Alma Mater Studiorum · Università di Bologna

SCUOLA DI SCIENZE

Corso di Laurea in Informatica

SUBTOPIC-ORIENTED BIOMEDICAL SUMMARIZATION USING PRETRAINED LANGUAGE MODELS

Relatore:
Chiar.mo Prof.
DANILO MONTESI

Correlatore:
Dott. FLAVIO BERTINI

Presentata da:
TIAN CHENG XIA

Sessione II
Anno Accademico 2022-2023

"The principal difficulty in your case, ... lay in the fact of there being too much evidence. What was vital was overlaid and hidden by what was irrelevant. ..."

Sherlock Holmes in *The Naval Treaty*,
Arthur Conan Doyle

Abstract

The ever-growing number of publications in the biomedical field is causing difficulties in finding insightful knowledge. In this work, we propose a subtopic-oriented summarization framework that aims to provide an overview on the state-of-the-art of a given subject. The method we propose clusters the papers retrieved from a query and then, for each cluster, extracts the subtopics and summarizes the abstracts. We conducted various experiments to select the most appropriate clustering approach and concluded that the best choices are MiniLM for text embedding, UMAP for dimensionality reduction and OPTICS as clustering algorithm. For summarization, we fine-tuned both general-domain and biomedical pretrained language models for the task of extractive summarization and selected Longformer as the most suited model. Experimental results on multi-document summarization datasets show that the proposed framework improves the overall recall of the generated summary with a small decrease in precision, which corresponds to slightly longer summaries but closer to the ground truth.

Sommario

Il crescente numero di pubblicazioni in ambito biomedico sta rendendo sempre più complicato individuare informazioni rilevanti. In questa tesi proponiamo un framework che utilizza sotto-tematiche per generare riassunti con l'obiettivo di fornire una panoramica sullo stato dell'arte di un dato argomento. Il metodo proposto individua i cluster dato un gruppo di pubblicazioni e, per ciascuno di questi, estrae le sotto-tematiche e genera un riassunto basandosi sui sommari. Abbiamo condotto diversi esperimenti per selezionare il metodo di clustering più appropriato e abbiamo concluso che le scelte migliori sono: MiniLM per l'*embedding* del testo, UMAP per la riduzione della dimensionalità e OPTICS come algoritmo di clustering. Per generare i riassunti, abbiamo selezionato diversi modelli di linguaggio pre-addestrati sia in ambito generale che in ambito biomedico e li abbiamo specializzati per generare riassunti in maniera estrattiva. Dai risultati abbiamo selezionato Longformer come il modello più adatto ai nostri scopi. Risultati sperimentali su dataset multi-documento mostrano che il framework migliora i valori di *recall* del riassunto generato con una piccola perdita in *precision*. Ciò corrisponde a riassunti lievemente più lunghi ma in grado di catturare più concetti.

Contents

List of Tables

1 Introduction

PubMed[1] is a search engine that indexes biomedical and life science related papers. The database is updated daily and, at the time of writing, contains more than 38 million publications. This overwhelming amount of information may cause a significant slowdown in making new discoveries, and a solution able to present the knowledge related to a query could help researchers gather the necessary information more quickly.

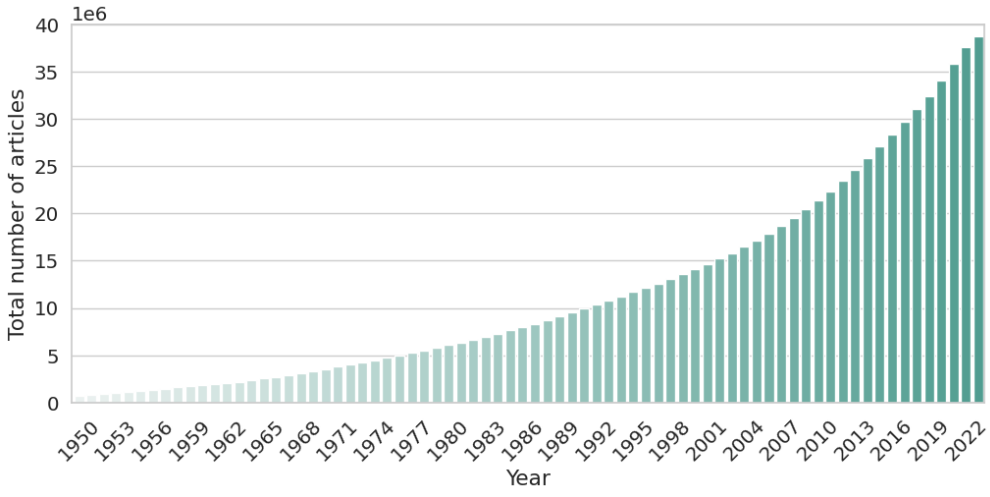

Figure 1.1: Total number of articles available on PubMed by publication year

For this purpose, in this work we present a possible approach to analyze the knowledge available on PubMed and present an overview on the state-of-the-art to the user. The method we propose individuates the subtopics of a given group

[1] `https://pubmed.ncbi.nlm.nih.gov`

of papers by clustering their abstracts. Then, the subtopic of each cluster is determined using a bag-of-words approach and a summary is created by selecting the most important sentences of the abstracts. Following the recent developments, for both the clustering and summarization phase, we experiment the performance of Transformer-based pretrained language models.

In the next Chapter, we discuss other automatic summarization work. In Chapter 3, we give the definition of language model, present the Transformer architecture and discuss existing general-domain and biomedical pretrained language models. In Chapter 4 and Chapter 5, we respectively define the task of clustering and summarization, and present the experiments we conducted to select the best approach for our work. In Chapter 6, we present the method we propose and report the experimental results we obtained. In Chapter 7, we conclude this thesis with some final observations and discuss possible future work.

Code availability

We make the implementation of our method and the code to reproduce the experiments of this work publicly available.

Code to evaluate clustering methods (Chapter 4)	`http://smartdata.cs.unibo.it/datasets#PubMedSum`
Code to evaluate pretrained language models on summarization (Chapter 5)	`https://github.com/NotXia/biomed-ext-summ`
Weights of the best summarization models we trained (Chapter 5)	`https://huggingface.co/NotXia/longformer-bio-ext-summ` `https://huggingface.co/NotXia/pubmedbert-bio-ext-summ`
Implementation of the method we propose, the code to evaluate it and a prototype to present the result to the end user (Chapter 6)	`https://github.com/NotXia/pubmed-summ`

2 Related work

2.1 Automatic text summarization

Automatic text summarization is an active field of research with a large spectrum of proposed methods. Existing solutions in both the general-domain [34, 35, 41] and the biomedical field [61, 12] often use as possible architecture:

Graph-based algorithms A graph where vertexes are sentences and edges represent a similarity metric. The summary is generated by selecting the first k sentences based on a centrality score.

Recurrent Neural Networks (RNNs) As they are designed to process sequential data, RNNs and its variants can be used in summarization. A common architecture is based on an encoder-decoder structure and attention mechanism. However, a significant disadvantage of RNNs is that they are unable to be parallelized as each step depends on the previous one.

Convolutional Neural Networks (CNNs) Convolutions, typically used in computer vision, can be employed in natural language processing to capture semantic and syntactic features. A possible approach consists in using multiple filters with different size followed by pooling layers. CNNs, differently from RNNs, are easier to parallelize. On the other hand, as there is no recurrence, the context is limited to the receptive field of the convolutions.

Graph Neural Networks (GNNs) Instead of feeding the plain corpus sequence to a neural network, a GNN uses a graph to produce a summary.

Starting from a graph built using some arbitrary criteria, it is first encoded and then passed through a GNN which leverages the input using, for instance, convolutions or an attention mechanism.

Point-generator networks [58] The summary is created using two operations: pointing and generating. Pointing allows to copy words from the input corpus to the summary. Generating allows to select words from a vocabulary to add in the summary. Moreover, to prevent repetitions, a coverage mask is employed to keep track of the words covered by the attention mechanism.

Transformer models Recent advancements in natural language processing often use pretrained language models based on the Transformer architecture. As this is an important component for this work, we discuss Transformers and language models more in detail in Chapter 3.

2.2 Subtopic-oriented summarization

In the literature, other work uses a similar approach based on subtopic-oriented summarization as we do in this thesis.

In the biomedical field, both Nasr Azadani et al. [47] and Moradi [46] propose a graph-based method that use itemset mining and clustering to extract subtopics and partition the vertexes before creating the summary. Mishra et al. [45] also propose a graph-based summarization model which clusters using UMLS concepts [8] extracted using MetaMap [5].

In the general domain, other similar approaches have been explored. Zhang et al. [65] propose a graph-based algorithm that uses a custom centrality score based on local and global features. The method iteratively forms clusters with the most central sentence and its neighbors within a threshold. Dai et al. [14] also use a graph-based algorithm with fuzzy clustering and strategies to prevent local and global redundancies. Gong et al. [23] consider the subtopics as a probability distribution and use statistical methods to extract and use them to weight the sentences. Mei et al. [43] propose a method that clusters the sentences based

on their similarity. Then, creates the summary by selecting sentences based on a position score with respect to the original documents and a cluster distance score. Zheng et al. [68] embed the input sentences using a RNN and cluster them to individuate the subtopics. The summary is created by using a saliency score for subtopics and sentences. Dong et al. [21] extract subtopics by building a paragraph-level graph. Then, create the summary by solving an optimization problem that aims to minimize the distance to the original document and maximize the topic diversity.

3 Pretrained language models

Pretrained Language Models (PLMs) are models for sequence processing pretrained on a large amount of textual data. These models can be specialized for downstream tasks with additional steps of fine-tuning, without the need to significantly change the overall architecture.

In this chapter, we first give the definition of language model and then describe the Transformer architecture on which recent PLMs are built on. At last, we present the existing general-domain and biomedical pretrained models we will use in this work.

3.1 Language models

Language Models (LMs) [33] are generative models that assign a probability to a sequence of words. More formally, LMs compute a probability distribution:

$$\mathbb{P}(w \mid k)$$

where w is a word and k is the prior knowledge (e.g. previous words of a sentence).

An example of a statistical LM is the n-gram model which assumes that the probability of a word depends only on a fixed number of previous words (Markov assumption). Specifically, n-gram models approximate the conditional probability of a word given its full context to a window of $n-1$ words:

$$\mathbb{P}(w_i \mid w_1, \ldots, w_{i-1}) \approx \mathbb{P}(w_i \mid w_{i-n+1}, \ldots, w_{i-1})$$

To compute this probability, the simplest and most straightforward approach is

based on the maximum likelihood estimation:

$$\mathbb{P}(w_i \mid w_{i-n+1}, \ldots, w_{i-1}) = \frac{C(w_{i-n+1} \ldots w_{i-1} w_i)}{\sum_w C(w_{i-n+1} \ldots w_{i-1} w)}$$

where $C(s)$ counts the number of occurrences of the string s in the training corpus.

In recent years, deep learning approaches have been developed to create LMs. These models, usually referred as Large Language Models (LLMs), are commonly built using networks with millions or billions of parameters and are able to achieve state-of-the-art results in many natural language processing tasks.

3.2 Transformer architecture

Recent pretrained language models are mostly based on the Transformer [59] architecture which proved to have more reliability on long term dependencies while being more parallelizable compared to recurrent neural networks.

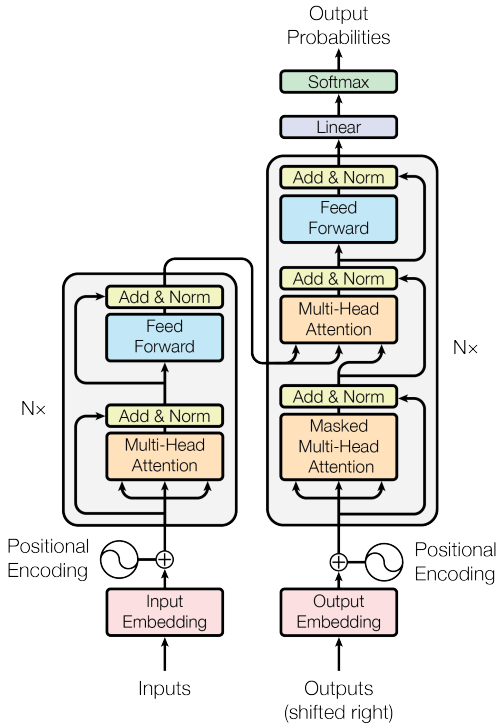

Figure 3.1: Transformer architecture. Source: [59]

As depicted in Figure 3.1, Transformers are composed of a stack of encoders followed by a stack of decoders and heavily utilize the self-attention mechanism. The main components of a Transformer are:

Input and positional embedding The textual input is first embedded using a learnt static embedding of size d_{model}. In addition, as the model does not have recurrence, positional information obtained through sinusoidal functions are injected into the embeddings.

Multi-head attention The multi-head attention mechanism allows the model to attend at different positions of the input at the same time. Let d_k be the size of the attention keys and queries, d_v the size of the attention values and h the number of heads of the multi-head attention mechanism. A single attention head uses a dot-product attention scaled by a factor of $\frac{1}{\sqrt{d_k}}$ to prevent the problem of vanishing gradient. In addition, queries, keys and values are projected using a learnt projection different for each head:

$$\texttt{attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V$$

$$\texttt{head}_i(Q, K, V) = \texttt{attention}(QW_i^Q, KW_i^K, VW_i^V)$$

where Q, K and V are respectively the matrices for queries, keys and values. $W_i^Q \in \mathbb{R}^{d_{\text{model}} \times d_k}$, $W_i^K \in \mathbb{R}^{d_{\text{model}} \times d_k}$ and $W_i^V \in \mathbb{R}^{d_{\text{model}} \times d_v}$ are the learnt projection matrices for queries, keys and values. The multi-head attention is defined as the concatenation of the attention heads with an additional final projection:

$$\texttt{multi\_head\_attention}(Q, K, V) = \texttt{concat}(\texttt{head}_1, \ldots, \texttt{head}_h)W^O$$

where $W^O \in \mathbb{R}^{hd_v \times d_{\text{model}}}$ is the learnt projection matrix for the output.

Additionally, an optional mask can be applied to prevent future tokens to be accessed during training.

Feed-forward layer The feed-forward layer is simply composed of two linear transformations and an activation function (usually ReLU or its variants).

The encoder stack extracts the relevant features of the input sequence. Each encoder takes as input the output of the previous encoder and passes it through a multi-head attention and feed-forward sub-layers.

The decoder stack takes as input the result of the encoder and the previously outputted token, and autoregressively generates the next token. First, the previous token is passed through a masked multi-head attention which prevents the model to attend at future tokens during training. Then, the output is used as the query of a second multi-head attention, while keys and values are taken from the encoder stack. The output of the multi-head attention sub-layers is then passed through a feed-forward sub-layer. The next token is determined by passing the result of the decoder stack through a softmax layer.

It must be noted that Transformer based models do not necessarily utilize the full architecture. Depending on the task, a model can be classified as:

Encoder-decoder Commonly used for sequence-to-sequence models where the input and output are both sequences (e.g. machine translation).

Encoder-only Suited for extracting features from the input sequence. Usually used for classification tasks.

Decoder-only Used for auto-regressive models that are only able to attend at previously generated tokens. Usually used for next token prediction.

3.3 General-domain pretrained models

For our work, we will only focus on encoder-only models. In this section, we present the most common general-domain encoder-only models.

BERT

BERT (Bidirectional Encoder Representations from Transformers) [19] is a language representation model based on a bidirectional Transformer encoder-only architecture that allows the model to attend at every position of the input.

BookCorpus [69] and English Wikipedia are used for pretraining and the objectives are the Masked Language Model (MLM) and Next Sentence Prediction (NSP) tasks. MLM consists in making the model predict randomly masked tokens from the input. NSP consists in predicting if a sentence `B` follows a sentence `A`.

During sequence processing, WordPiece, a subword-level tokenizer, is used to embed the input. Moreover, a classification token (`[CLS]`) is added in front of the input to hold the representation of the entire sequence for classification tasks. If a sentence pair is provided as input, a separator token (`[SEP]`) and segment embeddings are employed to distinguish between sequence `A` and `B`.

RoBERTa

RoBERTa (Robustly optimized BERT approach) [40] is a modified version of BERT, pretrained using different training tasks and a larger batch size. Instead of using a subword-level tokenizer, a byte-level Byte-Pair Encoding is used. The input is provided with a FULL-SENTENCES approach that consists in feeding the model with continuous sentences from one or more documents (i.e. at the end of a document, it continues with the next one). The NSP task and segment embeddings are removed, while, for the MLM task, RoBERTa uses a dynamic masking mechanism where the mask is generated at input time, opposed to BERT's static masking created at preprocess time.

DistilBERT

DistilBERT [56] is a model obtained through knowledge distillation with BERT as the teacher model. DistilBERT maintains BERT's architecture but halves the number of layers and removes segment embeddings. During training, three losses are used: a distillation loss over the probability distributions of the teacher and the student, BERT's standard MLM loss and a cosine embedding loss to align the direction of the teacher's and student's outputs.

Longformer

Longformer [7] is based on RoBERTa and introduces a modified version of the Transformer self-attention mechanism.

The original self-attention mechanism has a quadratic space and time complexity, which makes it impractical to handle long sequences. To solve this limitation, Longformer introduces an attention pattern that scales linearly with the input sequence based on a local and a global context. The local attention uses a sliding window, making each token attend to its surrounding within the window. The global attention works as the standard attention mechanism, allowing a token to attend at every position and to be attended by all other tokens. To maintain the linear complexity, the number of tokens with global attention should be negligible compared to the length of the sequence.

3.4 Biomedical pretrained models

Biomedical models are mostly trained using data from PubMed or Semantic Scholar[1]. In this section, we present the biomedical models that we will use for our experiments.

BioBERT

BioBERT [36] uses BERT's weights and tokenizer as baseline and is further pretrained on a biomedical corpus composed of PubMed abstracts and PubMed Central articles. The resulting model was tested on the tasks of named entity recognition, relation extraction and question answering.

Clinical BERT

Clinical BERT and Clinical BioBERT [2] are models based respectively on BERT and BioBERT, further pretrained on MIMIC-III [31], a database containing

[1] `https://www.semanticscholar.org/`

clinical data of ICU patients. The resulting model was tested on the tasks of named entity recognition and natural language inference.

BlueBERT

BlueBERT [50] is a model based on BERT's weights, additionally pretrained on PubMed abstracts and MIMIC-III. This work defines BLUE, a benchmark containing tasks on sentence similarity, named entity recognition, relation extraction, document classification and natural language inference.

SciBERT

SciBERT [6] uses the same architecture of BERT and is pretrained from scratch on 1.14 million computer science and biomedical full-text papers from Semantic Scholar. It also introduces SciVocab, a vocabulary for WordPiece built on a scientific corpus. The resulting model was tested on the tasks of named entity recognition, PICO extraction, text classification, relation extraction and dependency parsing.

PubMedBERT

PubMedBERT [25] is based on the architecture of BERT and is pretrained from scratch on abstracts from PubMed and full-texts from PubMed Central. It also generates a new WordPiece vocabulary based on the same training corpus. This work introduces and evaluates its model on a benchmark named BLURB, which includes named entity recognition, PICO extraction, relation extraction, sentence similarity, document classification and question answering.

BioLinkBERT

BioLinkBERT is the biomedical version of LinkBERT [64], a model based on BERT and pretrained using a new objective based on cross-document information. The training corpus of LinkBERT is seen as a graph of documents linked by

hyperlinks or references. The objectives for the model are the standard BERT's MLM and a new task named Document Relation Prediction (DRP) which requires the model to classify the relationship between two input segments as contiguous, random or linked. The resulting model was tested on the BLURB benchmark and additional question answering datasets.

BioMed-RoBERTa

BioMed-RoBERTa [27] uses RoBERTa's weights and continues the pretraining phase on a corpus of 2.68 million full-text scientific papers from Semantic Scholar. The resulting model was tested on the tasks of named entity recognition, relationship extraction and sentence classification.

DistilBioBERT

DistilBioBERT [54] is a distilled version of BioBERT obtained using the same losses of DistilBERT. The resulting model was tested on the tasks of named entity recognition, relationship extraction and question answering.

Clinical-Longformer

Clinical-Longformer [37] is based on Longformer's weights and further pretrained on the MIMIC-III database. The resulting model was tested on the tasks of named entity recognition, question answering and document classification.

4 Text clustering

Clustering is an unsupervised task that aims to find subgroups (clusters) in a dataset. Based on a similarity function, elements within a cluster should be similar to each other and different to elements of other clusters.

In this chapter, we present our choices for text clustering. We first present common word embedding methods, dimensionality reduction techniques and clustering algorithms respectively in Sections 4.1, 4.2 and 4.3. In Section 4.4, we describe a topic extraction method. In Section 4.5, we present metrics to evaluate clustering results. Finally, in Section 4.6, we present the experiments we conducted to select the best clustering approach.

4.1 Word embeddings

Word embeddings are methods to convert a text corpus into a vector space. The embedding should be able to capture meaningful features of the text as it usually improves the performance of subsequent elaboration steps.

4.1.1 Bag-of-words

Bag-of-words represents a document based on the occurrences of its tokens. To parse n documents, the algorithm uses a learnt or user defined dictionary of w words and encodes the documents into a usually sparse $n \times w$ matrix.

To refine the encoding and remove unnecessary information, many techniques can be applied. For instance:

Words removal Words that do not contribute to the representation can be removed. This includes stop words or common words across all documents.

Stemming Inflected words are reduced to their root form using heuristics (e.g. removing the "s" of the plural form).

Lemmatization Words are reduced to their canonical form (e.g. synonyms are replaced with a dictionary lookup). Differently from stemming, lemmatization considers the context of each word.

To normalize the encoding, Term Frequency-Inverse Document Frequency (TF-IDF) can be applied. Given a set of documents D and a vocabulary V, TF-IDF represents a term $t \in V$ of a document $d \in D$ by computing its term frequency TF and inverse document frequency IDF:

$$\mathrm{TF}(t, d) = \frac{f_{t,d}}{T_d}$$

$$\mathrm{IDF}(t, D) = \log\left(\frac{|D|}{|\{d \in D \mid t \in d\}|}\right)$$

where $f_{t,d}$ is the number of occurrences of the word t in the document d and T_d represents the number of words in the document d. Then, TF-IDF is computed as follows:

$$\text{TF-IDF}(t, d, D) = \mathrm{TF}(t, d) \cdot \mathrm{IDF}(t, D)$$

IDF can be seen as a weight applied to TF to filter out common words.

4.1.2 Pretrained embeddings

Pretrained embeddings [13] refer to text embedding methods that are pretrained on a large textual corpus using traditional machine learning techniques or shallow neural networks. This family of models usually relies on the distributional hypothesis [28], which states that words occurring in the same context tend to be related.

An important property of these embeddings is their capability to encode the semantics of words by creating an embedding space where word vectors are placed

in a meaningful manner. For this reason, in the vector space, related concepts are encoded with similar directions and algebraic operations produce semantically coherent results (e.g. $\mathit{Paris} - \mathit{France} + \mathit{Italy} = \mathit{Rome}$).

Word2vec

Word2vec [44] is trained using a shallow neural network with a single projection layer and has been proposed with two variants of the training objective. Given a vocabulary V, a context C_w of surrounding words of $w \in V$, the size of the embeddings N and the projection layer of the neural network $E \in \mathbb{R}^{|V| \times N}$; an embedding can be learnt as a continuous bag-of-words model or a continuous skip-gram model:

Continuous bag-of-words is a model that predicts a word w given its neighboring one-hot encoded tokens within a context C_w. Each input is passed through the same projection layer and the final vector is obtained as their average. Therefore, the objective is to maximize the log-likelihood:

$$\sum_{w \in V} \log \mathbb{P}(w \mid C_w)$$

Continuous skip-gram is a model that takes as input a one-hot encoded word w and predicts the tokens in its context C_w. Therefore, the objective is to maximize the log-likelihood:

$$\sum_{w \in V} \sum_{c \in C_w} \log \mathbb{P}(c \mid w)$$

For both approaches, the weights of the projection layer E represents the embeddings of the vocabulary V.

GloVe

GloVe (Global Vectors) [51] is a model that uses the word-word co-occurrence matrix of the training corpus.

Let V be the vocabulary, N the size of the embeddings, X the co-occurrence matrix, $X_{i,j}$ an element of X indicating the number of times a word j occurred in the context window of a word i, $f(X_{i,j})$ a weighting function to handle the noise in X. GloVe minimizes the loss function:

$$\mathcal{L}(W, \tilde{W}, B, \tilde{B}) = \sum_{i=1}^{|V|} \sum_{j=1}^{|V|} f(X_{i,j}) \cdot (w_i^T \tilde{w}_j + b_i + \tilde{b}_j - \log X_{i,j})^2$$

where $W \in \mathbb{R}^{|V| \times N}$ and $B \in \mathbb{R}^{|V| \times N}$ are the embeddings and biases of the words in V, while $\tilde{W} \in \mathbb{R}^{|V| \times N}$ and $\tilde{B} \in \mathbb{R}^{|V| \times N}$ can be seen as the embeddings and biases of the context of the words in V. The final word embeddings are obtained as $W + \tilde{W}$, as it can be empirically shown that this approach improves the encoding.

fastText

fastText [9] is an extension of Word2vec to handle out of vocabulary words by exploiting their morphology. It creates a bag of character n-grams from the original vocabulary and learns their embeddings using the skip-gram model. With fastText, the embedding of a word is obtained as the sum of the vectors representing its subwords.

BioWordVec

BioWordVec [67] is a biomedical version of fastText trained on PubMed and the Medical Subject Headings (MeSH) RDF. The corpus from PubMed is composed of both titles and abstracts, while the corpus from the MeSH RDF is extracted by identifying paths on the graph and sampling its nodes to create a sequence of words.

4.1.3 Contextual embeddings

Contextual embeddings are models able to encode words in a dynamic manner depending on the context they appear in. This is different from traditional pretrained embeddings for which the context is only considered during training.

Therefore, contextual embeddings are more suited to handle polysemous words compared to traditional approaches.

In this work, we use the method introduced in SBERT [53] which embeds a document by encoding it using a pretrained language model (e.g. BERT) and averaging all its output vectors. SBERT shows that directly using the outputs of the original BERT produces worse results than GloVe. Therefore, a fine-tuning step is performed to specialize the model on the task of sentence similarity. During fine-tuning, a siamese network is used to compute, using the same weights, the embeddings of two documents which are then used to determine the loss based on their distance.

For our experiments, we will use MiniLM [62] (a distilled version of BERT), BioBERT [18] and PubMedBERT [17] fine-tuned for sentence similarity.

4.2 Dimensionality reduction

Dimensionality reduction methods are used to transform high-dimensional features into a low-dimensional space. This process is useful to refine sparse features or to remove noise after word embedding and prevent the curse of dimensionality.

4.2.1 Principal Component Analysis

Principal Component Analysis (PCA) [30] aims to project the dataset into a new lower-dimensional space while maximizing the variance of the data.

Given a p-dimensional mean-centered dataset $X = (X_1, X_2, \ldots, X_p) \in \mathbb{R}^{n \times p}$, principal components are a linear combination of the features that maximizes the variance. For instance, the first principal component Z_1 is defined as:

$$Z_1 = \phi_{1,1} X_1 + \phi_{2,1} X_2 + \cdots + \phi_{p,1} X_p$$

where $\phi_1 = (\phi_{1,1}, \phi_{2,1}, \ldots, \phi_{p,1}) \in \mathbb{R}^p$ contains the first principal component loadings and is constrained to $\|\phi_1\|_2 = 1$ in order to prevent arbitrarily high variance. The following i-th principal component Z_i has the additional constraint that it

must be uncorrelated (orthogonal) to the previous $Z_1, \ldots, Z_{i-1}$ principal components.

It can be proven [32] that this problem can be reduced to the eigendecomposition of the covariance matrix of the features. Let $C \in \mathbb{R}^{p \times p}$ be the covariance matrix of the features in X, $\lambda_1 \geq \cdots \geq \lambda_p$ its ordered eigenvalues and $\phi_1, \ldots, \phi_p$ the associated eigenvectors. The i-th principal component loadings are in the eigenvector ϕ_i associated to the i-th biggest eigenvalue λ_i.

Given all the loadings vectors $\Phi = (\phi_1, \ldots, \phi_p) \in \mathbb{R}^{p \times p}$, the full principal component decomposition can be defined as:

$$Z = X\Phi$$

To reduce the dimensionality of X, the first k principal components are kept and the others are dropped. The result will be a $n \times k$ matrix Z_k:

$$Z_k = X\Phi_k$$

4.2.2 Latent Semantic Analysis

Latent Semantic Analysis (LSA) [16, 1], also referred as Latent Semantic Indexing (LSI), is a dimensionality reduction technique specific for textual data.

Let $X \in \mathbb{R}^{n \times t}$ be the document-term matrix (e.g. generated using bag-of-words). It must be noted that $X^T X \in \mathbb{R}^{t \times t}$ corresponds to the column-wise dot products and computes the correlations among the terms. This matrix can be seen as a scaled approximation of the correlation matrix that, similarly to PCA, can be used for dimensionality reduction.

We can find the eigenvalues and eigenvectors of $X^T X$ by decomposing X using Singular Value Decomposition (SVD):

$$X = U\Sigma V^T$$

where $\Sigma \in \mathbb{R}^{n \times t}$ is a diagonal matrix containing the singular values, $U \in \mathbb{R}^{n \times n}$ and $V \in \mathbb{R}^{t \times t}$ are orthogonal matrices respectively with the left and right singular vectors as columns. A property of SVD is that the left singular vectors are the

eigenvectors of XX^T, the right singular vectors are the eigenvector of X^TX and the non-zero singular values are the square roots of the eigenvalues of both XX^T and X^TX.

With the decomposition of X, we can project X into a low-dimensional basis in the same way as in PCA, by selecting the first k eigenvectors of X^TX that correspond to the highest eigenvalues (or singular values):

$$Z_k = XV_k \approx (U_k\Sigma_k V_k^T)V_k = U_k\Sigma_k$$

4.2.3 Uniform Manifold Approximation and Projection

Uniform Manifold Approximation and Projection (UMAP) [42] is a nonlinear dimensionality reduction technique based on the assumption that the dataset is uniformly distributed across a locally connected Riemannian manifold and that each point has a locally constant Riemannian metric.

UMAP builds a fuzzy neighborhood topology that can be viewed as a graph where the weights of the edges are based on a locally defined metric by each point. Therefore, given two nodes a and b, the weights of the edges that connect them are not necessarily the same. To handle different metrics, the graph is uniformed by converting it into an undirected form where the weight connecting two nodes a and b is given by the probability disjunction of their weights $w(a, b)$ and $w(b, a)$ in the original graph.

With the final fuzzy topology of the original dataset, a low-dimensional representation can be obtained as an optimization problem by minimizing the cross-entropy between the fuzzy topologies of the original and low-dimensional data.

4.3 Clustering algorithms

4.3.1 Centroid-based

In centroid-based clustering [30, 1, 63], the dataset is partitioned into k clusters based on the distance between the data and the cluster centroid. The most common algorithm of this family is k-means.

Given a dataset with n entries $X = (x_1, \ldots, x_n)$ and the number of clusters to find k, k-means is solved as an optimization problem with objective:

$$\arg\min_{C} \sum_{i=1}^{k} \sum_{x \in C_i} \|x - \mu_i\|$$

where $C = \{C_1, \ldots, C_k\}$ are the k clusters and $\mu_i = \frac{1}{|C_i|} \sum_{x \in C_i} x$ is the centroid of the cluster C_i. This problem is proven to be NP-hard, therefore existing solutions are based on approximations and heuristics that usually converge to a local optimum. To cope with this limitation, the result of k-means is usually obtained after multiple runs with different initialization seeds. Many variations have also been proposed, for instance:

K-medoids enforces the centroid to be an element of the dataset.

K-medians selects the centroid as the median of the elements in the cluster.

Farthest first traversal k-center ensures that the centroids are representative of the dataset variance.

Bisecting k-means iteratively splits a cluster into two new clusters. Centroid are determined progressively on the cluster to split.

The complexity of an iteration of k-means is $O(kn)$, where k is the number of clusters and n the size of the dataset. An important disadvantage is that the number of clusters k is a user set parameter. Moreover, the algorithm is not noise robust and is not able to discover clusters with an arbitrary shape.

4.3.2 Hierarchical

Hierarchical clustering [30, 1] partitions a dataset based on the distance among the documents by building a dendrogram, a tree-like structure where each leaf represents a document and intermediate nodes can be seen as a different granularity of clustering. The tree can be built using two different strategies:

Agglomerative The tree is built starting from the leaves. Each document is part of its own cluster and is incrementally merged with the others until all documents are in the same cluster.

Divisive The tree is built from the root. All documents start in a single cluster that is subsequently decomposed until each cluster contains a single document.

Moreover, given a document distance function `dist` and two clusters C_1 and C_2, different linkage criterions can be used to determine the distance between C_1 and C_2:

Single linkage The distance is determined as the distance of the two closest documents in the two clusters.

$$\min_{a \in C_1,\, b \in C_2} \texttt{dist}(a, b)$$

This approach is not resistant to outliers which may cause uncorrelated clusters to merge (chaining problem).

Complete linkage The distance is determined as the distance of the two farthest documents in the two clusters.

$$\max_{a \in C_1,\, b \in C_2} \texttt{dist}(a, b)$$

Outliers may cause unbalanced clusters since they can be considered to compute the distance.

Average linkage The distance is determined as the mean between all the distances of all the documents in the two clusters.

$$\frac{1}{|C_1| \cdot |C_2|} \sum_{a \in C_1} \sum_{b \in C_2} \texttt{dist}(a, b)$$

Compared to single and complete linkage, this method is more robust to noise, but is computationally more expensive.

In general, the complexity of hierarchical clustering is $O(n^3)$ and $O(2^{n-1})$ respectively for the agglomerative and divisive methods. In some cases, more efficient algorithms of complexity $O(n^2)$ can be applied.

4.3.3 Density-based

Density-based clustering [3, 63] identifies clusters as high-density areas of the dataset. The most common methods are DBSCAN and its variants.

DBSCAN (Density-Based Spatial Clustering of Applications with Noise) [22] is based, as in hierarchical clustering, on the distance between documents and uses an additional density criterion based on the number of neighbors around a document.

Let D be a dataset, $N_\varepsilon(d)$ the neighbors of a document $d \in D$ in a radius ε (including d itself) and $m_{\text{neighbors}}$ the minimum number of neighbors a document has to have to be considered dense. The algorithm marks a document d as a core point if d has at least $m_{\text{neighbors}}$ in an ε radius around it:

$$\texttt{core\_points} = \{d \in D : |N_\varepsilon(d)| \geq m_{\text{neighbors}}\}$$

$$\texttt{non\_core\_points} = D \setminus \texttt{core\_points}$$

Core samples $d_i, d_j \in \texttt{core\_points}$ are part of the same cluster only if they are neighbors ($d_j \in N_\varepsilon(d_i)$). A non-core point $d_k \in \texttt{non\_core\_points}$ is part of the same cluster of its nearest core point neighbor, if exists; otherwise, it is considered an outlier.

OPTICS (Ordering Points To Identify the Clustering Structure) [4] uses the same approach of DBSCAN but takes into account the fact that clusters may have different densities and require different radiuses.

Let D, $N_\varepsilon(d)$ and $m_{\text{neighbors}}$ be defined as in DBSCAN. Let $\texttt{dist}(d_1, d_2)$ be the distance between documents d_1 and d_2. OPTICS defines for each document d its core distance:

$$\texttt{core\_dist}(d) = \begin{cases} \texttt{dist}(d, d_k), d_k \ m_{\text{neighbors}}\text{-th neighbor of } d & \text{if } |N_\varepsilon(d)| \geq m_{\text{neighbors}} \\ \text{undefined} & \text{otherwise} \end{cases}$$

Then, it is possible to define the reachability distance of a document d starting from d_o:

$$\texttt{reach\_dist}(d, d_o) = \begin{cases} \max\{\texttt{core\_dist}(d_o), \texttt{dist}(d_o, d)\} & \text{if } |N_\varepsilon(d_o)| \geq m_{\text{neighbors}} \\ \text{undefined} & \text{otherwise} \end{cases}$$

OPTICS processes the documents and produces an ordering of the documents based on their reachability distance. With the output of OPTICS, it is possible to represent a reachability plot with the ordered documents on the x-axis and the reachability distances on the y-axis. The plot has the form of multiple valleys where each of them represents a cluster. To extract the clusters, it is possible to set a threshold (which produces DBSCAN-like results) or by using algorithms to detect valleys. It must be noted that, differently from DBSCAN, ε for OPTICS represents the maximum radius.

HDBSCAN (Hierarchical DBSCAN) [11] can be seen as an improvement of OPTICS that removes the ε parameter completely.

Let D, $m_{\text{neighbors}}$ and $\texttt{dist}(d_1,\ d_2)$ be defined as above. Differently from OPTICS, the core distance is defined without considering a maximum radius:

$$\texttt{core\_dist}(d) = \texttt{dist}(d, d_k), d_k\ m_{\text{neighbors}}\text{-th neighbor of } d$$

Next, it is possible to define the mutual reachability distance of two documents d_1 and d_2:

$$\texttt{m\_reach\_dist}(d_1, d_2) = \max\{\texttt{core\_dist}(d_1), \texttt{core\_dist}(d_2), \texttt{dist}(d_1, d_2)\}$$

Then, a mutual reachability graph G can be built by using the documents as vertexes and the mutual reachability distances as edges. With G, it is possible to define a graph $G_\varepsilon \subseteq G$ where edges greater than ε are removed and documents with a core distance lower than ε are considered as noise. To cluster the dataset with radius ε, it is sufficient to find the connected components in G_ε.

HDBSCAN finds the clusters for all the possible values of ε. To do this, it finds a minimum spanning tree from the full mutual reachability graph G and removes at each iteration the edges with the highest mutual reachability

distance until all connected components have been removed. The result of this procedure is a hierarchy where each level corresponds to a different radius ε.

To convert the hierarchy into an actual partition of the dataset, HDBSCAN solves a constrained optimization problem based on the stability of the clusters. Let $C_1, \ldots, C_k$ be the ordered clusters in the hierarchy (C_1 is the root), $\mathcal{L}$ the indexes of leaf clusters, $\mathcal{P}_h$ the indexes of the clusters on the path from C_h to the root C_1 excluded and $\mathcal{S}(C_i)$ the stability of the cluster C_i, which is based on its lifetime in the hierarchy. The optimization problem is defined as:

$$\max_{\delta_1,\ldots,\delta_k} \sum_{i=2}^{k} \delta_i \mathcal{S}(C_i)$$

$$\text{subject to } \begin{cases} \delta_i \in \{0,1\}, i \in \{2,\ldots,k\} \\ \sum_{j \in \mathcal{P}_h} \delta_i = 1, \forall h \in \mathcal{L} \end{cases}$$

where $\delta_i = 1$ if cluster C_i is in the final solution. The second constraint prevents from selecting more than one cluster in a leaf-root path.

Density-based clustering is noise robust as it is able to detect outliers. Moreover, it is able to recognize clusters of arbitrary shapes, but works on the assumption that a cluster only exists in high density areas. The time complexity of DBSCAN, OPTICS and HDBSCAN is $O(n^2)$ in the worst case.

4.4 Topic extraction

As a final step of the clustering process, it could be useful to find for each cluster a set of keywords to represent its subtopic.

For this task, we follow the same approach proposed in BERTopic [24] which finds the topics of each cluster with a bag-of-words approach based on a cluster level TF-IDF named c-TF-IDF. Let $C = \{C_1, \ldots, C_n\}$ be the clusters, c_i the concatenation of the documents in the cluster C_i, f_{t,c_i} the number of occurrences

of the term t in c_i and A the average number of words per cluster. The term frequency of a term t in a cluster C_i can be defined using the standard TF:

$$\text{c-TF}(t, C_i) = \text{TF}(t, c_i)$$

The inverse class frequency is defined as:

$$\text{c-IDF}(t, C) = \log\left(1 + \frac{A}{\sum_{C_i \in C} f_{t,c_i}}\right)$$

Finally, the cluster level TF-IDF can be computes as:

$$\text{c-TF-IDF}(t, C_i, C) = \text{c-TF}(t, C_i) \cdot \text{c-IDF}(t, C)$$

4.5 Evaluation metrics

In this section, we discuss some of the available evaluation metrics for the clustering task [57]. As, to the best of our knowledge, human labeled datasets for biomedical abstract clustering do not exist, in this work we only present internal metrics that do not require to know the ground truth.

4.5.1 Calinski-Harabasz index

The Calinski-Harabasz index [10] is a value based on the dispersion of the elements within a cluster and between all clusters.

Let D be the dataset, $C_1, \ldots, C_n$ the n clusters of D and $\texttt{dist}(d_1, d_2)$ the distance between two documents $d_1 \in D$ and $d_2 \in D$. The within-cluster dispersion W is given by the sum of the squared distances of each document to the cluster centroid c_i it belongs to:

$$W = \sum_{i=1}^{n} \sum_{d \in C_i} \texttt{dist}(d, c_i)^2$$

The between-cluster dispersion B is given by the weighted sum of the squared distances of each cluster centroid c_i to the dataset centroid c_D:

$$B = \sum_{i=1}^{n} |C_i| \cdot \texttt{dist}(c_i, c_D)^2$$

The Calinski-Harabasz index is computed as:

$$\mathcal{S}_{\text{CH}} = \left(\frac{B}{n-1}\right) / \left(\frac{W}{|D|-n}\right)$$

The value of $\mathcal{S}_{\text{CH}}$ does not have an upper bound. Higher values of $\mathcal{S}_{\text{CH}}$ generally indicate better clusters.

4.5.2 Davies-Bouldin index

The Davies-Bouldin index [15] is a value based on the dispersion of the documents within a cluster with respect to its centroid and the dispersion of the centroids of all clusters.

Let D, $C_1, \ldots, C_n$ and $\texttt{dist}(d_1, d_2)$ be defined as above. The within-cluster dispersion s_i of a cluster C_i is given by the mean distance between each document and the centroid c_i of the cluster:

$$s_i = \frac{1}{|C_i|} \sum_{d \in C_i} \texttt{dist}(d, c_i)$$

The distance $m_{i,j}$ of two clusters C_i and C_j is given by the distance of their centroids c_i and c_j:

$$m_{i,j} = \texttt{dist}(c_i, c_j)$$

The score $r_{i,j}$ to assess the separation of two clusters C_i and C_j is given by:

$$r_{i,j} = \frac{s_i + s_j}{m_{i,j}}$$

The Davies-Bouldin index of the dataset is defined as:

$$\mathcal{S}_{\text{DB}} = \frac{1}{n} \sum_{i=1}^{n} \max_{j=1,\ldots,n,\ j \neq i} r_{i,j}$$

By definition $\mathcal{S}_{\text{DB}} \geq 0$, where a value of 0 indicates better defined clusters.

4.5.3 Silhouette coefficient

The silhouette coefficient [55] is a score based on the distance of each document with respect to the cluster it belongs to and to the nearest cluster.

Let D, $C_1, \ldots, C_n$ and $\texttt{dist}(d_1, d_2)$ be defined as above. For each document $d \in C_i$, the silhouette coefficient computes two scores:

$$a(d) = \frac{1}{|C_i| - 1} \sum_{d_k \in C_i,\ d_k \neq d} \texttt{dist}(d, d_k)$$

$$b(d) = \min_{C_j,\ C_j \neq C_i} \frac{1}{|C_j|} \sum_{d_k \in C_j} \texttt{dist}(d, d_k)$$

where $a(d)$ is the mean dissimilarity of d to all the other documents in its cluster and $b(d)$ is the mean dissimilarity of d to all the documents in the nearest cluster. The silhouette coefficient of a document d is then defined as:

$$s(d) = \frac{b(d) - a(i)}{\max\{a(d), b(d)\}}$$

The silhouette coefficient of the entire dataset is defined as the average silhouette coefficients of its documents:

$$\mathcal{S}_{\text{SC}} = \frac{1}{|D|} \sum_{d \in D} s(d)$$

The value of $\mathcal{S}_{\text{SC}}$ is bounded in $[-1, 1]$, where $\mathcal{S}_{\text{SC}} = 1$ indicates well defined clusters, while $\mathcal{S}_{\text{SC}} = -1$ indicates incorrect clusters.

4.6 Experiments

4.6.1 Dataset

As, to the best of our knowledge, a labeled dataset for biomedical documents clustering does not exist, to evaluate the clustering algorithms we built a dataset suitable for our use case.

Our dataset is composed of PubMed abstracts obtained through 20 different queries for a total of 66k articles. The abstracts for each query represent a group of documents to cluster, therefore the evaluation will assess twenty different executions. The final score for the dataset is given by the average of the evaluation scores of each run. It must be noted that this dataset is not labeled and we only rely on internal metrics.

For more details on the dataset creation, we refer the reader to Appendix A.

4.6.2 Implementation details

For our experiments, we used the implementation of dimensionality reduction methods, clustering algorithms and evaluation metrics available on `scikit-learn` [49]. For traditional pretrained embeddings we used the `Gensim` [52] library, while for contextual embeddings we used the `SentenceTransformers` [53] library.

4.6.3 Results

We evaluated different combinations of word embeddings, dimensionality reduction methods and clustering algorithms using different initialization hyperparameters. Tables 4.1, 4.2 and 4.3 contain the results of the best performing run using the metrics we previously discussed.

Between PCA and LSA, we only present PCA as both methods are implemented using SVD and yield similar results. Other results are omitted when the clustering algorithm labels all documents as noise or as a single class for at least one entry of the dataset.

It is important to note that it is not straightforward to compare clustering results obtained using different embeddings (for this reason, Tables 4.1, 4.2 and 4.3 should be read column-wise). Therefore, we are only able to make high-level observations.

Regarding dimensionality reduction, it can be seen that UMAP is the most consistent method that allows each clustering algorithm to produce a valid classification that is not a single class or all outliers.

Among the clustering algorithms, OPTICS is the one with the best performance on the Silhouette coefficient and Davies-Bouldin index. It is also observable that k-means has good results on the Calinski-Harabasz index. This is most likely a consequence of the definition of the Calinski-Harabasz index as it has the within-cluster dispersion, which k-means minimizes, in the denominator.

Given these results, in this work we will use UMAP as dimensionality reduction method and OPTICS as clustering algorithm. We will further experiment word embeddings to determine the most suitable approach.

		Bag-of-words			**Word2Vec**			**GloVe**		
		No reduction	PCA	UMAP	No reduction	PCA	UMAP	No reduction	PCA	UMAP
K-means	SC	0.04 ± 0.01	0.14 ± 0.02	0.37 ± 0.03	0.07 ± 0.01	0.11 ± 0.01	0.37 ± 0.04	0.07 ± 0.01	0.11 ± 0.01	0.35 ± 0.04
	CH	**32** ± 16	**177** ± 88	1386 ± 813	245 ± 130	**434** ± 238	**3303** ± 1830	**234** ± 121	**382** ± 199	**2846** ± 1603
	DB	4.26 ± 0.59	1.87 ± 0.10	1.05 ± 0.09	2.91 ± 0.33	2.24 ± 0.21	0.96 ± 0.14	2.87 ± 0.25	2.22 ± 0.18	1.04 ± 0.12
Agglomerative	SC			0.37 ± 0.04			0.30 ± 0.05	0.06 ± 0.01	0.08 ± 0.01	0.32 ± 0.04
	CH	–	–	1169 ± 713	–	–	2300 ± 1101	223 ± 116	339 ± 175	2706 ± 1553
	DB			0.97 ± 0.09			1.10 ± 0.11	3.15 ± 0.36	2.58 ± 0.19	1.11 ± 0.17
DBSCAN	SC			0.01 ± 0.32				0.03 ± 0.01	0.05 ± 0.02	0.28 ± 0.06
	CH	–	–	272 ± 227	–	–	–	48 ± 47	90 ± 81	1999 ± 995
	DB			1.15 ± 0.33				2.40 ± 0.75	2.16 ± 0.35	1.14 ± 0.12
OPTICS	SC		**0.51** ± 0.04	**0.68** ± 0.04			**0.72** ± 0.05			**0.73** ± 0.05
	CH	–	69 ± 21	**2443** ± 1222	–	–	2309 ± 1063	–	–	2364 ± 1024
	DB		**0.71** ± 0.08	**0.45** ± 0.06			**0.39** ± 0.06			**0.37** ± 0.05
HDBSCAN	SC	**0.16** ± 0.09	0.28 ± 0.09	0.52 ± 0.13		**0.37** ± 0.30	0.46 ± 0.23	**0.31** ± 0.26	**0.36** ± 0.29	0.43 ± 0.27
	CH	14 ± 5	93 ± 94	1400 ± 1210	–	125 ± 123	609 ± 438	74 ± 93	110 ± 133	443 ± 368
	DB	**1.71** ± 0.201	1.03 ± 0.14	0.58 ± 0.13		**0.94** ± 0.54	0.56 ± 0.28	**1.27** ± 0.69	**1.01** ± 0.57	0.64 ± 0.40

Table 4.1: Silhouette coefficient (SC), Calinski-Harabasz index (CH) and Davies-Bouldin index (DB) on clustering using bag-of-words, Word2Vec and GloVe. Column-wise **best results** are in bold and underlined.

		fastText			**BioWordVec**		
		No reduction	PCA	UMAP	No reduction	PCA	UMAP
K-means	SC	**0.07** ± 0.01	**0.11** ± 0.01	0.38 ± 0.04	0.09 ± 0.01	0.14 ± 0.02	0.38 ± 0.04
	CH	**352** ± 193	**533** ± 296	**3351** ± 1771	**323** ± 173	**504** ± 269	**3164** ± 1732
	DB	**2.61** ± 0.23	**2.13** ± 0.20	0.93 ± 0.12	2.60 ± 0.22	2.02 ± 0.20	0.99 ± 0.12
Agglomerative	SC	–	–	0.29 ± 0.05	0.05 ± 0.02	0.07 ± 0.03	0.34 ± 0.05
	CH			2583 ± 1593	75 ± 70	138 ± 117	2568 ± 1232
	DB			1.09 ± 0.11	2.35 ± 0.42	2.00 ± 0.21	1.00 ± 0.11
DBSCAN	SC	–	–	–	–	–	–
	CH						
	DB						
OPTICS	SC	–	–	**0.61** ± 0.05	–	–	**0.68** ± 0.06
	CH			2335 ± 1692			2922 ± 1403
	DB			**0.54** ± 0.06			**0.43** ± 0.07
HDBSCAN	SC	–	–	0.47 ± 0.20	**0.38** ± 0.29	**0.38** ± 0.32	0.48 ± 0.20
	CH			524 ± 403	101 ± 112	141 ± 151	710 ± 583
	DB			0.56 ± 0.25	**1.00** ± 0.55	**0.88** ± 0.49	0.52 ± 0.22

Table 4.2: Silhouette coefficient (SC), Calinski-Harabasz index (CH) and Davies-Bouldin index (DB) on clustering using fastText and BioWordVec. Column-wise **best results** are in bold and underlined.

		MiniLM			**BioBERT**			**PubMedBERT**		
		No reduction	PCA	UMAP	No reduction	PCA	UMAP	No reduction	PCA	UMAP
K-means	SC	0.04 ± 0.02	0.11 ± 0.01	0.38 ± 0.04	0.03 ± 0.01	0.08 ± 0.02	0.34 ± 0.05	0.04 ± 0.01	0.09 ± 0.02	0.36 ± 0.06
	CH	**117** ± 52	**145** ± 66	1907 ± 880	**97** ± 46	**214** ± 103	1874 ± 939	**101** ± 45	**236** ± 109	2078 ± 982
	DB	3.87 ± 0.35	2.16 ± 0.11	0.97 ± 0.08	4.13 ± 0.38	2.78 ± 0.20	1.09 ± 0.15	4.08 ± 0.42	2.69 ± 0.21	1.08 ± 0.18
Agglomerative	SC	0.05 ± 0.03	0.09 ± 0.02	0.40 ± 0.03	0.04 ± 0.01	0.05 ± 0.02	0.33 ± 0.06	0.02 ± 0.01	0.05 ± 0.02	0.36 ± 0.05
	CH	27 ± 23	85 ± 65	1975 ± 980	7 ± 1	24 ± 5	1523 ± 623	22 ± 4	69 ± 19	1739 ± 744
	DB	2.55 ± 1.30	2.16 ± 0.33	0.89 ± 0.06	2.30 ± 0.12	2.30 ± 0.11	1.05 ± 0.10	3.70 ± 0.23	2.65 ± 0.20	1.01 ± 0.10
DBSCAN	SC		**0.44** ± 0.04	-0.01 ± 0.32			-0.05 ± 0.25			-0.004 ± 0.278
	CH	–	40 ± 6	416 ± 638	–	–	217 ± 176	–	–	334 ± 484
	DB		**0.86** ± 0.08	0.99 ± 0.26			1.09 ± 0.32			1.02 ± 0.35
OPTICS	SC			**0.69** ± 0.03		**0.46** ± 0.07	**0.70** ± 0.04	**0.34** ± 0.04	**0.40** ± 0.04	**0.68** ± 0.05
	CH	–	–	**3685** ± 1651	–	30 ± 11	**2407** ± 1205	14 ± 3	31 ± 9	**2940** ± 1312
	DB			**0.43** ± 0.03		**0.82** ± 0.12	**0.42** ± 0.05	**1.08** ± 0.09	**0.93** ± 0.08	**0.46** ± 0.07
HDBSCAN	SC	**0.13** ± 0.05	0.13 ± 0.08	0.56 ± 0.04	**0.16** ± 0.06	0.19 ± 0.06	0.54 ± 0.13		0.13 ± 0.07	0.51 ± 0.05
	CH	13 ± 7	24 ± 19	1876 ± 990	16 ± 8	25 ± 15	841 ± 576	–	25 ± 20	1298 ± 701
	DB	**1.52** ± 0.25	1.31 ± 0.16	0.55 ± 0.05	**1.45** ± 0.36	1.21 ± 0.25	0.52 ± 0.20		1.38 ± 0.18	0.59 ± 0.07

Table 4.3: Silhouette coefficient (SC), Calinski-Harabasz index (CH) and Davies-Bouldin index (DB) on clustering using MiniLM, BioBERT and PubMedBERT. Column-wise **best results** are in bold and underlined.

5 Summarization

In this chapter, we present the task of automatic text summarization. We first give the definition of the task in Section 5.1, then describe evaluation metrics in Section 5.2 and datasets for biomedical summarization in Section 5.3. Finally, in Section 5.4, we discuss the experiments we conducted to choose the best summarization model.

5.1 Automatic text summarization

Summarization is the task of creating a shorter version of a document. Depending on how it is produced, summarization techniques can be classified [34] as:

Extractive The summary is created by selecting the most relevant sentences from the source document. The problem is a classification task, where a document D is defined by its sentences $D = \{s_i \mid i = 1 \dots n\}$. Each sentence s_i is classified with a positive label if s_i is part of the summary, a negative if not.

Abstractive The summary is produced as a paraphrase of the source document. The problem can be solved using a sequence-to-sequence model that takes as input the source document and outputs the summary.

Hybrid The summary is generated using both extractive and abstractive approaches. Usually, the source document is first processed using an extractive method, then the selected sentences are rewritten using an abstractive model.

Moreover, summarization can be classified based on the number of documents to process as:

Single-document The result is a straightforward summary of a single document.

Multi-document The summary involves more than one document. In this case, a strategy to process multiple documents is required (e.g. concatenation).

In this work, we will focus on extractive summarization and will use multi-document datasets.

5.2 Evaluation metrics

5.2.1 ROUGE

Recall-Oriented Understudy for Gisting Evaluation (ROUGE) [38] is a set of metrics based on the overlap between the generated summary and the ground truth. The most common ROUGE metrics are:

ROUGE-N computes the number of overlapping n-grams between the generated and reference summary. Let s be the reference summary, $\hat{s}$ the generated summary and $\texttt{grams}_n(d)$ the n-grams of a document d. The original formulation of ROUGE-N can be computed as:

$$\texttt{ROUGE-N}_{\text{recall}}(s, \hat{s}) = \frac{|\texttt{grams}_n(s) \cap \texttt{grams}_n(\hat{s})|}{|\texttt{grams}_n(s)|}$$

In existing implementations (e.g. `rouge-score`[1]), a ROUGE precision measure is also available:

$$\texttt{ROUGE-N}_{\text{precision}}(s, \hat{s}) = \frac{|\texttt{grams}_n(s) \cap \texttt{grams}_n(\hat{s})|}{|\texttt{grams}_n(\hat{s})|}$$

[1] `https://github.com/google-research/google-research/blob/master/rouge`

It must be noted that this measure is closely related to BLEU [48], a precision-oriented metric for machine translation. Differently from ROUGE, BLEU also has a brevity penalty for predictions shorter than the ground truth.

For our experiments, we will use ROUGE-1 and ROUGE-2 as they are commonly used in other work.

ROUGE-L is based on the longest common subsequence between the generated and reference summary. Let s be the reference summary, $\hat{s}$ the generate summary and $\texttt{LCS}(d_1, d_2)$ the longest common subsequence of two documents d_1 and d_2. ROUGE-L can be computed as:

$$\texttt{ROUGE-L}_{\text{recall}}(s, \hat{s}) = \frac{|\texttt{LCS}(s, \hat{s})|}{|s|}$$

$$\texttt{ROUGE-L}_{\text{precision}}(s, \hat{s}) = \frac{|\texttt{LCS}(s, \hat{s})|}{|\hat{s}|}$$

where $|\cdot|$ indicates the length of a textual sequence.

5.2.2 BERTScore

BERTScore [66] uses a pretrained language model (e.g. BERT) to create and compare the contextual embeddings of the generated summary and the ground truth. Differently from ROUGE, this approach is not based on an exact match and allows to better handle synonyms and reordering.

Let s be the reference summary, $\hat{s}$ the generated summary and $E(d)$ the contextual embeddings of a document d. We denote with $p_i \in E(d)$ the embedding of the i-th token of d. BERTScore is obtained by computing the cosine similarity between $E(s)$ and $E(\hat{s})$. The score associated to a token t of s (or $\hat{s}$) is determined by greedily selecting the maximum similarity between t and the other tokens in $\hat{s}$ (or s):

$$\texttt{BERTScore}_{\text{recall}} = \frac{1}{|s|} \sum_{p_i \in E(s)} \max_{q_j \in E(\hat{s})} p_i^T q_j$$

$$\texttt{BERTScore}_{\text{precision}} = \frac{1}{|\hat{s}|} \sum_{q_j \in E(\hat{s})} \max_{p_i \in E(s)} q_j^T p_i$$

where $|d|$ indicates the number of tokens in a document d.

5.3 Datasets

In this work, we use three biomedical datasets: MSˆ2, Cochrane and SumPubMed. All these datasets are built using scientific articles, mostly from PubMed. We also use CNN/Dailymail, a general-domain dataset on which we test the capabilities of our domain-specific fine-tuned models.

5.3.1 MSˆ2

MSˆ2 [20] is a multi-document summarization dataset with around 20k summaries of 470k documents cited in systematic literature reviews indexed by PubMed. The construction of the dataset utilizes a SciBERT classifier, trained on human annotated data, to identify in each review the research question (BACKGROUND) and the obtained results (TARGET). The documents to summarize are the abstracts of the studies cited in the reviews and the BACKGROUND sentences. The TARGET sentences form the reference summary.

5.3.2 Cochrane

Cochrane [60] is a multi-document summarization dataset created using 4528 systematic reviews from Cochrane[2]. Each review collects from PubMed trials on a specific clinical question. The documents to summarize are the abstracts of the trials cited in the review. The reference summary is the "conclusion" section of the review.

5.3.3 SumPubMed

SumPubMed [26] is a summarization dataset based on 33,772 full-text articles from BMC[3]. Each document is split into a front and a body: the front section contains the abstract and is considered as the reference summary; the body section

[2] https://www.cochranelibrary.com/

[3] https://www.biomedcentral.com

contains the main document, without non-textual elements (e.g. images, tables, ...), acknowledgments and references, and is considered as the source document.

5.3.4 CNN/Dailymail

CNN/Dailymail [29] is a summarization dataset built using 93k articles from CNN and 220k articles from Daily Mail. On the websites of both news providers, each article has some bullet points that summarize its content. The dataset uses these points as reference summary and the article as main document.

5.4 Experiments

5.4.1 Model architecture

The overall architecture for extractive summarization follows the same approach of BertSum [39] and is depicted in Figure 5.1. The document is first passed through a pretrained encoder, then a task specific layer is applied and the score for each sentence is outputted from a sigmoid.

Specifically, differently from the standard approach, each sentence fed to the pretrained encoder has its own classification token to learn sentence-level features. Moreover, for BERT-based models, segment embeddings are alternated between segment `A` and segment `B` to distinguish each individual sentence of the document. For Longformer-based models, the global attention mask is applied on all classification tokens to better capture inter-sentence features. The output of the encoder is filtered to only consider classification tokens, which are passed through some task specific Transformer encoders to learn inter-sentence features. The output score for each sentence is given by a final sigmoid layer.

5.4.2 Data preprocessing

To create an extractive summarization dataset, we use a method based on a greedy algorithm similar to BertSum. The algorithm iterates over the sentences

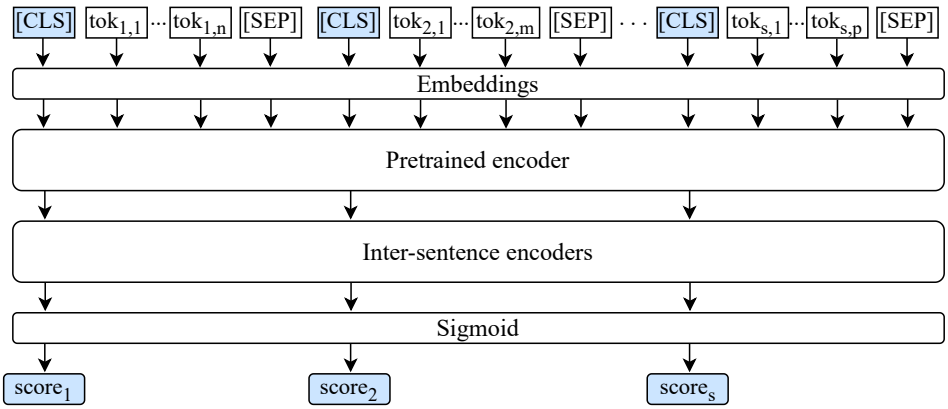

Figure 5.1: Model architecture for extractive summarization.

of the source document (as MSˆ2 and Cochrane are multi-document datasets, we concatenate the abstracts in each entry and treat them as a single document) and, at each iteration, selects the one that maximizes ROUGE-1 and ROUGE-2. The number of sentences to select is fixed to $\max(3, \text{sents}_{\text{ref}})$, where $\text{sents}_{\text{ref}}$ is the number of sentences in the reference summary. This value represents an upper bound since the greedy method may end before reaching the given number of selected sentences, when none of the remaining ones improve the score.

For documents longer than the input size of the model, we first try to reduce the number of tokens without truncation, by randomly removing sentences that are not part of the summary. This approach aims to preserve the semantic of the final summary. If after this process the document is still too long, we truncate it.

In Table 5.1, we report some statistics for each preprocessed dataset we will use for evaluation. For MSˆ2 and Cochrane, we use the validation set as the test set is not publicly available.

5.4.3 Training details

All the models are implemented using PyTorch and HuggingFace's `transformers` and `datasets` libraries to load models and datasets.

We trained our models on a single NVIDIA V100 (16 GB) GPU for 10 epochs using Binary Cross Entropy as loss and Adam as optimizer with $\beta_1 = 0.9$ and

Dataset	Documents	Avg. tokens	Avg. sents.	Avg. sel. sents.
MSˆ2 (val.)	2021	6597.05	291.80	3.43
Cochrane (val.)	470	2347.51	103.60	3.67
SumPubMed (test)	3269	4088.43	165.00	11.82
CNN/DailyMail (test)	11490	694.73	34.77	3.90

Table 5.1: Evaluation datasets statistics: number of documents, average number of tokens and sentences in the source document and average number of sentences selected by the greedy algorithm (Oracle).

$\beta_2 = 0.999$. At the end of training, we selected the checkpoint with the best results as final model.

Table 5.2 reports the configuration we used to train each model. As learning rate, we tested and selected the best value among $3e^{-6}$, $8e^{-6}$ and $1e^{-5}$. We fixed the batch size to 16 and used gradient accumulation when needed to cope with memory limitations. For efficiency purposes, we trained Longformer-based models using mixed precision (using both half-precision and single-precision floating-point).

Model family	Learning rate	Batch size	Gradient accum.	Training time
BERT	$3e^{-6}$	16	1	110 min
RoBERTa	$3e^{-6}$	16	1	110 min
DistilBERT	$1e^{-5}$	16	1	80 min
Longformer	$1e^{-5}$	1	16	20 h

Table 5.2: Training configuration per each model family

5.4.4 Evaluation method

We evaluate our models on the datasets we previously described. During evaluation, we make our models select the same number of sentences as the average number of sentences selected by the Oracle (see Table 5.1). As an additional baseline, we use Lead-n, which considers as the summary the first n sentences of

the document.

To summarize a document longer than the input size of the model, we create smaller chunks by partitioning the sentences. Each chunk is processed by the model individually and the output is created by concatenating each independent scores vector.

5.4.5 Results

Results on MSˆ2 and Cochrane are respectively reported in Table 5.3 and Table 5.4. On both datasets, Longformer performs the best on both ROUGE and BERTScore F1 scores.

Results on SumPubMed are reported in Table 5.5. In this case, PubMedBERT (abstracts + full-texts) is the model that yields the best results on both ROUGE and BERTScore F1 scores.

Results on CNN/Dailymail are reported in Table 5.6. Curiously, models pretrained on biomedical data archive better results compared to general domain models. In fact, the best performing model is SciBERT on both ROUGE and BERTScore. It must be noted that Lead-4 obtains better results than the pretrained models on this dataset, which is most likely caused by the fact that news articles are substantially different from scientific publications.

For each dataset, we report the standard deviation of the scores obtained by the pretrained models. It can be seen that almost all models have similar results with only small differences regardless of the pretraining corpus. More in details, biomedical models tend to have higher recall, which results in summaries closer to the ground truth. On the other hand, models pretrained on the general-domain have slightly higher precision, which corresponds to shorter generated summaries.

Moreover, it can be observed that the highest precision and F1 scores of ROUGE and BERTScore are reasonably correlated, while, considering only recall, a connection between ROUGE and BERTScore is not clearly notable.

Come to this conclusion, in the continuation of this work, we will use the base version of Longformer as it has the best precision-recall tradeoff and allows to process a significantly longer sequence compared to the other models.

Model	ROUGE-1 F1 (recall, prec.)	ROUGE-2 F1 (recall, prec.)	ROUGE-L F1 (recall, prec.)	BERTScore F1 (recall, prec.)
ORACLE	29.80 (49.82, 22.98)	11.31 (19.59, 8.64)	17.00 (29.31, 12.97)	85.43 (87.59, 83.41)
LEAD-3	17.43 (23.13, 16.85)	1.92 (2.64, 1.86)	11.19 (15.32, 10.70)	83.93 (85.04, 82.89)
$BERT_{BASE}$ (cased)	17.94 (32.62, 14.07)	2.45 (4.61, 1.91)	10.99 (20.93, 8.46)	83.17 (85.42, 81.08)
$BERT_{BASE}$ (uncased)	18.55 (35.08, 14.21)	2.67 (5.38, 2.02)	11.23 (22.30, 8.45)	83.48 (85.75, 81.38)
$RoBERTa_{BASE}$	18.95 (35.47, 14.58)	2.85 (5.66, 2.17)	11.42 (22.43, 8.62)	83.69 (85.85, 81.68)
DistilBERT	18.59 (33.65, 14.58)	2.68 (5.10, 2.09)	11.31 (21.48, 8.70)	83.56 (85.68, 81.59)
$Longformer_{BASE}$	**19.48** (33.49, **15.64**)	**2.88** (5.23, **2.29**)	**11.83** (21.28, **9.33**)	**84.12** (**85.91**, **82.45**)
$BioBERT_{BASE}$ (v1.2)	18.51 (35.71, 14.05)	2.82 (**5.68**, 2.12)	11.23 (22.77, 8.36)	83.37 (85.71, 81.21)
Clinical BioBERT	17.99 (34.86, 13.61)	2.64 (5.36, 1.97)	10.99 (22.35, 8.16)	83.19 (85.57, 81.00)
BlueBERT	18.84 (32.29, 15.21)	2.63 (4.73, 2.10)	11.49 (20.60, 9.10)	83.74 (85.71, 81.91)
PubMedBERT (abstracts only)	18.77 (35.35, 14.39)	2.82 (5.50, 2.15)	11.37 (22.49, 8.55)	83.58 (85.78, 81.54)
PubMedBERT (abs. + full-texts)	18.67 (35.62, 14.26)	2.83 (5.66, 2.14)	11.32 (22.71, 8.48)	83.51 (85.78, 81.42)
BioLinkBERT	18.65 (34.22, 14.54)	2.74 (5.25, 2.12)	11.33 (21.77, 8.67)	83.60 (85.74, 81.61)
SciBERT (cased)	18.87 (**35.88**, 14.36)	2.84 (5.67, 2.15)	11.42 (22.84, 8.53)	83.61 (85.84, 81.54)
SciBERT (uncased)	18.17 (35.86, 13.64)	2.74 (5.65, 2.04)	11.02 (**22.86**, 8.12)	83.17 (85.67, 80.86)
BioMed RoBERTa	18.91 (34.98, 14.65)	2.80 (5.44, 2.16)	11.40 (22.15, 8.67)	83.62 (85.81, 81.59)
DistilBioBERT	18.40 (31.72, 14.87)	2.52 (4.51, 2.03)	11.29 (20.37, 8.96)	83.55 (85.52, 81.71)
Clinical-Longformer	19.07 (33.52, 15.15)	2.73 (5.05, 2.15)	11.65 (21.49, 9.09)	83.93 (85.78, 82.20)
Standard deviation	0.40 (1.36, 0.55)	0.12 (0.39, 0.09)	0.22 (0.82, 0.34)	0.26 (0.13, 0.41)

Table 5.3: Evaluation on MSˆ2 validation set with 3 sentences selected. **Best results** are bolded and underlined.

Model	ROUGE-1 F1 (recall, prec.)	ROUGE-2 F1 (recall, prec.)	ROUGE-L F1 (recall, prec.)	BERTScore F1 (recall, prec.)
Oracle	30.68 (41.73, 25.66)	9.67 (13.42, 8.06)	17.15 (23.84, 14.24)	84.19 (86.08, 82.41)
Lead-4	20.74 (27.08, 19.09)	2.88 (3.78, 2.67)	12.48 (16.67, 11.38)	83.62 (85.48, 81.91)
$\text{BERT}_{\text{BASE}}$ (cased)	22.10 (35.36, 17.72)	3.74 (6.17, 2.97)	12.86 (21.41, 10.13)	83.59 (85.91, 81.46)
$\text{BERT}_{\text{BASE}}$ (uncased)	22.38 (36.63, 17.68)	3.90 (6.48, 3.08)	12.98 (22.07, 10.09)	83.73 (86.07, 81.58)
$\text{RoBERTa}_{\text{BASE}}$	23.08 (36.85, 18.46)	**4.17** (**6.86**, **3.30**)	13.38 (22.21, 10.52)	83.96 (**86.23**, 81.89)
DistilBERT	22.59 (35.75, 18.14)	3.91 (6.31, 3.11)	13.08 (21.50, 10.32)	83.82 (86.03, 81.79)
$\text{Longformer}_{\text{BASE}}$	**23.19** (36.01, **18.84**)	4.10 (6.66, 3.27)	**13.41** (21.69, **10.69**)	**84.02** (86.18, **82.03**)
$\text{BioBERT}_{\text{BASE}}$ (v1.2)	22.47 (**37.25**, 17.61)	4.02 (6.78, 3.13)	12.97 (22.38, 9.99)	83.70 (86.08, 81.53)
Clinical BioBERT	22.18 (36.65, 17.42)	3.83 (6.50, 2.99)	12.85 (22.03, 9.93)	83.73 (86.08, 81.56)
BlueBERT	22.52 (34.97, 18.42)	3.72 (5.91, 3.02)	13.06 (20.91, 10.53)	83.81 (86.00, 81.79)
PubMedBERT (abstracts only)	22.73 (37.06, 18.00)	4.04 (6.77, 3.16)	13.20 (22.35, 10.28)	83.91 (86.18, 81.82)
PubMedBERT (abs. + full-texts)	22.62 (37.17, 17.88)	3.99 (6.76, 3.12)	13.13 (**22.44**, 10.20)	83.77 (86.09, 81.64)
BioLinkBERT	22.67 (36.34, 18.15)	3.91 (6.39, 3.10)	13.14 (21.79, 10.35)	83.87 (86.11, 81.81)
SciBERT (cased)	22.63 (37.15, 17.85)	4.01 (6.70, 3.12)	13.02 (22.12, 10.11)	83.80 (86.14, 81.65)
SciBERT (uncased)	22.07 (36.99, 17.24)	3.92 (6.67, 3.05)	12.81 (22.31, 9.84)	83.53 (86.00, 81.27)
BioMed RoBERTa	22.36 (36.19, 17.82)	3.88 (6.51, 3.05)	12.84 (21.60, 10.06)	83.73 (86.04, 81.61)
DistilBioBERT	22.20 (34.64, 17.97)	3.70 (5.97, 2.96)	12.75 (20.62, 10.16)	83.74 (85.90, 81.75)
Clinical-Longformer	22.51 (36.49, 17.88)	3.96 (6.61, 3.12)	12.99 (21.93, 10.15)	83.80 (86.03, 81.74)
Standard deviation	0.32 (0.81, 0.40)	0.13 (0.29, 0.09)	0.19 (0.53, 0.23)	0.12 (0.09, 0.18)

Table 5.4: Evaluation on Cochrane validation set with 4 sentences selected. **Best results** are bolded and underlined.

Model	ROUGE-1 F1 (recall, prec.)	ROUGE-2 F1 (recall, prec.)	ROUGE-L F1 (recall, prec.)	BERTScore F1 (recall, prec.)
ORACLE	55.97 (70.35, 47.13)	30.09 (37.73, 25.38)	28.97 (36.29, 24.46)	88.34 (89.14, 87.56)
LEAD-12	40.17 (42.87, 39.28)	11.06 (11.92, 10.74)	18.21 (19.50, 17.77)	85.57 (85.75, 85.40)
$\text{BERT}_{\text{BASE}}$ (cased)	40.38 (57.01, 32.06)	13.29 (18.79, 10.54)	18.45 (26.25, 14.59)	86.05 (87.09, 85.06)
$\text{BERT}_{\text{BASE}}$ (uncased)	40.86 (60.32, 31.62)	14.22 (21.03, 10.99)	18.86 (28.05, 14.54)	86.46 (87.54, 85.42)
$\text{RoBERTa}_{\text{BASE}}$	40.81 (61.03, 31.38)	14.60 (21.92, 11.20)	19.00 (28.66, 14.56)	86.51 (87.63, 85.43)
DistilBERT	40.63 (59.16, 31.66)	13.92 (20.28, 10.84)	18.75 (27.52, 14.56)	86.31 (87.36, 85.29)
$\text{Longformer}_{\text{BASE}}$	**42.23** (58.32, **33.95**)	14.09 (19.56, 11.29)	19.05 (26.52, **15.26**)	86.43 (87.32, 85.55)
$\text{BioBERT}_{\text{BASE}}$ (v1.2)	39.89 (60.94, 30.34)	14.04 (21.49, 10.66)	18.58 (28.61, 14.08)	86.43 (87.59, 85.32)
Clinical BioBERT	39.42 (60.24, 29.94)	13.59 (20.80, 10.32)	18.21 (28.06, 13.78)	86.33 (87.47, 85.23)
BlueBERT	41.32 (57.90, 32.93)	13.87 (19.47, 11.04)	19.00 (26.82, 15.09)	86.24 (87.25, 85.27)
PubMedBERT (abstracts only)	40.67 (**61.42**, 31.09)	14.61 (22.10, 11.16)	18.95 (28.84, 14.44)	86.54 (87.64, 85.47)
PubMedBERT (abs. + full-texts)	41.49 (61.35, 32.11)	**15.08** (**22.36**, **11.65**)	**19.40** (**28.90**, 14.96)	**86.63** (**87.67**, **85.63**)
BioLinkBERT	41.34 (60.95, 32.03)	14.73 (21.77, 11.39)	19.15 (28.45, 14.78)	86.52 (87.60, 85.48)
SciBERT (cased)	40.50 (60.95, 31.03)	14.32 (21.59, 10.96)	18.82 (28.57, 14.37)	86.51 (87.61, 85.45)
SciBERT (uncased)	40.20 (60.79, 30.74)	14.18 (21.49, 10.83)	18.67 (28.46, 14.22)	86.46 (87.58, 85.39)
BioMed RoBERTa	40.91 (61.19, 31.46)	14.67 (22.00, 11.26)	19.09 (28.78, 14.63)	86.49 (87.66, 85.36)
DistilBioBERT	40.92 (57.65, 32.48)	13.67 (19.27, 10.85)	18.76 (26.63, 14.84)	86.14 (87.19, 85.13)
Clinical-Longformer	42.05 (59.08, 33.49)	14.34 (20.24, 11.39)	19.18 (27.16, 15.21)	86.47 (87.40, 85.58)
Standard deviation	0.73 (1.48, 1.08)	0.47 (1.14, 0.35)	0.30 (0.93, 0.40)	0.16 (0.18, 0.16)

Table 5.5: Evaluation on SUMPUBMED test set with 12 sentences selected. **Best results** are bolded and underlined.

Model	ROUGE-1 F1 (recall, prec.)	ROUGE-2 F1 (recall, prec.)	ROUGE-L F1 (recall, prec.)	BERTScore F1 (recall, prec.)
Oracle	49.93 (73.26, 39.01)	29.84 (43.34, 23.50)	34.35 (50.13, 26.95)	88.71 (90.36, 87.14)
Lead-4	37.77 (55.76, 30.03)	17.04 (25.35, 13.50)	24.12 (35.94, 19.09)	86.39 (87.56, 85.28)
$\text{BERT}_{\text{BASE}}$ (cased)	28.57 (46.38, 21.74)	9.85 (16.24, 7.44)	17.81 (29.28, 13.48)	84.85 (86.18, 83.59)
$\text{BERT}_{\text{BASE}}$ (uncased)	28.09 (45.33, 21.48)	9.29 (15.22, 7.04)	17.32 (28.31, 13.16)	84.72 (85.96, 83.56)
$\text{RoBERTa}_{\text{BASE}}$	27.98 (44.83, 21.46)	9.15 (14.97, 6.94)	17.20 (27.93, 13.10)	84.68 (85.85, 83.57)
DistilBERT	27.35 (45.35, 20.57)	8.92 (14.92, 6.69)	16.87 (28.32, 12.62)	84.59 (85.91, 83.33)
$\text{Longformer}_{\text{BASE}}$	29.00 (47.59, 21.84)	9.70 (16.06, 7.29)	17.71 (29.42, 13.27)	84.81 (86.13, 83.55)
$\text{BioBERT}_{\text{BASE}}$ (v1.2)	29.23 (**48.94**, 21.82)	10.40 (17.54, 7.75)	18.24 (30.90, 13.55)	84.92 (86.37, 83.55)
Clinical BioBERT	28.39 (48.67, 20.97)	9.88 (16.98, 7.30)	17.65 (30.61, 12.98)	84.73 (86.24, 83.29)
BlueBERT	27.44 (44.88, 20.73)	8.78 (14.43, 6.62)	16.75 (27.71, 12.59)	84.58 (85.86, 83.35)
PubMedBERT (abstracts only)	27.44 (48.23, 20.01)	9.27 (16.25, 6.77)	16.95 (30.11, 12.31)	84.48 (86.04, 83.00)
PubMedBERT (abs. + full-texts)	28.65 (48.37, 21.32)	9.95 (16.89, 7.39)	17.79 (30.39, 13.18)	84.81 (86.25, 83.44)
BioLinkBERT	27.40 (46.74, 20.30)	9.04 (15.47, 6.70)	16.77 (28.95, 12.37)	84.48 (85.91, 83.12)
SciBERT (cased)	**29.83** (50.36, **22.14**)	**10.84** (**18.38**, **8.04**)	**18.65** (**31.85**, **13.79**)	**85.06** (**86.58**, **83.61**)
SciBERT (uncased)	29.44 (49.77, 21.87)	10.53 (17.86, 7.82)	18.38 (31.42, 13.59)	84.96 (86.47, 83.54)
BioMed RoBERTa	28.47 (47.33, 21.35)	9.74 (16.36, 7.28)	17.55 (29.54, 13.10)	84.79 (86.16, 83.48)
DistilBioBERT	27.24 (44.46, 20.64)	8.68 (14.31, 6.55)	16.74 (27.65, 12.61)	84.55 (85.83, 83.33)
Clinical-Longformer	29.02 (49.82, 21.37)	10.17 (17.54, 7.50)	17.90 (31.10, 13.13)	84.82 (86.36, 83.37)
Standard deviation	0.82 (1.96, 0.62)	0.65 (1.25, 0.46)	0.61 (1.37, 0.44)	0.17 (0.24, 0.18)

Table 5.6: Evaluation on CNN/Dailymail test set with 4 sentences selected. **Best results** are bolded and underlined.

6 Proposed method

6.1 Framework

The method we propose is mainly based on the idea of summarizing subtopic related articles to minimize the possibility of mixing or losing information.

As presented in Figure 6.1, given a group of abstracts A to summarize, the framework applies the following steps:

1. Cluster A to obtain its clusters $A_1, \ldots, A_n$. Outliers are also considered as their own cluster.

2. Summarize each A_i to obtain the corresponding summary S_i.

3. Summarize the concatenation of each S_i to obtain the overall summary S.

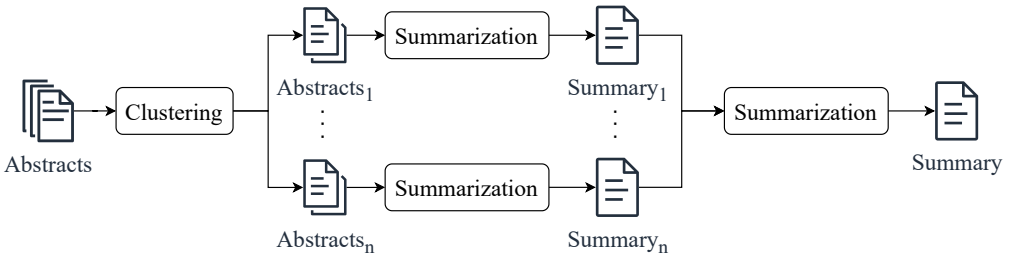

Figure 6.1: Framework steps

Given the results of the framework, S can be considered as the final summary. Alternatively, $S_1, \ldots, S_n$, along side the subtopics obtained using c-TF-IDF (Chapter 4), can be presented as a subtopic-oriented summary.

6.2 Experiments

6.2.1 Dataset

To evaluate our method, we use the validation set of MSˆ2 and Cochrane as we did in Chapter 5. Since the framework is designed for multi-document summarization, we exclude the entries with less than 15 articles. Statistics of the resulting datasets are presented in Table 6.1.

Dataset	**Entries**	**Avg. abstracts per entry**	**Avg. sel. sents.**
MSˆ2 (val.)	1262	33.23	3.50
Cochrane (val.)	105	29.70	4.16

Table 6.1: Evaluation datasets statistics: number of entries, average number of abstracts per entry and average number of sentences selected by the Oracle.

6.2.2 Evaluation method

As mentioned in Chapter 4, for the clustering phase, we use UMAP and OPTICS respectively as dimensionality reduction method and clustering algorithm. UMAP reduces the features into a 10-dimensional space. Parameters for OPTICS are $\varepsilon = +\infty$ and $m_{\text{neighbors}} = 2$. Since previously we did not individuate the best embedding, we experiment and report the results for all the embeddings we presented.

For the summarization phase, as concluded in Chapter 5, we use the base version of Longformer. Similarly to the previous experiments, we make our model select the same number of sentences as the average number of sentences selected by the Oracle.

6.2.3 Results

Framework evaluation

Results on MSˆ2 are presented in Table 6.2. Considering only F1 scores, summaries created only using Longformer yield better results, while our method obtains slightly lower scores. A notable property, which we will discuss further on, is that the framework obtains higher recall but lower precision.

Results on Cochrane are presented in Table 6.3. This time, our method has slightly increased F1 scores compared to only using Longformer. Again, the framework increases recall but decreases precision.

Across the different text embeddings, with the current results, we think that MiniLM is the most suitable one as it has one of the highest ROUGE recall and BERTScore. Still, it is not clear which of them performs better and further study may be required.

Clustering criteria analysis

As a further experiment, we use an alternative formulation of our framework that clusters using PubMed keywords and MeSH terms instead of full-text abstracts.

Results using this configuration are reported in Table 6.4. For computational limitations, we are only able to evaluate on Cochrane. As in the previous cases, there are only minor differences in F1 scores. Recall and precision also have the same behavior we noted before.

Compared to using full-text abstracts, clustering using keywords resulted in a small decrease in performance. This is most likely given by the fact that embeddings that use abstracts are able to capture more information compared to only using keywords.

Model	ROUGE-1 F1 (recall, prec.)	ROUGE-2 F1 (recall, prec.)	ROUGE-L F1 (recall, prec.)	BERTScore F1 (recall, prec.)
ORACLE	30.65 (51.06, 23.58)	12.23 (21.05, 9.33)	17.33 (29.73, 13.20)	85.50 (87.72, 83.43)
Longformer only	**18.72** (38.11, **13.85**)	3.00 (6.43, **2.20**)	**11.10** (23.90, **8.07**)	**83.45** (85.86, **81.21**)
Framework embedding				
Bag-of-words	18.43 (40.10, 13.20)	2.97 (6.91, 2.10)	10.86 (25.02, 7.63)	83.14 (85.89, 80.60)
Word2Vec	18.45 (40.04, 13.22)	2.98 (6.87, 2.12)	10.87 (25.04, 7.65)	83.17 (85.90, 80.65)
GloVe	18.29 (39.93, 13.10)	3.02 (**7.00**, 2.13)	10.85 (25.00, 7.64)	83.06 (85.84, 80.50)
fastText	18.33 (39.71, 13.16)	2.99 (6.89, 2.11)	10.89 (24.97, 7.67)	83.11 (85.85, 80.60)
BioWordVec	18.48 (40.04, 13.25)	**3.05** (6.94, 2.17)	10.91 (25.03, 7.68)	83.13 (85.87, 80.62)
MiniLM	18.59 (**40.20**, 13.34)	3.01 (6.92, 2.14)	11.05 (**25.24**, 7.79)	83.20 (**85.91**, 80.72)
BioBERT	18.49 (40.17, 13.23)	3.04 (6.98, 2.15)	10.94 (25.13, 7.68)	83.15 (85.88, 80.63)
PubMedBERT	18.50 (39.95, 13.28)	3.03 (6.96, 2.15)	10.98 (25.14, 7.74)	83.17 (85.90, 80.65)
Average framework improvement	-0.27 (+1.91, -0.63)	+0.01 (+0.50, -0.07)	-0.18 (+1.17, -0.39)	-0.31 (+0.02, -0.59)

Table 6.2: Evaluation with abstracts clustering on MSˆ2 validation set with 4 sentences summaries. **Best results** are bolded and underlined.

Model	ROUGE-1 F1 (recall, prec.)	ROUGE-2 F1 (recall, prec.)	ROUGE-L F1 (recall, prec.)	BERTScore F1 (recall, prec.)
ORACLE	32.14 (44.81, 26.52)	11.46 (16.64, 9.38)	17.36 (25.02, 14.16)	85.20 (87.10, 83.41)
Longformer only	22.99 (36.02, **19.12**)	3.33 (5.66, 2.72)	**12.93** (21.26, **10.49**)	83.94 (86.04, 81.98)
Framework embedding				
Bag-of-words	22.97 (38.05, 18.46)	3.49 (6.08, 2.74)	12.70 (22.15, 9.95)	83.93 (86.06, 81.95)
Word2Vec	**23.39** (**38.59**, 18.73)	**3.57** (6.10, **2.80**)	12.86 (22.19, 10.08)	84.14 (86.14, 82.27)
GloVe	22.46 (37.44, 17.95)	3.41 (6.03, 2.69)	12.56 (21.99, 9.82)	83.87 (86.05, 81.84)
fastText	23.03 (38.38, 18.47)	3.54 (**6.36**, 2.75)	12.58 (22.15, 9.84)	83.97 (86.00, 82.07)
BioWordVec	22.96 (38.50, 18.27)	3.49 (6.13, 2.73)	12.73 (**22.26**, 9.89)	84.02 (86.16, 82.02)
MiniLM	23.21 (38.17, 18.83)	3.41 (5.87, 2.71)	12.62 (21.72, 9.99)	**84.17** (**86.17**, **82.31**)
BioBERT	23.11 (38.31, 18.43)	3.55 (6.24, **2.80**)	12.60 (21.90, 9.83)	84.00 (86.10, 82.03)
PubMedBERT	22.67 (37.63, 18.22)	3.48 (5.98, 2.78)	12.81 (22.05, 10.09)	84.04 (86.12, 82.11)
Average framework improvement	-0.01 (+2.11, -0.70)	+0.16 (+0.43, +0.03)	-0.25 (+0.79, -0.55)	+0.08 (+0.06, +0.09)

Table 6.3: Evaluation with abstracts clustering on Cochrane validation set with 4 sentences summaries. **Best results** are bolded and underlined.

Model	ROUGE-1 F1 (recall, prec.)	ROUGE-2 F1 (recall, prec.)	ROUGE-L F1 (recall, prec.)	BERTScore F1 (recall, prec.)
Longformer only	22.99 (36.02, **19.12**)	3.33 (5.66, 2.72)	**12.93** (21.26, **10.49**)	83.94 (86.04, 81.98)
Framework embedding				
Bag-of-words	22.86 (37.85, 18.41)	3.48 (6.04, 2.77)	12.54 (21.61, 9.89)	**84.07** (**86.15**, **82.13**)
Word2Vec	**23.00** (38.02, 18.49)	3.44 (5.96, 2.72)	12.67 (22.05, 9.95)	**84.07** (**86.15**, **82.13**)
GloVe	22.95 (38.17, 18.22)	3.44 (5.85, 2.71)	12.78 (22.10, 9.98)	84.04 (86.07, 82.13)
fastText	22.74 (37.82, 18.27)	3.44 (5.86, 2.75)	12.61 (22.05, 9.85)	83.98 (86.06, 82.04)
BioWordVec	22.77 (37.83, 18.39)	3.45 (5.86, 2.78)	12.64 (21.88, 10.00)	83.93 (86.01, 81.99)
MiniLM	22.83 (38.05, 18.25)	3.30 (5.81, 2.59)	12.53 (21.93, 9.78)	84.00 (86.14, 82.01)
BioBERT	22.73 (37.68, 18.22)	**3.58** (**6.19**, **2.86**)	12.68 (22.19, 9.93)	84.00 (86.07, 82.07)
PubMedBERT	22.97 (**38.20**, 18.46)	3.47 (5.99, 2.75)	12.73 (**22.25**, 9.99)	83.96 (86.11, 81.95)
Average framework improvement	-0.13 (+1.93, -0.78)	+0.12 (+0.28, +0.02)	-0.28 (+0.75, -0.57)	+0.07 (+0.05, +0.08)

Table 6.4: Evaluation with keywords clustering on Cochrane validation set with 4 sentences summaries. **Best results** are bolded and underlined.

Precision-recall analysis

The difference in recall and precision is reasonably given by the fact that the generated summaries are longer but overlap better with the ground truth. To further study this behavior, we report in Table 6.5 the average number of words in the generated summaries on Cochrane.

Model/embedding	**Avg. words in summaries**
Longformer only	140.19
Bag-of-words	153.38
Word2Vec	153.02
GloVe	154.93
fastText	154.01
BioWordVec	153.85
MiniLM	152.16
BioBERT	153.90
PubMedBERT	153.07

Table 6.5: Average number of words in Cochrane generated summaries

Compared to the results of only using Longformer, our method selects slightly longer sentences for the summary, which explains the decrease in precision. Considered that the gain in recall is generally larger in comparison to the loss in precision, we believe that the method we propose is a valid approach to improve the quality of a summary.

7 Conclusion

In this thesis, we presented a subtopic-oriented framework to summarize abstracts in the biomedical field. We first evaluated clustering algorithms paired with different text embeddings and dimensionality reduction methods and concluded that, for our use case, the best dimensionality reduction method is UMAP and the best clustering algorithm is OPTICS. Then, we experimented different general-domain and biomedical pretrained language models fine-tuned for the task of extractive summarization and found out that models pretrained on a biomedical corpus generally obtain higher recall, while general-domain models result in higher precision. Among all, we selected Longformer, which has the best precision-recall tradeoff, as our summarization model. Finally, we assessed the performance of our framework. We observed that, compared to directly using a summarization model, our method increases the recall and slightly decreases the precision of the generated summary, which results in more accurate but longer summaries. We studied this behavior by analyzing the size of the generated summaries and concluded that the loss in precision is acceptable as the overhead in size is negligible.

With these results, we believe that the method we discussed is a valid approach to improve the quality of a summary and to make the result more interpretable as it is enriched with information on the subtopics.

Future expansions of this work could consider to: (i) explore the use of fuzzy clustering algorithms as an alternative to the hard clustering algorithms we employed, (ii) use additional clustering criteria such as time information, (iii) experiment abstractive or hybrid summarization.

Bibliography

[1] Aggarwal Charu and Zhai Chengxiang. "A Survey of Text Clustering Algorithms". In: *Mining Text Data* (Aug. 2012). DOI: 10.1007/978-1-4614-3223-4_4.

[2] Alsentzer Emily et al. "Publicly Available Clinical BERT Embeddings". In: *Proceedings of the 2nd Clinical Natural Language Processing Workshop*. Association for Computational Linguistics, June 2019, pp. 72–78. DOI: 10.18653/v1/W19-1909.

[3] Andreopoulos Bill et al. "A roadmap of clustering algorithms: finding a match for a biomedical application". In: *Briefings in Bioinformatics* 10.3 (Feb. 2009), pp. 297–314. DOI: 10.1093/bib/bbn058.

[4] Ankerst Mihael et al. "OPTICS: Ordering Points to Identify the Clustering Structure". In: *Proceedings of the 1999 ACM SIGMOD International Conference on Management of Data*. Association for Computing Machinery, June 1999, pp. 49–60. DOI: 10.1145/304182.304187.

[5] Aronson Alan R and Lang François-Michel. "An overview of MetaMap: historical perspective and recent advances". In: *Journal of the American Medical Informatics Association* 17.3 (May 2010), pp. 229–236. ISSN: 1067-5027. DOI: 10.1136/jamia.2009.002733.

[6] Beltagy Iz, Lo Kyle, and Cohan Arman. "SciBERT: A Pretrained Language Model for Scientific Text". In: *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*. As-

sociation for Computational Linguistics, Nov. 2019, pp. 3615–3620. DOI: 10.18653/v1/D19-1371.

[7] Beltagy Iz, Peters Matthew E., and Cohan Arman. *Longformer: The Long-Document Transformer*. Dec. 2020. arXiv: 2004.05150 [cs.CL].

[8] Bodenreider Olivier. "The Unified Medical Language System (UMLS): integrating biomedical terminology". In: *Nucleic Acids Res.* 32.Database issue (Jan. 2004), pp. D267–70. DOI: 10.1093/nar/gkh061.

[9] Bojanowski Piotr et al. *Enriching Word Vectors with Subword Information*. June 2017. arXiv: 1607.04606 [cs.CL].

[10] Caliński Tadeusz and JA Harabasz. "A Dendrite Method for Cluster Analysis". In: *Communications in Statistics - Theory and Methods* 3 (Jan. 1974), pp. 1–27. DOI: 10.1080/03610927408827101.

[11] Campello Ricardo J. G. B., Moulavi Davoud, and Sander Joerg. "Density-Based Clustering Based on Hierarchical Density Estimates". In: *Advances in Knowledge Discovery and Data Mining*. Springer Berlin Heidelberg, 2013, pp. 160–172. DOI: 10.1007/978-3-642-37456-2_14.

[12] Chaves Andrea, Kesiku Cyrille, and Garcia-Zapirain Begonya. "Automatic Text Summarization of Biomedical Text Data: A Systematic Review". In: *Information* 13.8 (Aug. 2022). ISSN: 2078-2489. DOI: 10.3390/info13080393.

[13] Chiu Billy and Baker Simon. "Word embeddings for biomedical natural language processing: A survey". In: *Language and Linguistics Compass* 14.12 (Dec. 2020), e12402. DOI: https://doi.org/10.1111/lnc3.12402.

[14] Dai Lin, Tang Ji-Liang, and Xia Yun-Qing. "Subtopic-based multi-document summarization". In: *2009 International Conference on Machine Learning and Cybernetics*. Vol. 6. July 2009, pp. 3505–3510. DOI: 10.1109/ICMLC.2009.5212767.

[15] Davies David L. and Bouldin Donald W. "A Cluster Separation Measure". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* PAMI-1.2 (Apr. 1979), pp. 224–227. DOI: 10.1109/TPAMI.1979.4766909.

[16] Deerwester Scott et al. “Indexing by latent semantic analysis”. In: *Journal of the American Society for Information Science* 41.6 (Sept. 1990), pp. 391–407. DOI: https://doi.org/10.1002/(SICI)1097-4571(199009)41:6<391::AID-ASI1>3.0.CO;2-9.

[17] Deka Pritam, Jurek-Loughrey Anna, and P Deepak. “Evidence Extraction to Validate Medical Claims in Fake News Detection”. In: *Health Information Science*. Springer Nature Switzerland, 2022, pp. 3–15. DOI: 10.1007/978-3-031-20627-6_1.

[18] Deka Pritam, Jurek-Loughrey Anna, and Padmanabhan Deepak. “Improved methods to aid unsupervised evidence-based fact checking for online health news”. In: *Journal of Data Intelligence* 3.4 (Nov. 2022), pp. 474–505. DOI: 10.26421/JDI3.4-5.

[19] Devlin Jacob et al. *BERT: pretraining of Deep Bidirectional Transformers for Language Understanding*. May 2019. arXiv: 1810.04805 [cs.CL].

[20] DeYoung Jay et al. “MSˆ2: Multi-Document Summarization of Medical Studies”. In: *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, Nov. 2021, pp. 7494–7513. DOI: 10.18653/v1/2021.emnlp-main.594.

[21] Dong Luobing et al. “Two-Phase Multidocument Summarization Through Content-Attention-Based Subtopic Detection”. In: *IEEE Transactions on Computational Social Systems* 8.6 (July 2021), pp. 1379–1392. DOI: 10.1109/TCSS.2021.3079206.

[22] Ester Martin et al. “A Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise”. In: *Proceedings of the Second International Conference on Knowledge Discovery and Data Mining*. AAAI Press, Aug. 1996, pp. 226–231. DOI: 10.5555/3001460.3001507.

[23] Gong Shu, Qu Youli, and Tian Shengfeng. “Subtopic-based Multi-documents Summarization”. In: *2010 Third International Joint Conference on Computational Science and Optimization*. Vol. 2. May 2010, pp. 382–386. DOI: 10.1109/CSO.2010.239.

[24] Grootendorst Maarten. *BERTopic: Neural topic modeling with a class-based TF-IDF procedure*. Mar. 2022. arXiv: 2203.05794 [cs.CL].

[25] Gu Yu et al. "Domain-Specific Language Model Pretraining for Biomedical Natural Language Processing". In: *ACM Transactions on Computing for Healthcare* 3.1 (Oct. 2021), pp. 1–23. DOI: 10.1145/3458754.

[26] Gupta Vivek et al. "SumPubMed: Summarization Dataset of PubMed Scientific Articles". In: Association for Computational Linguistics, Aug. 2021, pp. 292–303. DOI: 10.18653/v1/2021.acl-srw.30.

[27] Gururangan Suchin et al. *Don't Stop Pretraining: Adapt Language Models to Domains and Tasks*. May 2020. arXiv: 2004.10964 [cs.CL].

[28] Harris Zellig S. "Distributional Structure". In: *Word* 10.2-3 (1954), pp. 146–162. DOI: 10.1080/00437956.1954.11659520.

[29] Hermann Karl Moritz et al. *Teaching Machines to Read and Comprehend*. Nov. 2015. arXiv: 1506.03340 [cs.CL].

[30] James Gareth et al. *An introduction to statistical learning*. Springer International Publishing, 2023. DOI: 10.1007/978-3-031-38747-0. URL: https://www.statlearning.com.

[31] Johnson Alistair E.W. et al. "MIMIC-III, a freely accessible critical care database". In: *Scientific Data* 3.1 (May 2016). DOI: 10.1038/sdata.2016.35.

[32] Jolliffe Ian T. and Cadima Jorge. "Principal component analysis: A review and recent developments". In: *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences* 374.2065 (Apr. 2016). DOI: 10.1098/rsta.2015.0202.

[33] Jurafsky Dan; Martin James H. *Speech and Language Processing*. Jan. 2023. URL: https://web.stanford.edu/~jurafsky/slp3/.

[34] El-Kassas Wafaa S. et al. "Automatic text summarization: A comprehensive survey". In: *Expert Systems with Applications* 165 (Mar. 2021), p. 113679. ISSN: 0957-4174. DOI: https://doi.org/10.1016/j.eswa.2020.113679.

[35] Koh Huan Yee et al. “An Empirical Survey on Long Document Summarization: Datasets, Models, and Metrics”. In: *ACM Computing Surveys* 55.8 (Dec. 2022), pp. 1–35. DOI: 10.1145/3545176.

[36] Lee Jinhyuk et al. “BioBERT: a pretrained biomedical language representation model for biomedical text mining”. In: *Bioinformatics* 36.4 (Sept. 2019), pp. 1234–1240. ISSN: 1367-4803. DOI: 10.1093/bioinformatics/btz682.

[37] Li Yikuan et al. *Clinical-Longformer and Clinical-BigBird: Transformers for long clinical sequences*. Apr. 2022. arXiv: 2201.11838 [cs.CL].

[38] Lin Chin-Yew. “ROUGE: A Package for Automatic Evaluation of Summaries”. In: *Text Summarization Branches Out*. Barcelona, Spain: Association for Computational Linguistics, July 2004, pp. 74–81. URL: https://aclanthology.org/W04-1013.

[39] Liu Yang and Lapata Mirella. *Text Summarization with Pretrained Encoders*. Aug. 2019. arXiv: 1908.08345 [cs.CL].

[40] Liu Yinhan et al. *RoBERTa: A Robustly Optimized BERT Pretraining Approach*. July 2019. arXiv: 1907.11692 [cs.CL].

[41] Ma Congbo et al. “Multi-Document Summarization via Deep Learning Techniques: A Survey”. In: *ACM Comput. Surv.* 55.5 (Dec. 2022). DOI: 10.1145/3529754.

[42] McInnes Leland, Healy John, and Melville James. *UMAP: Uniform Manifold Approximation and Projection for Dimension Reduction*. Sept. 2020. arXiv: 1802.03426 [stat.ML].

[43] Mei Jian-Ping and Chen Lihui. “SumCR: A new subtopic-based extractive approach for text summarization”. In: *Knowledge and Information Systems - KAIS* 31 (June 2012), pp. 527–545. DOI: 10.1007/s10115-011-0437-x.

[44] Mikolov Tomas et al. *Efficient Estimation of Word Representations in Vector Space*. Sept. 2013. arXiv: 1301.3781 [cs.CL].

[45] Mishra Rashmi et al. “Text summarization in the biomedical domain: A systematic review of recent research”. In: *Journal of Biomedical Informatics* 52 (Dec. 2014), pp. 457–467. DOI: https://doi.org/10.1016/j.jbi.2014.06.009.

[46] Moradi Milad. “CIBS: A biomedical text summarizer using topic-based sentence clustering”. In: *Journal of Biomedical Informatics* 88 (Dec. 2018), pp. 53–61. DOI: https://doi.org/10.1016/j.jbi.2018.11.006.

[47] Nasr Azadani Mozhgan, Ghadiri Nasser, and Davoodijam Ensieh. “Graph-based biomedical text summarization: An itemset mining and sentence clustering approach”. In: *Journal of Biomedical Informatics* 84 (Aug. 2018), pp. 42–58. DOI: https://doi.org/10.1016/j.jbi.2018.06.005.

[48] Papineni Kishore et al. “BLEU: A Method for Automatic Evaluation of Machine Translation”. In: *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*. Association for Computational Linguistics, June 2002, pp. 311–318. DOI: 10.3115/1073083.1073135.

[49] Pedregosa Fabian et al. “Scikit-learn: Machine Learning in Python”. In: *Journal of Machine Learning Research* 12.85 (2011), pp. 2825–2830. URL: http://jmlr.org/papers/v12/pedregosa11a.html.

[50] Peng Yifan, Yan Shankai, and Lu Zhiyong. “Transfer Learning in Biomedical Natural Language Processing: An Evaluation of BERT and ELMo on Ten Benchmarking Datasets”. In: *Proceedings of the 18th BioNLP Workshop and Shared Task*. Association for Computational Linguistics, Aug. 2019, pp. 58–65. DOI: 10.18653/v1/W19-5006.

[51] Pennington Jeffrey, Socher Richard, and Manning Christopher. “GloVe: Global Vectors for Word Representation”. In: *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Association for Computational Linguistics, Oct. 2014, pp. 1532–1543. DOI: 10.3115/v1/D14-1162.

[52] Řehůřek Radim and Sojka Petr. "Software Framework for Topic Modelling with Large Corpora". English. In: *Proceedings of the LREC 2010 Workshop on New Challenges for NLP Frameworks*. http://is.muni.cz/publication/884893/en. Valletta, Malta: ELRA, May 2010, pp. 45–50.

[53] Reimers Nils and Gurevych Iryna. *Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks*. Aug. 2019. arXiv: 1908.10084 [cs.CL].

[54] Rohanian Omid et al. "On the effectiveness of compact biomedical transformers". In: *Bioinformatics* 39.3 (Feb. 2023), btad103. ISSN: 1367-4811. DOI: 10.1093/bioinformatics/btad103.

[55] Rousseeuw Peter J. "Silhouettes: A graphical aid to the interpretation and validation of cluster analysis". In: *Journal of Computational and Applied Mathematics* 20 (Nov. 1987), pp. 53–65. DOI: https://doi.org/10.1016/0377-0427(87)90125-7.

[56] Sanh Victor et al. *DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter*. Mar. 2020. arXiv: 1910.01108 [cs.CL].

[57] scikit-learn developers. *Clustering performance evaluation*. URL: https://scikit-learn.org/stable/modules/clustering.html#clustering-performance-evaluation.

[58] See Abigail, Liu Peter J., and Manning Christopher D. *Get To The Point: Summarization with Pointer-Generator Networks*. Apr. 2017. arXiv: 1704.04368 [cs.CL].

[59] Vaswani Ashish et al. *Attention Is All You Need*. Dec. 2017. arXiv: 1706.03762 [cs.CL].

[60] Wallace Byron C. et al. "Generating (Factual?) Narrative Summaries of RCTs: Experiments with Neural Multi-Document Summarization". In: (May 2021). URL: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC8378607/.

[61] Wang Mengqian et al. “A systematic review of automatic text summarization for biomedical literature and EHRs”. In: *J. Am. Med. Inform. Assoc.* 28.10 (Sept. 2021), pp. 2287–2297. URL: https://pubmed.ncbi.nlm.nih.gov/34338801/.

[62] Wang Wenhui et al. *MiniLM: Deep Self-Attention Distillation for Task-Agnostic Compression of Pre-Trained Transformers*. Apr. 2020. arXiv: 2002.10957 [cs.CL].

[63] Wikipedia contributors. *Cluster analysis — Wikipedia, The Free Encyclopedia*. 2023. URL: https://en.wikipedia.org/w/index.php?title=Cluster_analysis&oldid=1166878665.

[64] Yasunaga Michihiro, Leskovec Jure, and Liang Percy. *LinkBERT: Pretraining Language Models with Document Links*. Mar. 2022. arXiv: 2203.15827 [cs.CL].

[65] Zhang Jin, Cheng Xueqi, and Xu Hongbo. “GSPSummary: A Graph-Based Sub-topic Partition Algorithm for Summarization”. In: *Information Retrieval Technology*. Springer Berlin Heidelberg, 2008, pp. 321–334. DOI: 10.1007/978-3-540-68636-1_31.

[66] Zhang Tianyi et al. *BERTScore: Evaluating Text Generation with BERT*. Feb. 2020. arXiv: 1904.09675 [cs.CL].

[67] Zhang Yijia et al. “BioWordVec, improving biomedical word embeddings with subword information and MeSH”. In: *Scientific Data* 6.1 (May 2019), p. 52. ISSN: 2052-4463. DOI: 10.1038/s41597-019-0055-0.

[68] Zheng Xin et al. “Subtopic-driven Multi-Document Summarization”. In: *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*. Association for Computational Linguistics, Oct. 2019, pp. 3153–3162. DOI: 10.18653/v1/D19-1311.

[69] Zhu Yukun et al. *Aligning Books and Movies: Towards Story-like Visual Explanations by Watching Movies and Reading Books*. June 2015. arXiv: 1506.06724 [cs.CV].

A Clustering dataset creation

To evaluate the clustering algorithms we discussed in Chapter 4, we created a dataset of PubMed abstracts as, to the best of our knowledge, none of the existing datasets are suited for our use case.

The dataset is composed of 66,388 abstracts divided into 20 different entries. Each entry contains abstracts related to a disease that we selected from lists maintained by NHS inform[1] and the National Organization for Rare Disorders[2].

For each query, we used the PubMed API[3] to fetch PMIDs and abstracts. We set an upper bound of 5000 abstracts and, for easier reproducibility, we only queried 2022 publications. Of the fetched articles, we excluded those without an abstract.

In Table A.1, we report the queries we used and the number of abstracts for each of them.

[1] `https://www.nhsinform.scot/illnesses-and-conditions/a-to-z`

[2] `https://rarediseases.org/rare-diseases`

[3] `https://www.ncbi.nlm.nih.gov/books/NBK25499/`

Query	**Abstracts**
asthma	4427
attention deficit hyperactivity disorder	2359
autistic spectrum disorder	4463
brain tumours	4713
bronchitis	917
common cold	900
covid-19	4546
dementia	4707
depression	4851
gilbert syndrome	373
heart failure	4518
hepatitis B	3454
hiv	4591
kidney cancer	4591
meningitis	3440
migraine	2122
multiple sclerosis	4669
radiation sickness	1633
type 2 diabetes	4722
yellow fever	392
Total	66,388

Table A.1: Clustering evaluation dataset composition