Document Summarization using Neural Networks

HARSHAL BHATIA 1DA15CS037

KALPITHA V BEKAL 1DA15CS046

MANU HEGDE 1DA15CS067

MOHAN KRISHNA S 1DA15CS069

INTRODUCTION

Document summarization has been an requirement for a long time ever since the digitization of paperwork began. However the contention has always been is to handle the complexity and nuances of human language.
For long time, approaches to document summarization depended on patterns found about language by linguistics.

This has the drawback of intensively reliant on the given language or given dialect of a language. These systems were far away from true understanding of human language.

In 2013, with resurgence of neural networks, new techniques were developed that allowed for understanding and interpretation of human language. This lead to breakthroughs in translation and NMT(Neural Machine Translation) was developed. This lead to the development of modern summarization models which can try to understand and represent human languages efficiently.

Major types of Document Summarization

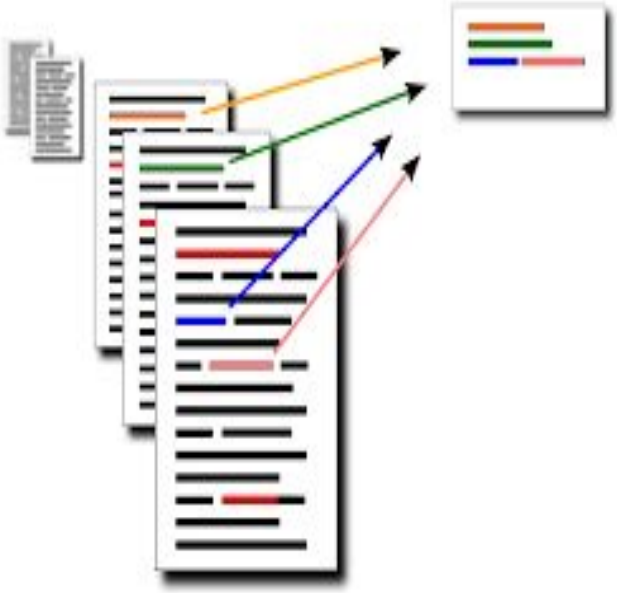
- **Extractive document summarization** is generating hand coded rule based summaries by selecting phrases, sentences which may effectively represent the gist of the document.

- **Abstractive document summarization** is generating summaries that may contain words other than those present in the original document by understanding the meanings of sentences in the source document.

NEED FOR A DOCUMENT SUMMARIZER

- Document summarization is the process of expressing a large and detailed text content in form of a few sentences that carry the overall meaning.
- Helps in quick outlining of a document and also makes it simpler for the reader to understand.
- Particularly helpful in quick understanding of legal or scientific documents which are often very verbose.
- There is an enormous amount of textual material, and it is only growing every single day.
- There is a great need to reduce much of this text data to shorter, focused summaries that capture the salient details, both so we can

- navigate it more effectively
- check whether the larger documents contain the information that we are looking for.



USE CASES

- **Newsletters**
  Many weekly newsletters take the form of an introduction followed by a curated selection of relevant articles. Summarization would allow organizations to further enrich newsletters with a stream of summaries, which can be a particularly convenient format in mobile.

- **Social media marketing**
  Companies producing long-form content, like whitepapers, e-books and blogs, might be able to leverage summarization to break down this content and make it sharable on social media sites like Twitter or Facebook. This would allow companies to further re-use existing content.

- **Search marketing and SEO**
  When evaluating search queries for SEO, it is critical to have a well-rounded understanding of what your competitors are talking about in their content. This has become particularly important since Google updated its algorithm and shifted focus towards topical authority (versus keywords). Multi-document summarization can be a powerful tool to quickly analyze dozens of search results, understand shared themes and skim the most important points.

- **Financial research**
  Investment banking firms spend large amounts of money acquiring information to drive their decision-making, including automated stock trading. When you are a financial analyst looking at market reports and news everyday, you will inevitably hit a wall and won’t be able to read everything. Summarization systems tailored to financial documents like earning reports and financial news can help analysts quickly derive market signals from content.

- **Books and literature**
  Summarization can help consumers quickly understand what a book is about as part of their buying process.

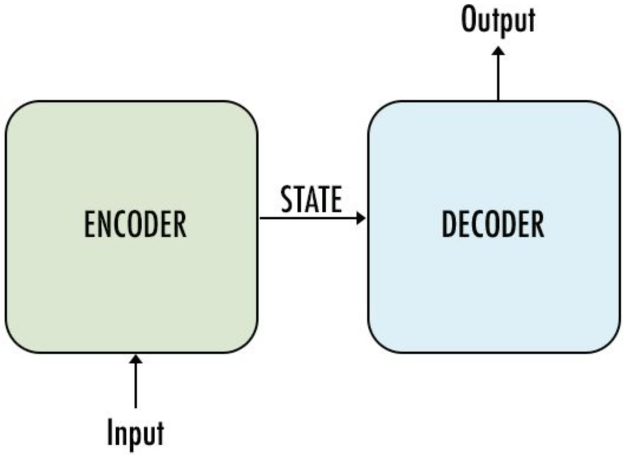
- **Internal document workflow**
  Large companies are constantly producing internal knowledge, which frequently gets stored and under-used in databases as unstructured data. These companies should embrace tools that let them re-use already existing knowledge. Summarization can enable analysts to quickly understand everything the company has already done in a given subject, and quickly assemble reports that incorporate different points of view.

DRAWBACKS OF EXISTING MODELS

PURELY EXTRACTIVE RULE BASED SYSTEMS

- These systems need to be re-designed for every language.
- Features to be used have to be decided manually
- They fail to capture overall meaning and context in larger documents.

PURELY RECURRENT ENCODER DECODER NETWORKS



- These networks have the main drawback of repetition and coverage.
- This means that the summary yielded contains repeated phrases, sentences and either omit or repeat topics, contexts, present in the source document.

- Inability to handle new words
- Words previously not seen during training results in outputting ‘UNK’ in place of such words.
- Results in summaries that are incomplete and unable to convey the full meaning of the document.
- High possibility of misrepresentation and misunderstanding of the information.

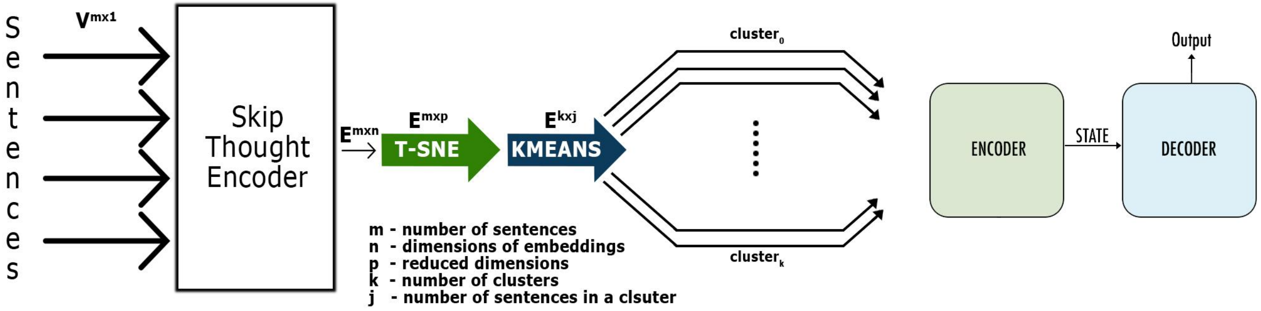
MODELS WITH EXPLICIT COPY MECHANISM

- Models like pointer generator networks have explicit copy mechanism
- The model tries to copy a word from source text if the decoder is unable to produce output for a portion of text that contains words not seen by the model during training.
- These models very often end up copying text from source document even when there are no unknown words.
- They also need explicit mechanisms to track coverage of source document to prevent repetition

TRANSFORMERS AND OTHER HYBRID MODELS

- These models try to solve the problem of copying, but do not perform well in case of context coverage
- They tend to mix up sentences of different contexts.
- Although their output is semantically correct, they can end up generating misinterpreted summaries.
- They are also one of the most computationally intensive models employed for summarization

PROPOSED MODEL



Sentences

V^{mx1}

Skip Thought Encoder

E^{mxn}

E^{mxp}

T-SNE

E^{kxj}

KMEANS

cluster$_0$

cluster$_k$

ENCODER

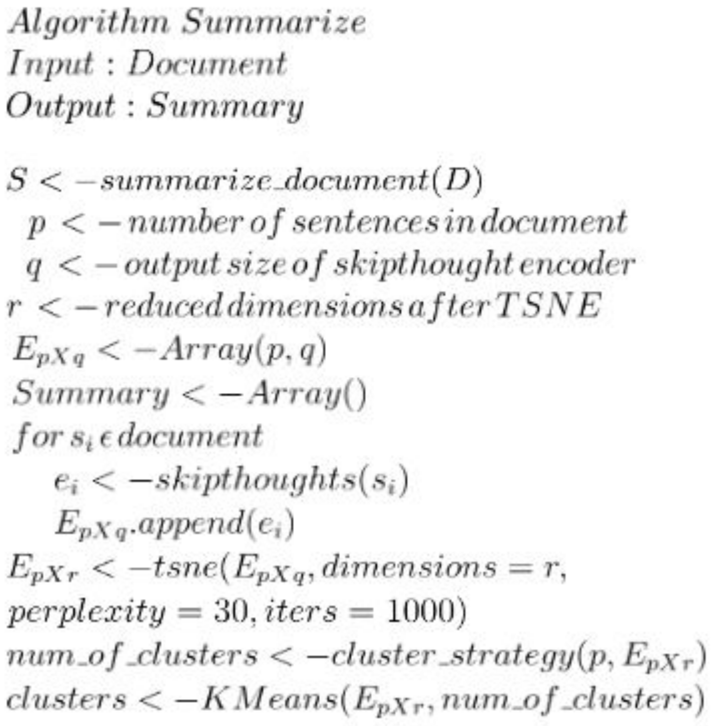
STATE

DECODER

Output

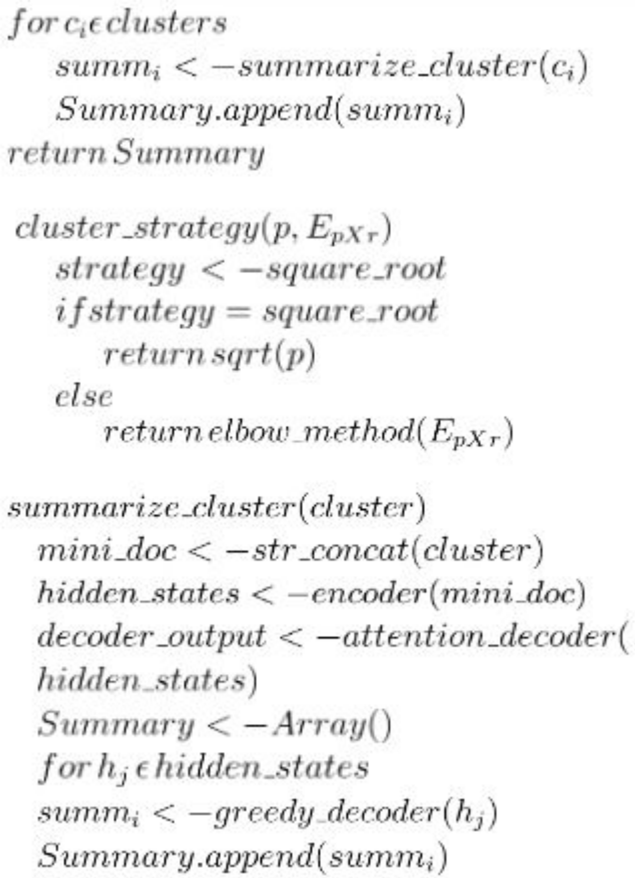
m - number of sentences
n - dimensions of embeddings
p - reduced dimensions
k - number of clusters
j - number of sentences in a clsuter

ALGORITHM



```
$Algorithm\ Summarize$
$Input : Document$
$Output : Summary$

$S <- summarize\_document(D)$
  $p <- number\ of\ sentences\ in\ document$
  $q <- output\ size\ of\ skipthought\ encoder$
$r <- reduced\ dimensions\ after\ TSNE$
$E_{pXq} <- Array(p, q)$
$Summary <- Array()$
$for\ s_i \epsilon\ document$
    $e_i <- skipthoughts(s_i)$
    $E_{pXq}.append(e_i)$
$E_{pXr} <- tsne(E_{pXq}, dimensions = r,$
$perplexity = 30, iters = 1000)$
$num\_of\_clusters <- cluster\_strategy(p, E_{pXr})$
$clusters <- KMeans(E_{pXr}, num\_of\_clusters)$
```



```
$for\ c_i \epsilon\ clusters$
    $summ_i <- summarize\_cluster(c_i)$
    $Summary.append(summ_i)$
$return\ Summary$

$cluster\_strategy(p, E_{pXr})$
    $strategy <- square\_root$
    $if\ strategy = square\_root$
        $return\ sqrt(p)$
    $else$
        $return\ elbow\_method(E_{pXr})$

$summarize\_cluster(cluster)$
  $mini\_doc <- str\_concat(cluster)$
  $hidden\_states <- encoder(mini\_doc)$
  $decoder\_output <- attention\_decoder($
  $hidden\_states)$
  $Summary <- Array()$
  $for\ h_j \epsilon\ hidden\_states$
  $summ_i <- greedy\_decoder(h_j)$
  $Summary.append(summ_i)$
```

WORKING OF THE ALGORITHM

The Algorithm operates as follows.

- Obtain embeddings for every sentence.
- Reduce dimensionality of embeddings using T-SNE.
- Cluster the embeddings using K-Means.
- Obtain one summary for each cluster separately.

A. Obtain Embeddings for every Sentence

Skip-Thought encoder to obtain fixed length sentence embeddings, for sentences of varying length. This can step is necessary to obtain meaningful representation of a sentence in form of a vector i.e an embedding in a higher dimension, such that the sentences with similar meaning are mapped close to each other and vice versa. Also this space is such that a dot product between two sentences can be used to obtain a measure of their similarity or dissimilarity. The length of these vectors is 4800. Using Skipthought sentence embeddings can solve the misunderstanding of sentences due to UNKs, because of its vocabulary expansion capability to handle previously unknown words.

B. Reduce Dimensionality of Embeddings using T-SNE

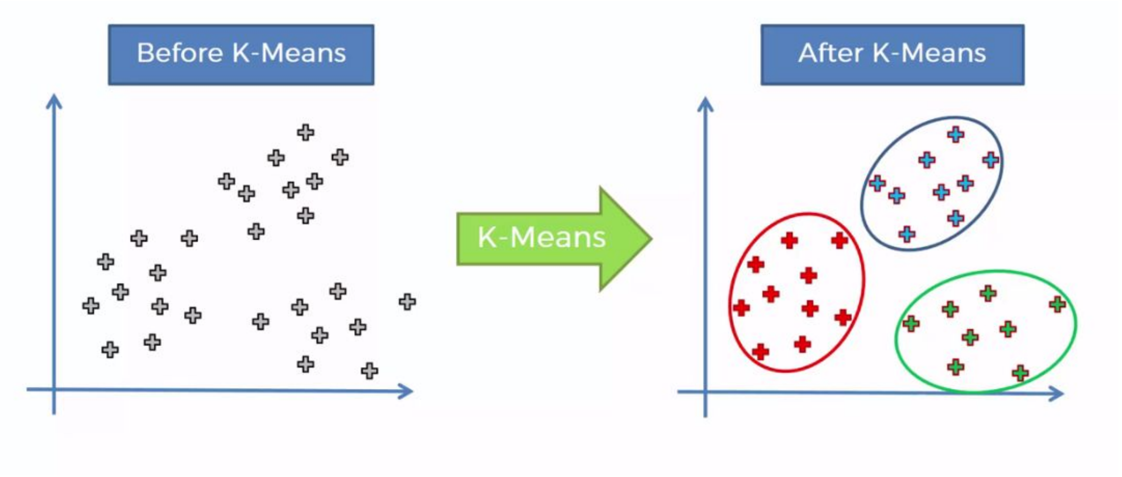
T-SNE (t-Distributed Stochastic Neighbour Embedding)[9] is widely known algorithm for dimensionality reduction. The T-SNE algorithm is designed to project a set of points in higher dimension to a lower dimension, yet trying to minimize the loss in pairwise distances between all set of input points.

Where is the probability density under a Gaussian centered at .

Since the sentence embeddings each are of length 4800. Effectively projecting them to lower dimension without loss of information is necessary and would lead to better performance. This would result in much better result from K-Means Clustering. Hence T-SNE was chosen over other algorithms like SVD or PCA.

C.Cluster the embeddings using K-Means

K Means is used for clustering the sentence embeddings. The number of clusters is directly related to the desired length of summary. Usually this number is chosen as square root of the number of sentences in the document. This can solve the problem of loss of context as well as requirement of large computational resource for large documents. Here, each cluster is formed such that all sentences in a cluster have similar meaning or are very closely related to each other. Hence the document is effectively divided based on content similarity.



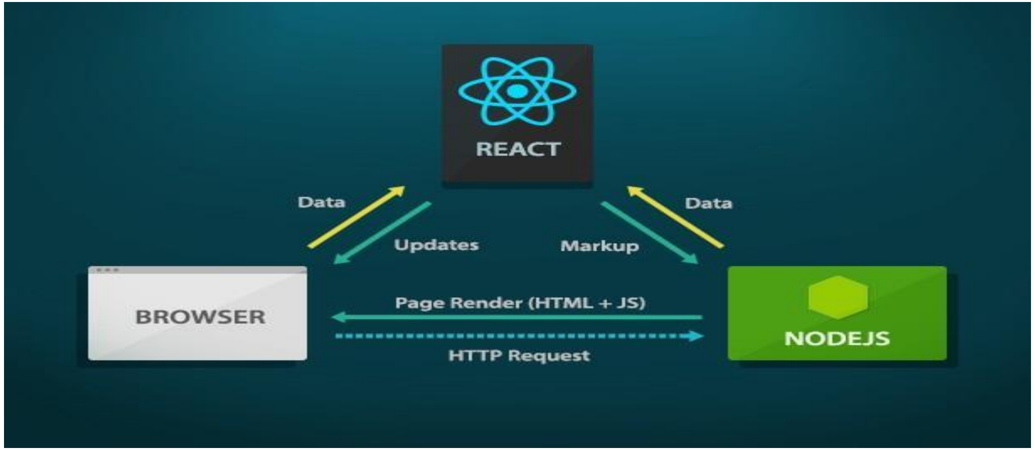
D.Obtain one summary for each cluster separately

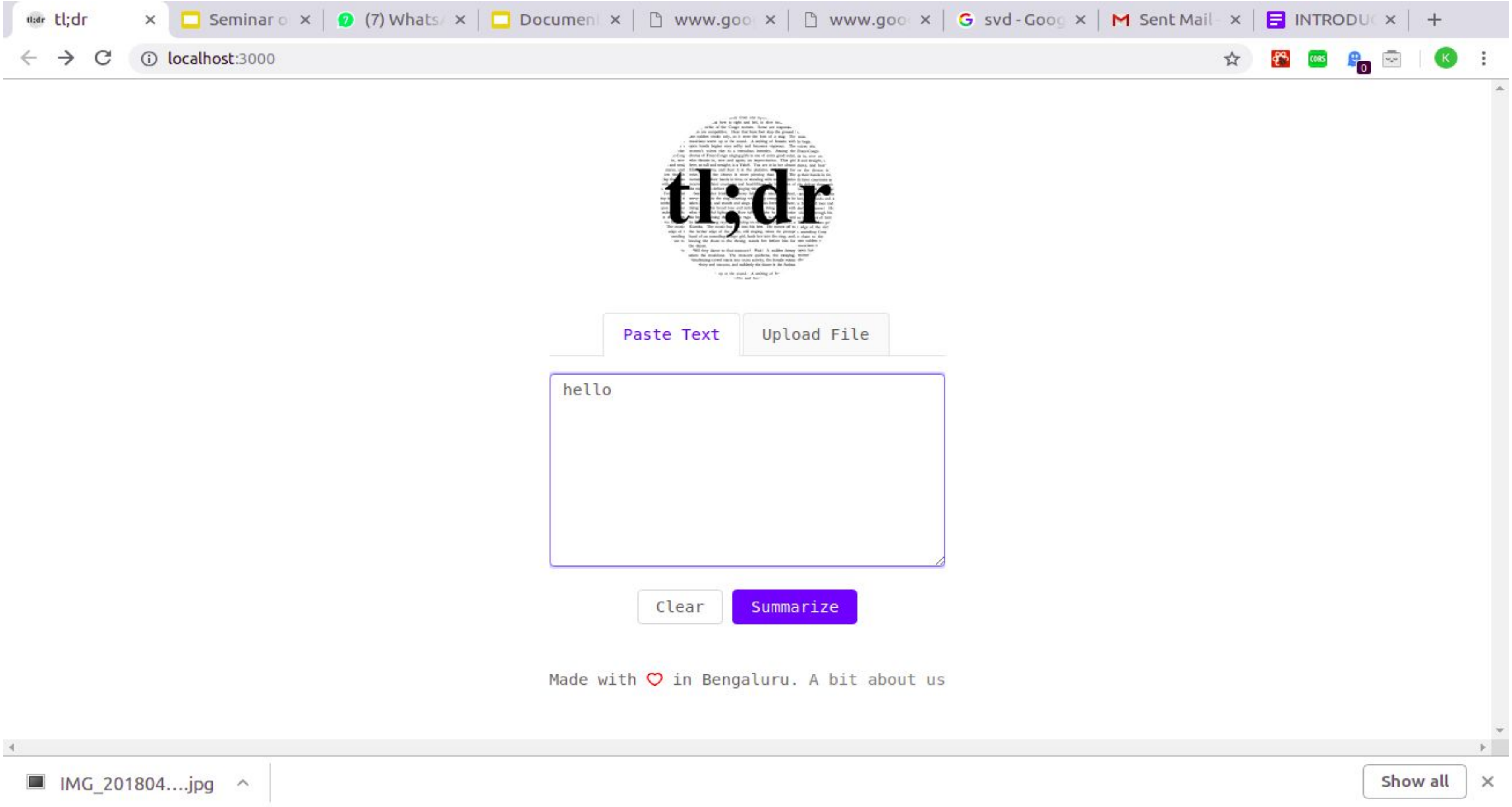
Finally, set of sentences in each cluster are fed to an encoder decoder model. This helps in solving the problem of coverage tracking. Also since all sentences in a cluster have similar meaning or are closely related to each other, the encoder decoder model would process sentences of same or similar topic. This would allow for a model to be used that is computationally less intensive due to smaller input size and smaller context.

IMPLEMENTATION

ReactJS

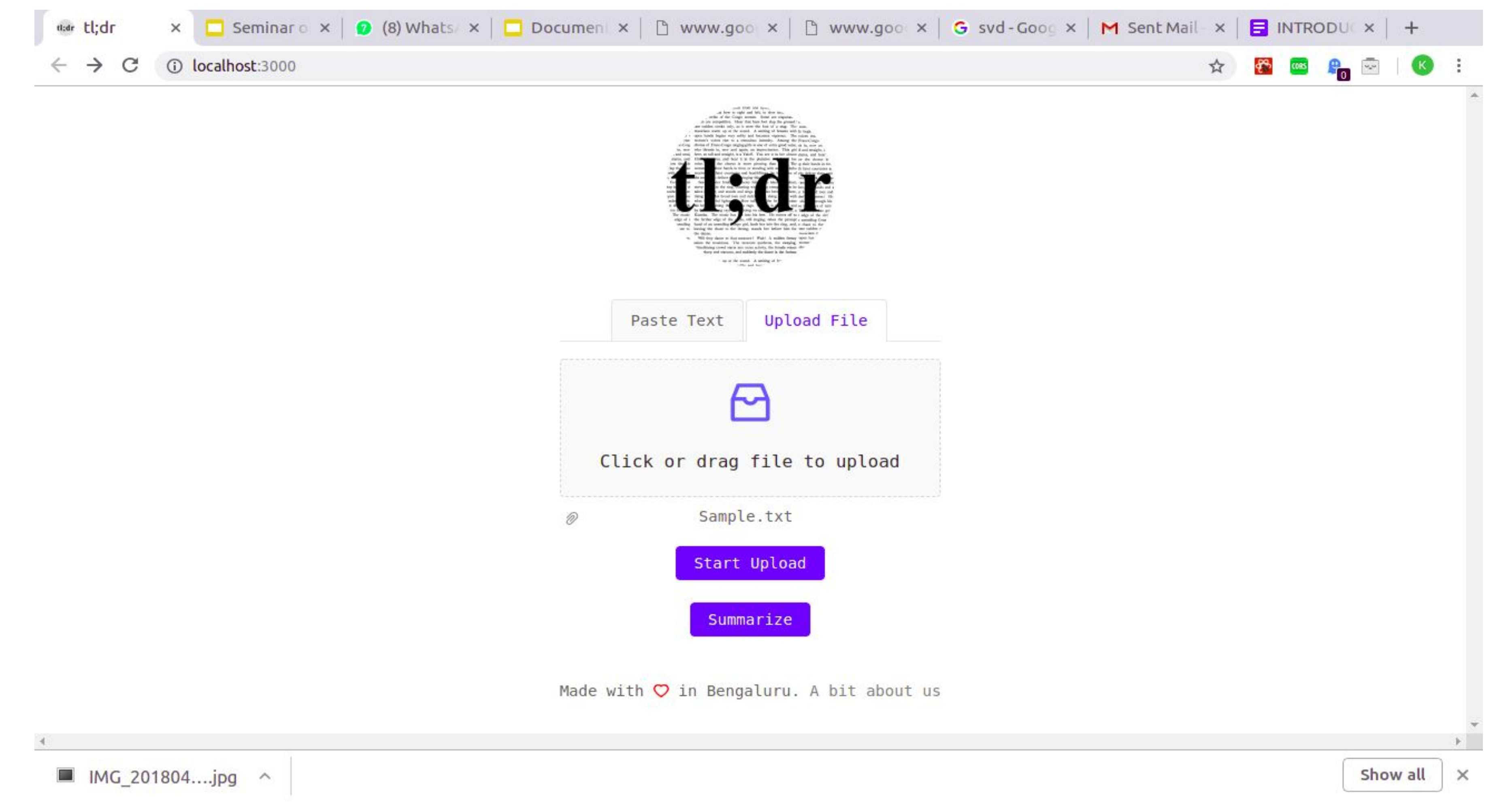
A Javascript library for building user interfaces. React makes it painless to create interactive UIs. Design simple views for each state in your application, and React will efficiently update and render just the right components when your data changes.





hello

Made with ♡ in Bengaluru. A bit about us



Click or drag file to upload

Made with ♡ in Bengaluru. A bit about us

Skip-Thought

“Skip-Thought Vectors” or simply “Skip-Thoughts” is name given to a simple Neural Networks model for learning fixed length representations of sentences in any Natural Language without any labelled data or supervised learning. Fixed representations make it easy to replace any sentence with an equivalent vector of numbers. This makes the process of understanding, acting upon or responding to Natural Language mathematically straightforward.

T-SNE

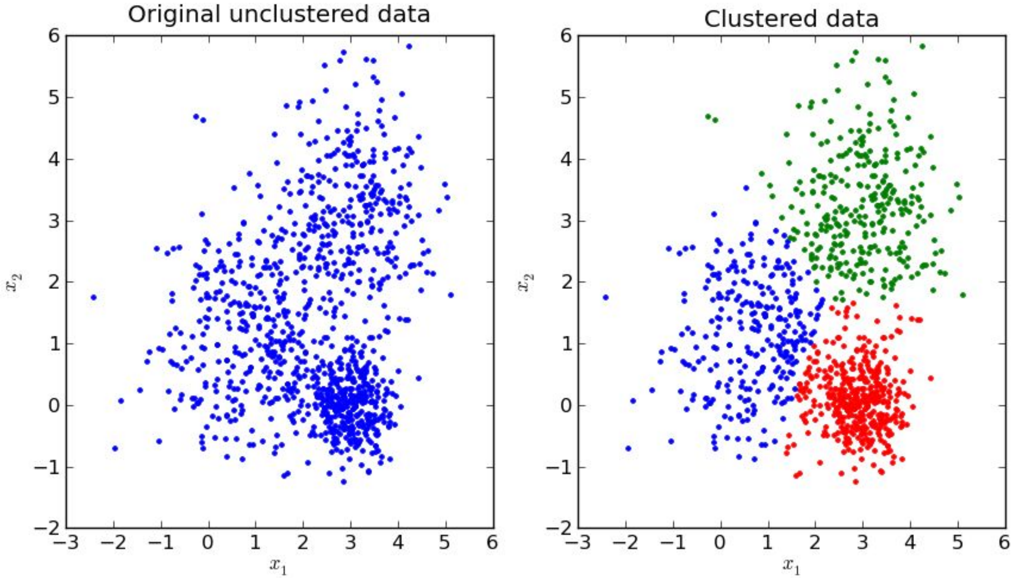
T-distributed Stochastic Neighbor Embedding (t-SNE) is a machine learning algorithm for visualization developed by Laurens van der Maaten and Geoffrey Hinton. It is a nonlinear dimensionality reduction technique well-suited for embedding high-dimensional data for visualization in a low-dimensional space of two or three dimensions. Specifically, it models each high-dimensional object by a two- or three-dimensional point in such a way that similar objects are modeled by nearby points and dissimilar objects are modeled by distant points with high probability.

Dimensionality reduction can be achieved in the following ways:

- **Feature Elimination**: You reduce the feature space by eliminating features. This has a disadvantage though, as you gain no information from those features that you have dropped.
- **Feature Selection**: You apply some statistical tests in order to rank them according to their importance and then select a subset of features for your work. This again suffers from information loss and is less stable as different test gives different importance score to features. You can check more on this here.
- **Feature Extraction**: You create new independent features, where each new independent feature is a combination of each of the old independent features. These techniques can further be divided into linear and non-linear dimensionality reduction techniques.

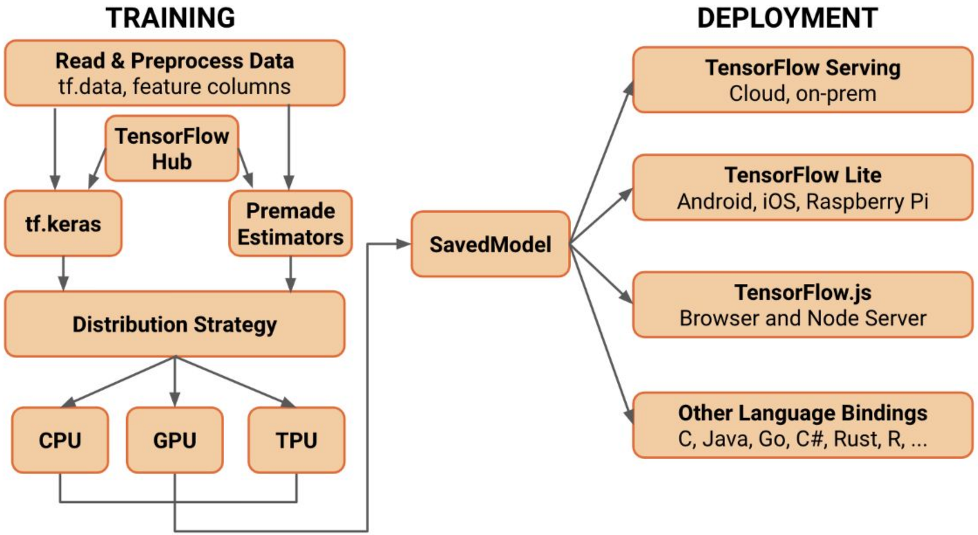
K-Means

K-means clustering is a type of unsupervised learning, which is used when you have unlabeled data (i.e., data without defined categories or groups). The goal of this algorithm is to find groups in the data, with the number of groups represented by the variable *K*. The algorithm works iteratively to assign each data point to one of *K* groups based on the features that are provided. Data points are clustered based on feature similarity.



TensorFlow

TensorFlow is an end-to-end open source platform for machine learning. It has a comprehensive, flexible ecosystem of tools, libraries and community resources that lets researchers push the state-of-the-art in ML and developers easily build and deploy ML powered applications.



FUTURE SCOPE

Documents are always time consuming to deal with. In this era of Internet, the amount of data generated only keeps growing. The rate at which the information is growing is tremendous. Hence, very soon in the future if not already, a document summarizer will be a necessity. This research could also be a stepping stone towards achieving the goal of building a multilingual summarization system.