingle filter that creates combinations of adjacent tokens (shingles) of size 1 and 2. It incorporates a lowercase filter and uses the standard tokenizer. The shingle process allows the analyzer to consider single words and their immediate co-occurrence with adjacent words, enhancing the ability to search for phrases and proximity-based queries without requiring exact phrase matches.

Custom analyzer with lowercase filter and shingles of size 3

Similar to the previous custom analyzer, this one also uses the standard tokenizer, but specifically generates shingles of size 3. This approach extends the context captured by the analyzer, making it adept at understanding and retrieving texts where three-word phrases or contexts are important. This can be particularly useful in specialized knowledge domains where specific three-word terms have significant meaning, like legal documents or technical materials.

Stop analyzer with custom stop list

The stop analyzer configured with a custom stop list uses the standard tokenizer to break down the text and then filters out terms that appear in the provided `common_words.txt` file. This custom list allows for precise control over which words are considered irrelevant to search queries, potentially differing from generic stop lists. This can optimize search results by ignoring frequently occurring but unimportant words specific to a given dataset or domain.

Deliverable 10

	whitespace	english	custom_shingles_unigrams	custom_shingles	custom_stop
document count	3202	3202	3202	3202	3202
term count	103275	72298	237189	144518	66555
top 10 terms	1 : of (1534)	1 : which (781)	1 : the (1541)	1 : in this paper (111)	1 : The (1072)
	2 : the (1501)	2 : us (778)	2 : of (1534)	2 : the use of (108)	2 : A (690)
	3 : is (1382)	3 : comput (663)	3 : a (1426)	3 : the number of (106)	3 : This (465)
	4 : and (1369)	4 : program (635)	4 : is (1384)	4 : it is shown (97)	4 : computer (441)
	5 : a (1321)	5 : system (586)	5 : and (1376)	5 : a set of (88)	5 : system (429)
	6 : to (1293)	6 : present (514)	6 : to (1301)	6 : in terms of (82)	6 : paper (421)
	7 : in (1188)	7 : describ (505)	7 : in (1234)	7 : the problem of (77)	7 : presented (372)
	8 : for (1167)	8 : paper (428)	8 : for (1182)	8 : is shown that (71)	8 : time (354)
	9 : The (1072)	9 : can (421)	9 : are (1025)	9 : a number of (67)	9 : program (339)
	10 : are (1022)	10 : gener (411)	10 : of the (938)	10 : as well as (63)	10 : data (309)
index disk size	1.88 MB	1.57 MB	3.68 MB	3.81 MB	1.59 MB
index time	182 ms	186 ms	276 ms	270 ms	176 ms

Table 2 – Result of the statistics on the indexes created hereinabove

Deliverable 11

Effectiveness of stemming and stop words : The English analyzer, which includes stemming and stop word removal, results in a significantly lower term count (72,298 terms) compared to the whitespace analyzer (103,275 terms). This indicates the effectiveness of stemming in reducing word variants to their root forms and stop words in filtering out common but less informative words. Moreover, the top terms in the English analyzer ("which", "us", "comput") suggest a focus on more contextually relevant terms, unlike common conjunctions and prepositions prevalent in the whitespace analysis.

Impact of shingle size on term complexity : The custom analyzers that generate shingles of sizes 1 and 2, and size 3 respectively, show a substantial increase in the number of indexed terms (237,189 and 144,518 terms) compared to the basic English and whitespace analyzers. The shingle size directly influences the complexity and quantity of terms, with larger shingles capturing more complex and less frequent phrase patterns. This is evident from the unique phrases captured in the top terms for the analyzer with shingle size 3, such as "the number of", "the use of", "in terms of", highlighting its utility in capturing longer contextual information. We'll add thereon that the top terms in the size 3 shingle analyzer have a much lower occurrence count than the other analyzers, making it explicit once again that this analyzer would be much preferred for queries with longer and more detailed phrases.

Performance and storage efficiency : Analyzers with additional processing like shingle creation and stemming tend to have larger index sizes and require longer indexing times. For instance, the custom analyzer with shingles of size 1 and 2 has the largest index size (3.68 MB) and took the longest to index (276 ms), whereas the stop analyzer with a custom stop list, despite its tailored stop word filtering, maintains a smaller index size (1.59 MB) close to the English analyzer (1.57 MB) and a moderate indexing time (176 ms). This suggests that while advanced processing can enhance search capabilities, it also demands more resources in terms of storage and processing time.

Deliverable 12

12.a)

```
GET /cacm_english/_search
{
  "stored_fields": [ "id" ],
  "query": {
    "query_string": {
      "query": "summary:\"Information Retrieval\""
    }
  }
}
```

Listing 15 – Requests for deliverable 12.1

12.b)

```
GET /cacm_english/_search
{
  "stored_fields": [ "id" ],
  "query": {
    "query_string": {
      "query": "+summary:Information +summary:Retrieval"
    }
  }
}
```

Listing 16 – Requests for deliverable 12.2

12.c)

```
GET /cacm_english/_search
{
  "stored_fields": [ "id" ],
  "query": {
    "query_string": {
      "query": "+summary:Retrieval summary:Information -summary:Database"
    }
  }
}
```

Listing 17 – Requests for deliverable 12.3

12.d)

```
GET /cacm_english/_search
{
  "stored_fields": [ "id" ],
  "query": {
    "query_string": {
      "query": "summary:Info*"
    }
  }
}
```

Listing 18 – Requests for deliverable 12.4

12.e)

```
GET /cacm_english/_search
{
  "stored_fields": [ "id" ],
  "query": {
    "query_string": {
      "query": "summary:\"Information Retrieval\"~5"
    }
  }
}
```

Listing 19 – Requests for deliverable 12.5

Deliverable 13

13.a)

```
{
  "hits": {
    "total": {
      "value": 20,
      "relation": "eq"
    }
  }
}
```

Listing 20 – Total hits from request in listing 15

13.b)

```
{
  "hits": {
    "total": {
      "value": 36,
      "relation": "eq"
    }
  }
}
```

Listing 21 – Total hits from request in listing 16

13.c)

```
{
  "hits": {
    "total": {
      "value": 69,
      "relation": "eq"
    }
  }
}
```

Listing 22 – Total hits from request in listing 17

13.d)

```
{
  "hits": {
    "total": {
      "value": 205,
      "relation": "eq"
    }
  }
}
```

Listing 23 – Total hits from request in listing 18

13.e)

```
{
  "hits": {
    "total": {
      "value": 30,
      "relation": "eq"
    }
  }
}
```

Listing 24 – Total hits from request in listing 19

Deliverable 14

```
GET /cacm_english/_search
{
  "query": {
    "function_score": {
      "query": {
        "query_string": {
          "query": "compiler program"
        }
      },
      "linear": {
        "date": {
          "origin": "1970-01",
          "scale": "90d",
          "decay": 0.5
        }
      }
    }
  }
}
```

Listing 25 – Requests for deliverable 14