INFORMATION PROCESSING AND RETRIEVAL

INSTITUTO SUPERIOR TÉCNICO 2022

PROJECT v0

Consider the IR task of automatically summarizing the contents of a text document. To answer this task, some IR systems follow an *extractive approach* by selecting the most informative sentences from the original document, and producing a summary where these sentences are ordered by relevance or location in the original document. The fundamental difficulty of this task lies in identifying and ranking the sentences that capture the main ideas behind a document, while ensuring their coherence and minimum redundancy.

Material

BBC News Summary repository contains the document collection and corresponding summaries.

Document collection. The corpus is composed of 2225 journalistic texts collected from the online newspaper BBC News, $D = \{d_1, .., d_{2225}\}$, along five categories: business, entertainment, politics, sports and tech news, $D = D_{business} \cup D_{entertainment} \cup D_{politics} \cup D_{sports} \cup D_{tech}$.

category	**size**	**document** (avg #tokens/text)	**summary** (avg #tokens/text)
business	510	328.88 ± 135.8	139.93 ± 59.4
entertainment	386	330.62 ± 261.5	144.05 ± 124.0
politics	417	453.97 ± 299.8	195.71 ± 139.6
sports	511	329.26 ± 187.8	143.19 ± 80.8
tech	401	502.70 ± 239.6	213.84 ± 111.4

Table 1: BBC News Summary: essential statistics

Summaries. A reference summary is a composition of both informative and non-redundant sentences from the corresponding document using a state-of-the-art extractive approach. The extractive approach relies on a form of kernel text subspace clustering that ensures the centrality and non-redundancy of selected statements by addressing problems of diagonal dominance. Although further details on the reference IR system are not necessary for the accomplishment of the current project, the applied approach is described in

D. Greene and P. Cunningham. *Practical Solutions to the Problem of Diagonal Dominance in Kernel Document Clustering*, ICML 2006.

Please note that the reference extracts, $R = \{r_1, .., r_{2225}\}$, may not correspond to the optimal extracts, yet they offer a stable reference ground.

Download. *BBC News Summary* repository is temporarily made available at: `https://fenix.tecnico.ulisboa.pt/downloadFile/845043405587757/BBC%20News%20Summary.zip` (4.3Mb)

Acknowledgments. BBC (`http://mlg.ucd.ie/datasets/bbc.html`), Kaggle (`https://www.kaggle.com/datasets/pariza/bbc-news-summary`), D. Greene, P. Cunningham, Pariza Sharif

Project goals

The target IR system will be developed in two steps:

1. in the first delivery, summarization will be done unsupervisedly in the document collection D. Extracts are uniquely used as reference summaries for evaluation purposes;
2. in the second delivery, we will explore the structure of the corpus. Furthermore, we will use the reference extracts as feedback to guide the learning process.

General guidelines

- **Deadlines** for project deliveries available at the course's webpage;
- **Grading** of project deliveries: 50% first delivery + 50% second delivery;
- Please always **consult the FAQ** on the course's webpage *before posting questions to your faculty hosts*. The subject of e-mails to your faculty should be preceded by [PRI project].
- **Delivery**: source code, report, and Jupyter notebook demo highlighting major facilities;
- The **templates** for the reports and notebooks are listed in the course's webpage;
- We suggest **reports** to be organized in two parts: i) major decisions placed along the implementation of the summarization tasks; and ii) point-by-point direct answering of the posted questions. Answers must be supported by quantitive empirical evidence;
- The **page limits** for each delivery report are 6 pages;
- Students should disclose the contribution of each member at the start of the project reports whenever efforts are unequal. *Marks are individual*;
- **Submissions**: Gxx.ZIP file via Fenix, where xx corresponds to your group number. The ZIP file should contain three files: Gxx_code.ZIP with your source code, Gxx_notebook.ipynb with your notebook demo, and Gxx_report.pdf with your report;
- Please note that it is possible to submit files several times on Fenix. Only the last submission is stored in Fenix. You can submit versions ahead to prevent late-time problems.

Copy and plagiarism

- The project code and reports will be subjected to strict copy and plagiarism checkers;
- If copy is detected after manual clearance for any of the above cases, the registration in PRI is nullified and IST guidelines will apply. These guidelines are also valid for students sharing the copied contents, irrespectively of the underlying intent.

Complementary notes

- the target IR system should be developed in Python. Students are free to use available Python libraries to program the IR system, including scikit-learn, spacy or nltk libraries;
- students can use external resources (including dictionaries, thesauri, ontologies, etc...) to guide text processing and scoring as long as their use is clearly identified;
- students can implement advanced summarization solutions provided in Kaggle or described in scientific literature. Yet, as the project will be solely evaluated on the ability to answer the requirements and questions in this statement, this direction is discouraged.

Project I (*first delivery*)

Topics: IR evaluation, indexing, IR models, ranking, text processing

Task: On the first part of the project, students should be able to establish a sound solution for the content summarization task using *classic IR querying* principles.

Functionality

Program the following functions:

1. indexing(D,args)

@input	document collection D and optional arguments on text preprocessing
@behavior	preprocesses each document in D and builds an inverted index with the necessary statistics for the subsequent summarization functions
@output	tuple with the inverted index I, indexing time, and memory space

2. ranking(d,p,l,o,I,args)

@input	document d, optional maximum number of sentences (p), optional maximum summary length l in characters, order of presentation o (appearance in text *vs* relevance), inverted index I, and arguments on IR models
@behavior	preprocesses d, assesses the relevance of each sentence in d against I using classic vector space models or probabilistic retrieval models according to args, and presents them in accordance with p, l and o
@output	summary of document d

3. visualize(d,p,l,o,I,args)

@input	*idem*
@behavior	using the previous function, it offers an HTML visualization of the full original text with summary marks (e.g. bold). When o sorting considers text relevance, coloring, size or tickness options should be explored along the summary marks
@output	HTML text display with sentence highlighting

4. evaluation(D_{set},S^I_{set},I,args)

@input	a set of documents $D_{set} \subseteq D$ (e.g. single document, category documents, all collection), the corresponding reference extracts R_{set}, inverted index I, and optional arguments on processing, IR models and evaluation options
@behavior	assesses the produced summaries against the reference ones using the auxiliary function (b) and the target evaluation criteria
@output	extensive evaluation statistics, including recall-and-precision curves at different p-or-l summary sizes, MAP, cumulative gains and efficiency

Summarization options

Multiple solutions can be envisioned to answer the summarization task. Term-and-document frequencies can be assessed at sentence-and-document level or, in alternative, at document-and-collection levels. Depending on your choices you may need to adapt the (a) function in order to produce a dictionary of inverted indices, one per document.

Your summarization system should be able to support TF, TF-IDF, and BM25. Do not forget that the average length parameter in BM25 can as well correspond to sentence or document level stances. BM25 should be considered with the default parameters, k_1=1.2 and b=0.75.

The faculty hosts are unable to offer precise clues on the design of solutions for the summarization task. Get creative and, if you are pursuing more than one solution, you can compare the accuracy of the tested solutions. Otherwise, provide the rationale for your choice.

Text preprocessing

Your vector space should not only include words but also: i) bigrams; and ii) noun phrases, i.e. word sequences matching a parts-ofspeech regular expression pattern like $\{$(<JJ>* <NN.*>+ <IN>)? <JJ>* <NN.*>+g+$\}$. Assess the impact of including bigrams and noun phrases.

Assessing the impact of alternative processing options, such as stop word removal or lemmatization versus stemming choices, is considered to be out of the scope of the project. Place preprocessing choices and fix them.

System evaluation

Performance improvements on the IR system should be primarily based on average uninterpolated precision. In addition, the F-β measure is also suggested. This measure, a function of recall and precision, uses the free β parameter to determine the relative weight of recall and precision. For this project, a value of β=0.5 has been chosen, corresponding to an emphasis on precision (β=1 is neutral). Provide a brief rationale for the selected measures in the report.

As reference, length parameters can be fixed as p=8 and l=500. Note, nevertheless, that precision and recall can be assessed at different thresholds to study the impact of summary length.

Evaluation can be conducted for a single document or a set of documents. In this latter case, multiple estimates are acquired. Particular attention should be placed to the presentation of results from document sets by selecting adequate statistics or chart visualizations.

Statistical comparisons should be undertaken to assess the performance of the different IR models, preprocessing options and, if available, summarization solutions. Finally, the ability to summarize documents can be assess for the two journals or across the thematic sections.

Handling ranking differences

The diversity of options associated with the design of the target IR system (e.g., text processing and IR models) can lead to arbitrarily-high differences on the produced summaries. In the presence of multiple sentence ranks, CombSUM, CombMNZ or Reciprocal Rank Fusion (RRF) can be used for placing consensus. In this context, the ranking outputs produced under different options can be combined,

$$\mathrm{RRF}(s) = \sum_{f \in F} \frac{1}{\mu + \mathrm{rank}(f(s))},$$

where F is the set of available options and μ is a constant that can be fixed at μ=5. Under this score, you can assess whether consensus improves retrieval or, in alternative, whether there are specific options yielding optimal performance.

Handling sentence diversity

A notable problem of previous solutions is the potential high redundancy among the selected sentences. In the paper entitled "*The use of MMR: diversity-based reranking for reordering documents and producing summaries*", Carbonell and Goldstein propose Maximal Marginal Relevance (MRR) to increase sentence diversity. The proposed method iteratively selects the most informative sentence, adding it to the set of sentences in the summary and removing it from the document. Sentence selection is accomplished using the MMR score that simultaneously attempts to select sentences that are relevant and dissimilar to the sentences that have been selected so far (non-redundant),

$$\text{MMR}(s) = (1-\lambda) \times sim(s,d) - \lambda \sum_{v \in S} sim(s,v). \tag{1}$$

where S is the sentences composing the summary at a given time and sim corresponds to the cosine similarity. Fix a text processing option and IR model. Given a document, assess the differences produced by varying λ. Assess the impact of MMR by comparing it against our previous redundancy-unaware summarization system. Compare their performance on the overall document collection.

Questions to explore in the report

Guidance on angles to conduct your analyzes and write your report:

1. Characterize the corpus D and summaries R. What is the distribution of informative terms before and after text processing?
2. How does the developed summarization system perform for the full collection? And within each category? Any intuition for the observed differences?
3. Consider the Maximal Marginal Relevance (MMR) stance. How λ impacts the redundancy and accuracy (against ideal extracts) of the summaries? Should λ be a fixed threshold or depend on the provided topic document (d-specific)?
4. Given specific length thresholds (p and l), is the system better at providing recall or precision guarantees? How would you fix the length thresholds if the user prefers: minimizing false positives, minimizing false negatives, or maximizing true positives?
5. Does the inclusion of phrases and bigrams produces a positive impact on summarization?
6. How IR models affect retrieval? Is Reciprocal Rank Fusion (RRF) useful to aid decisions?
7. Lengthier sentences are more verbose and therefore have higher chance of being judged as relevant for a randomly selected topic. If this is an undesirable bias of your summarization system, how would you extend your system to handle this bias?

Project II (*second delivery*)

Topics: clustering, classification, page ranking, IR evaluation

In this second stage of the IR system development, we consider three major improvement strategies. *First*, we will see whether the unsupervised organization of the corpus can offer guidance. *Second*, we will assess the aided value from relevance feedback. *Finally*, we will check whether modeling sentence interdependencies yields improvements on the target summarization.

Grading: 25% clustering approach + 40% relevance feedback + 35% graph approach

A) Clustering approach: corpus organization

Forms of unsupervised corpus exploration, such as topic modeling and concept analysis, can be placed to guide IR tasks. Here, we will assess the role of clustering. The primary goal is to understand the organization of contents in D, so that principled improvements can be envisioned.

Functionalities to implement

1. clustering(D,args)

@input	document collection D, optional arguments on clustering analysis
@behavior	selects an adequate clustering algorithm and distance criteria to identify the best *number of clusters* for grouping the D documents
@output	clustering solution

2. interpret(cluster,D,args)

@input	cluster and document collection D
@behavior	describes the documents in the cluster considering the most informative terms in the centroid (weighted with IDF to avoid the presence of stop words), where the cluster's centroid is given by the *median* criteria
@output	order set of most informative terms (cluster description)

3. evaluate(D,args)

@input	document collection D
@behavior	evaluates the produced solution recurring to the clustering function using *internal* criteria (e.g., silhouette, dendrogram visuals) and *external* criteria (considering categories as reference labels)
@output	clustering evaluation

Questions to explore

1. What is the (hypothesized) number of document clusters? Are the clusters cohesive? And well separated? Are clusters aligned with the underlying document categories?
2. What the clustering reveals regarding the conceptual organization? How could the (clustering-based) organization of the collection be used to guide the targeted summarization task?

B) Supervised approach using reference summaries

In the presence of relevance feedback from reference extracts, summarization can be formulated as a supervised learning-to-rank task. In this case, the sentences from a given document can be represented through a set of descriptive features (e.g., position in the paragraph, ranking scores), and a learned mapping between the features of a sentence and its presence or absence in the reference summary. To this end, the summarization system can be trained using using sentences from documents in D_{train} collection and the corresponding reference extracts, R_{train}.

Functionalities to implement

1. feature˙extraction(s,d,args)

@input	sentence s and the enclosed document d
@behavior	extracts features of potential interest considering the characteristics of the sentence and wrapping document
@output	sentence-specific feature vector

2. training(D_{train},R_{train},args)

@input	training document collection D_{train}, reference extracts R_{train}, and optional arguments on the classification process
@behavior	learns a classifier to predict the presence of a sentence in the summary
@output	classification model

3. classify(s,d,M,args)

@input	sentence s in document $d \in D_{test}$, and classification model M
@behavior	estimates the probability of sentence s in d be selected as part of the document summary using M
@output	probabilistic classification output

4. evaluate(D_{test},R_{test},args)

@input	testing document collection D_{test}, corresponding reference extracts R_{test}, and summarization-and-evaluation arguments
@behavior	evaluates the summarization system in the presence and absence of relevance feedback from reference extracts, using previous functions
@output	performance statistics on the behavior of the aided summarization system

Feature extraction

With the aim of describing sentences, the following features should be considered:

- position of the sentence in the enclosed paragraph and document;
- cosine similarity of the sentence against document contents using TF, TF-IDF and BM25.

Students wanting to go an extra mile can explore and propose other features of potential interest.

Performance evaluation

To assess the impact of reference extracts on the target IR system, compare the performance of the summarization system behavior in the absence and presence of feedback from reference extracts recovering IR evaluation gathered in the first delivery.

Classification models

Select and compare the performance of either a single or two classifiers. Possibilities: i) Bayesian classifiers, such as naïve Bayes; ii) analogizers, such as kNN (k-Nearest Neighbor); iii) associative classifiers, such as random forests; or iv) neural networks, such as a multi-layer perceptron. The hyperparameterization of the selected classifiers is optional. Your implementation can fully rely on packages with classification and hyperparameterization facilities, such as scikit-learn.

Extension towards ranking

Previous classifiers are only able to determine whether a given sentence s is relevant or not for a given document d. As such, they are as-is unable to rank sentences to, for instance, sort them in order of relevance. Consider the following strategies to extend the system to perform ranking:

- using the probabilistic outputs of the classification system, P(relevant | d,q), as a rough scoring criterion for ranking documents;
- using the introduce classification system for removing non-relevant documents, followed by the application of traditional scoring criteria as given for instance by the application of cosine operator over the relevant documents.

Select one of the introduced ranking strategies and briefly discuss its merits and limitations.

Questions to explore

1. Does the incorporation of relevance feedback from ideal extracts significantly impact the performance of the IR system? Hypothesize why is that so.
2. Are the observed differences uniform across documents? Are the learned models able to generalize from one category to another? Justify.
3. Which features appear to be more relevant to the target summarization task?

C) Graph ranking approach: sentence centrality

Several previous studies have proposed to leverage graph centrality metrics to aid summarization tasks. Methods such as TextRank[1] bring principles from PageRank towards sentence extraction. Essentially, these methods represent a document as a graph, where nodes correspond to sentences and edges encode relationships between sentences based on their content and location. For instance, an edge can encode the cosine similarity between vector representations of two sentences. In this context, students should develop a PageRank-based approach to aid summarization.

Functionalities to implement

1. build_graph(d,sim,θ,args)

@input	document d in D, similarity criteria, and minimum similarity threshold θ
@behavior	computes pairwise similarities for the sentences in d using the inputted similarity criterion; maps the pairwise relationships into a weighted undirected graph; and applies θ threshold to remove edges with low similarity
@output	undirected graph capturing sentence similarity relationships

[1]`http://web.eecs.umich.edu/~mihalcea/papers/mihalcea.emnlp04.pdf`

2. undirected_page_rank(d,p,sim,θ,args)

@input	document d, number of top sentences to return (p), and parameters related with the graph construction and page rank algorithms
@behavior	applies a modified page rank* over undirected graphs for sentence ranking
@output	summary composed by the sorted top-p sentences (descending score order)

   *candidate sentences should be ranked according to a variation of the PageRank algorithm for undirected graphs by scoring each candidate according to an iterative procedure,

$$\mathrm{PR}(s_i) = \frac{\lambda}{N} + (1-\lambda) \times \sum_{s_j \in \mathrm{Links}(s_i)} \frac{\mathrm{PR}(s_j)}{\mathrm{Links}(s_j)}$$

   where $s_1, \ldots, s_N$ are the sentence candidates, $\mathrm{Links}(s_i)$ is the set of candidates similar to s_i (nodes linked to s_i). Consider a uniform residual probability per node and $\lambda = 0.15$.

   The iterative procedure should be applied up to a maximum um 50 iterations.

   You can use existing implementations of PageRank (e.g., NetworkX package), yet keep in mind that you should use a variant for unweighted graphs given by the previous equation.

3. evaluate(D_{set},R_{set},args)

@input	set of documents, corresponding extracts, and evaluation options
@behavior	evaluates the summarization system using first delivery principles
@output	evaluation statistics (e.g., recall-precision curves, MAP)

***Optional* graph ranking improvements**. Although not part of the project evaluation, students that want to go an extra mile can attempt to improve the graph-ranking method by considering a prior probability per sequence (e.g., sentence scores from the original IR system) or weight edges in the graph (e.g., raw similarity scores). In this case, PageRank can be iteratively computed as

$$\mathrm{PR}(s_i) = \lambda \times \frac{\mathrm{prior}(s_i)}{\sum_{s_j} \mathrm{prior}(s_j)} + (1-\lambda) \times \sum_{s_j \in \mathrm{Links}(s_i)} \mathrm{PR}(s_j) \times \frac{\mathrm{weight}(s_j, s_i)}{\sum_{s_k \in \mathrm{Links}(s_j)} \mathrm{weight}(s_j, s_k)}.$$

Questions to explore

1. Does the graph ranking method aid summarization? Hypothesize why is that so.
2. Upon the analysis of some graphs, are there sentences with delineate higher centrality score? How does sentence ranking performance vary with θ?

END