NLP Lab project: Text summarization; midterm recap

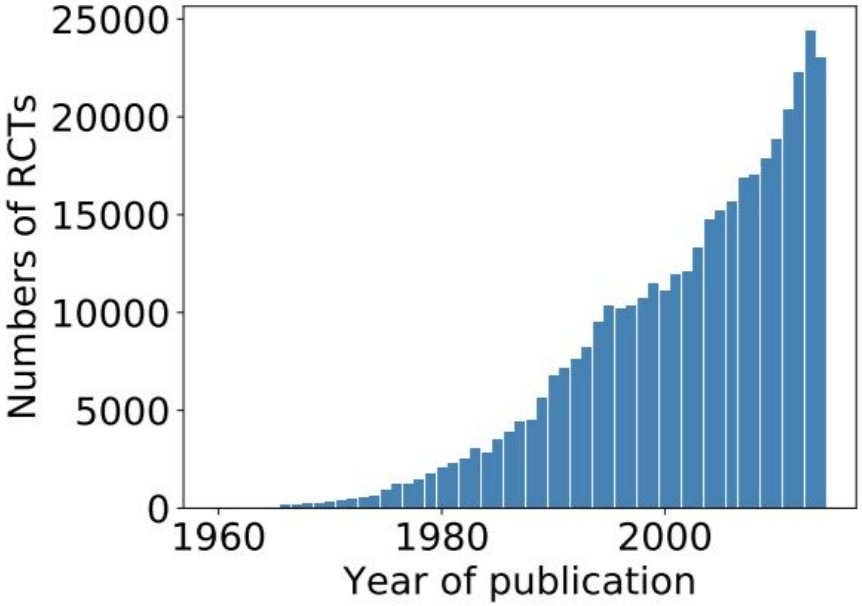
SuSe2018
Ricardo Martinez
Dejan Đukić
Supervisor: Diego Esteves

What is text summarization?

- Summary: ‘A brief statement or account of the main points of something’
- In NLP: it is the attempt of producing a brief, accurate and coherent summary of a longer text document
- The ideal summary should be as fluent as a new standalone document
- Two main approaches:
  - Extractive (selects the most important preexisting sentences from the source)
  - Abstractive (generating novel phrases distilling the meaning of the source)

Why use text summarization?

- Exponential growth in data availability
- Staying up-to-date with scientific literature almost unfeasible (PubMed currently containing over 26 million articles (June 2018))
- Goal: Obtaining as much data as possible in a shortest possible amount of time

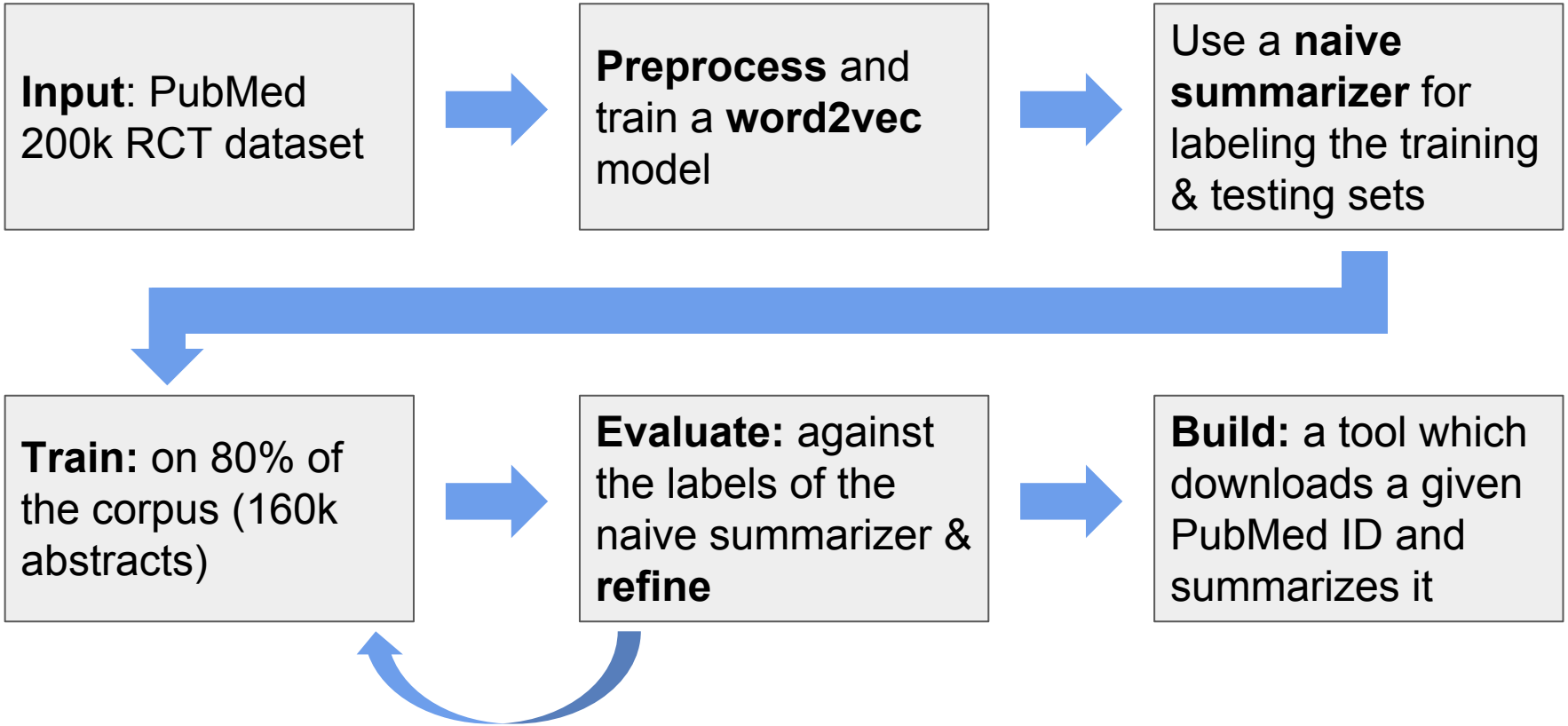


RCT - Randomized Control Trials

Approaches

- Extractive: Naive frequentist approach, relying on a metric (i.e. inverse-document frequency) to extract interesting parts of the source and join them back together:
  - Original Text: *Alice and Bob took the train to visit the zoo. They saw a baby giraffe, a lion, and a flock of colorful tropical birds.*
  - Extractive Summary: *Alice and Bob visit the zoo. saw a flock of birds.*
- Abstractive: Human-like, non-verbatim rephrasing of the key points from the source; represented by sequence2sequence learning models:
  - (Same original text as above) Abstractive summary: *Alice and Bob visited the zoo and saw animals and birds.*
- Important to note that a purely abstractive approach does not yet exist (reliance on the extractive pre-processing steps)

Overview of the process



The dataset

- **PubMed 200k RCT** (randomised control trials publications)
  - https://github.com/Franck-Dernoncourt/pubmed-rct
- 2.3 million sentences
- Sentences labeled with their role in the abstract (background, objective, method, result or conclusion);
  - won't be using these as features as they might simply overfit to conclusion sentences
- Will be used for retraining a word2vec model, producing a domain-specific word embeddings layer
- Will be used for training the sequence2sequence LSTM producing the summaries

Naive Summarization

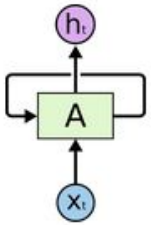
- Based on the premise that the most recurrent words in the text probably cover the major topic of the text.
- Therefore: obtain word frequencies & return the sentences where the most frequent words occur
- The Process
  - Sentence & Word Tokenization
  - Stopword & Punctuation removal
  - Word Count/Frequency calculations
  - Return: Top n-sentences
  - Use these Naive Summaries as the labels for seq2seq model

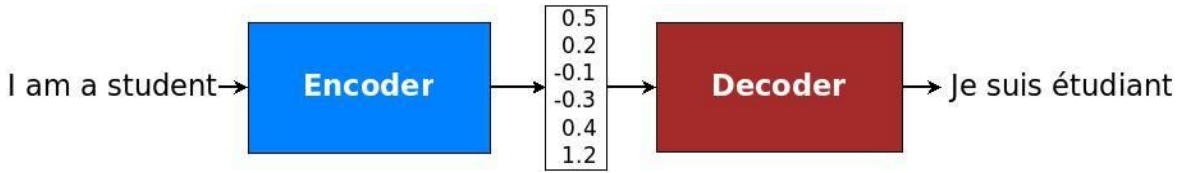
Word2vec

- Group of related models which produce **word embeddings**;
  - Words or phrases that are mapped to vectors of real numbers
- It is a shallow Neural Network which reconstructs linguistic context
- Takes our **RTC** data and produces a domain-specific vector space
- Trains words against other words that neighbor them in the input corpus
- Already done

Sequence2Sequence

- Composed of two Recurrent Neural Networks -- *to maintain the **Persitence of Information**---i.e. links output of previous computations to later computations*
- **Encoder** : processes the variable-length input and maps it to a fixed-length ***thought vector***--which represents meaning
- **Decoder**: generates the output, i.e. maps the vector representation back to a variable-length target sequence
- Word2vec embedding is used by our RNN (Encoder/Decoder)
- RNN Summarization is trained/tested using Naive Summarization





Evaluation: Rouge-N

- An N-gram measure between the model and the gold summary
  - ratio of the count of N-gram phrases which occur in **both the model and gold summary**, to the count of all N-gram phrases that are present in the gold summary.
- Rouge-1: #matching words / #words in gold standard
- Rouge-2: #matching side-by-side words / #side-by-side words in gold standard
- Rouge-3: #matching side-by-side-by-side words…. Etc.

Example:

Gold Summary: *A good diet must have apples and bananas.*
Model: *Apples and bananas are must for a good diet.*
ROUGE-1: 7/8 = 0.875
ROUGE-2: 4/7 = 0.571

Architecture

- Use **word2vec** create ***embedding*** for Seq2Seq RNNs
  - word2vec is fed the entire RTC corpus
  - outputs **embedding** -save as pickle
- Train Seq2Seq RNN with training split from **naive_summarizer** label summary
  - naive summarizer produces label-summaries from our Abstracts --the gold-standard
  - a split of the abstracts is created, and only the Abstracts are fed to seq2seq
  - **embeddings** pickle is loaded
  - Sequential stacked LSRT RNN created

Tool/API

- Once the training is satisfactory, we train in the Wild!
- **Entrez** - a NCBI Molecular Database system allows us to access **PubMed** Abstracts using the Biopython library's Entrez class
- Tool will access PubMed and download X number of abstracts
- We use naive_summarizer to get a 'standard summary'
- The abstracts are runned through our model & compared!
- Satisfaction!