Information Processing and Retrieval

Instituto Superior Técnico 2020

Lab 4: Topic modeling and concept analysis

In this lab, we will look to ways of improving our IR system. First, we will retrieve the major topics underlying a given collection. Second, we will discover formal and coherent concepts with the aim of categorizing the documents in the collection and aiding retrieval.

To our end, we will continue to use the CFC collection, The CFC collection can be loaded from the file pri_cfc.txt (one document per line).

Attention: the notion of *topic* introduced in this lab differs from the project's notion of *topic*.

1 Topic modeling

Topics capture the hidden structure of a collection. More intuitively they can be used as a way of tagging documents, unraveling hidden knowledge, and reducing the high dimensionality of vector space models whereby a document is seen as a mixture of topics each having a specific weight. Amidst the different algorithms for topic modeling, we will consider Latent Dirichlet Allocation (LDA) for this lab.

LDA is a probabilistic model that assumes each topic is a mixture of terms, and each document is a mixture of topics. Given n documents, m terms and k topics, LDA identifies:

- ψ the distribution of terms per topic
- ϕ the distribution of topics per document

These distributions are controlled by:

- β parameter to control topic-word density
  (high β leads to a higher number of terms per topic)
- α parameter to control document-topic density
  (high α leads to a higher number of topics per document)

1.1

Create a word cloud from the given collection.

```
from wordcloud import WordCloud
wordcloud = WordCloud(background_color="white", max_words=5000,
                      contour_width=3, contour_color='steelblue')
wordcloud.generate(all_docs_text_concatenated) # generate the word cloud
wordcloud.to_image() # visualize the word cloud
```

1.2

Find the top 10 topics in the CFC collection. Plot the top 10 terms per topic.
Consider α = 0.2 and β = 0.5 as default values to LDA, yet change them to assess their impact.

```
from sklearn.decomposition import LatentDirichletAllocation as LDA
from sklearn.feature_extraction.text import CountVectorizer

def print_topics(model, word_vector, n_top_words):
    words = word_vector.get_feature_names()
    for topic_idx, topic in enumerate(model.components_):
        print("\nTopic #%d:" % topic_idx)
        lstwords = [words[i] for i in topic.argsort()[:-n_top_words-1:-1]]
        print(" ".join(lstwords))

number_topics = 10
number_words = 10
alpha = 0.2
beta = 0.5

# Process the collection
count_vectorizer = CountVectorizer(stop_words='english')
doc_term_matrix = count_vectorizer.fit_transform(vector_with_documents_text)

# Create and fit the LDA model
lda = LDA(n_components=number_topics, doc_topic_prior=alpha,
          topic_word_prior=beta, n_jobs=-1)
lda.fit(doc_term_matrix)
print("Topics found via LDA:")
print_topics(lda, count_vectorizer, number_words)
```

1.3

Let us now visualize the properties of the found topics. You can use the pyLDAvis pack to:

- select the top terms for a given topic using different thresholds;
- understand the relationships between the topics (Intertopic Distance Plot).

```
from pyLDAvis import sklearn as sklearn_lda
import pickle, pyLDAvis
LDAvis_data_filepath = os.path.join('./ldavis_prepared_'+str(number_topics))

LDAvis_prepared = sklearn_lda.prepare(lda,doc_term_matrix,count_vectorizer)
with open(LDAvis_data_filepath, 'w') as f:
    pickle.dump(LDAvis_prepared, f)

# load the prepared pyLDAvis data from disk
with open(LDAvis_data_filepath) as f:
    LDAvis_prepared = pickle.load(f)
    pyLDAvis.save_html(LDAvis_prepared, './ldavis_prepared_'+ str(
        number_topics) +'.html')
```

2 Formal concept analysis (FCA)

Concepts in document collections capture relationship between terms/topics and documents. Concepts can be used to support document categorization, guide document navigation, and aid document retrieval.

A formal concept generally corresponds to a subset of terms/topics relevant to a subset of documents. Relevance either corresponds to term presence or scoring above a predefined threshold (Boolean stance on relevance).

Generally, the *extension* of a set of terms/topics corresponds to the set of documents where they occur, while the *intension* of a set of documents is the set of shared terms/topics. The set of all formal concepts – *concept lattice* – can be used to characterize the corpus.

Package to FCA: https://github.com/xflr6/concepts
Installation: you can install concepts using **pip install concepts**

2.1

a) Create a document-topic incidence matrix from the distribution of the 10 topics per document produced in **exercise 1.3**.

```
doc_topic_incidence_matrix= lda.transform(doc_term_matrix)
```

b) Binarize the previous real-valued matrix by considering a threshold of θ=1E-2.

```
bool_data = np.where(topic_data > 0.01, 1, 0)
```

2.2

Run formal concept analysis on the previously binarized document-topic incidence matrix.

```
formatted_matrix = np.where(bool_data==0, '', bool_data)
formatted_matrix = np.where(formatted_matrix=='1', 'X', formatted_matrix)

header = ["topic_"+str(i) for i in range(number_topics)]
indices = ["doc_"+str(i) for i in range(len(textT))]
df = pd.DataFrame(formatted_matrix, index=indices, columns=header)
df.to_csv('df.csv', index=True, header=True, sep=',')

dc = concepts.Context.fromfile("df.csv",frmat='csv')
```

2.3

Explore the extension of specific topic sets and intension of specific document sets.

```
dc.extension(['topic_1','topic_2'])
dc.intension(['doc_1','doc_2'])
```

2.4

Discover the set of all formal concepts present in the collection.

```
for extent, intent in dc.lattice:
    print('%r %r' % (extent, intent))
```

2.5

Visualize the concept lattice associated with the given collection.

```
dc.lattice.graphviz()
```

3 Coherent concept analysis (*optional*)

In contrast with formal concepts, coherent concepts are sensitive to the relevance of each term on a given document, surpassing the need for discretization.

To explore the pros and cons of coherent concept analysis, we suggest you to use the BicPAMS tool on a pre-prepared document-topic relevance matrix to this end.

BicPAMS GUI: `https://www.dropbox.com/s/q0iv2jd2xrkzbgv/bicpams4.0.4gui.jar`

Document-topic file: doc_topic_relevance.csv (in the webpage)

Guidelines: `https://www.youtube.com/watch?v=QWFBwKHGz4o&feature=youtu.be`

3.1

Open BicPAMS, load the doc_topic_relevance file, and preserving default parameters find concepts with varying coherence: constant assumption (3, 4 and 6 symbols) and order-preserving assumption (20 symbols). Visualize the found concepts.