NLP Project Report

Understanding Query Focused Summarization

Priyank Modi (20171055) & Aashna Jena (20171095)

Link to Repo : https://github.com/priyankmodiPM/NLP_Project.git

Aims Of The Project

Understanding Automatic Summarisation : The aim of our project was to understand how automatic summarisation works, and how summarises can be built specific to given queries. Generic summarisation has two major approaches - extractive and abstractive. Extractive summarization deals with selecting relevant sentences from the given text and joining them to form a summary. Abstractive summarisation aims to improve the fluency and coherence of summaries, by generating summaries that capture the relevant content of the document, and paraphrase them into a coherent sequence of sentences.

Investigating and implementing a summarisation model : To understand the state-of-art models of both types, we read up on multiple papers (mentioned below) and investigated the procedure, performance and need for both approaches. We implement the "pointer generator network model with attention" for abstractive summaries. *(Paper 1. Under Literature section)*

Understanding the problem of Query Focused Summarisation : Next, we aimed to understand how to build on top of these to include query relevance. The problem statement is - given a document(s) and a query, we want to output a summary which captures the document's content relevant to that query. We read up on several papers (mentioned below) and compared the different approaches for single vs multi-word queries, length of summaries and handling of single/multi document situations.

Implementing and comparing the generic and query focused summarser : We aimed to implement both the generic and query-focused summarisation models *(Refer paper 6. In Literature Section)* side by side and qualitatively analyse the differences between the outputs of (a) Both models for the same articles and (b) The query focused model of two different queries for the same article. We wished to understand how the presence of a query is affecting a summary regarding :

1. Is the coverage similar? Does a query-focused summariser cover all content from the query's viewpoint or only looks for the sentences with mention of the keyword(s),
2. How much is the query able to affect the summary (how different is it from the generic summary). Does it only add content to the generic summary? Or is some information being omitted/replaced and
3. How well is the query understood? How is the content different when two different queries are applied to the same article? If there are any common content and if yes, is it coming from the generic summary? WHY is the content which is different is chosen in each case. How is the query actually affecting the summary?

Analyse the concerns with state-of-the-art models : We wished to note the logical shortcomings for two models that we studied with regard to how multi-document situations are handled, how the query is incorporated in the model, how appropriate the dataset is and how the relevance is measured. We discussed some of these concerns with our advisor during the intermediate evaluation, and implemented some of the simple modifications.

Literature

Abstractive Summarization :

1. Abigail See, Peter J. Liu and Christopher D. Manning. Get To The Point: Summarization with Pointer-Generator Networks. *(implemented for generic summarisation)*
2. Xinyu Hua, Lu Wang. A Pilot Study of Domain Adaptation Effect for Neural Abstractive Summarization
3. Angela Fan, David Grangier, Michael Auli. Controllable Abstractive Summarization
4. Linqing Liu, Yao Lu, Min Yang, Qiang Qu, Jia Zhu, Hongyan Li. Generative Adversarial Network for Abstractive Text Summarization

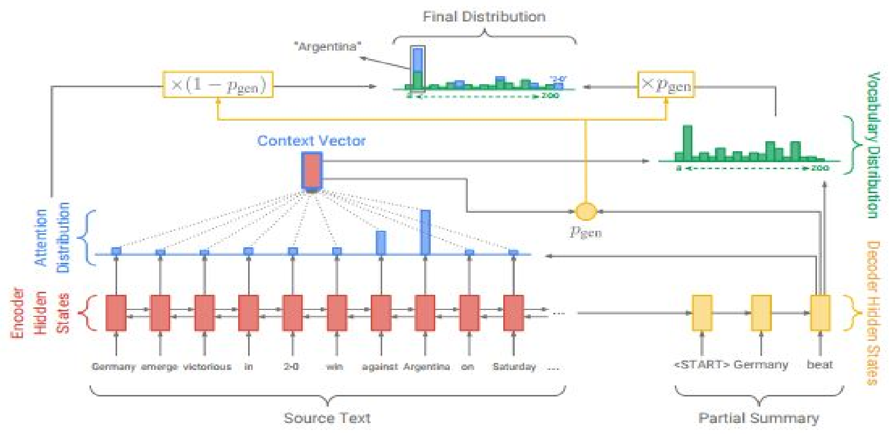
Query Focused Summarisation :

5. Tal Baumel, Matan Eyal, Michael Elhadad. Query Focused Abstractive Summarization: Incorporating Query Relevance, Multi-Document Coverage, and Summary Length Constraints into seq2seq Models
6. ***(Paper used for paper presentation)*** Johan Hasselqvist, Niklas Helmertz, Mikael Kågebäck. Query-Based Abstractive Summarization Using Neural Networks. Link to the paper *(implemented for query focused summarisation)*

Observations and Analysis

Extractive vs Abstractive approaches : As explained above, extractive methods select relevant phrases of the input document and concatenate them to form a summary (like "copy-and-paste"). This makes them quite robust since they use existing natural-language phrases that are taken straight from the input. But they lack in flexibility since they cannot use novel words or connectors. They also cannot paraphrase like people sometimes do. As a result, the summaries tend to be incoherent, especially because there is context missing for understanding most sentences in the summary. Abstractive summaries are an improvement in this regard. They can use words that were not in the original input. It enables to make more fluent and natural summaries. But it is also a much harder problem as you now require the model to generate coherent phrases and connectors.

Comparison between different abstractive approaches : Early neural approaches to abstractive summarisation employ the concepts of encoder-decoder networks. The encoder-decoder model is composed of encoder and decoder like its name. The encoder converts an input document to a latent representation (vector), and the decoder generates a summary by using it. But the problem of How to set the focus on the important sentence, keyword. This was handled by the neural “Attention” models, which added an attention layer to the encoder decoder networks (between the encoder and decoder), which calculates the importance of a word (how much “attention” a word must receive) in predicting the next word (from the vocabulary, not just from the text), given the last prediction and the hidden state. The problem of handling rare words/ named entities was solved with pointer-generator networks (Paper 1 in Literature section) [figure below]. The gist of the paper is to maintain a pointer to the “current” word in the text sequence and learn to predict probabilities of whether a word should be copied from the text, or a word from the vocabulary should be generated. This extra layer of computation is added on top of the neural attention encoder-decoder network. A different adversarial approach, using two-player minimax method was explored in [paper 4]. The objective is for one player to generate an automatic summary and for the other to learn to distinguish it as human or machine-generated. The problem is that the commonality that this approach will learn over 1000s of articles is the grammar and fluency, not how the content is to be selected. [Paper 2] explores how to make use of out-domain knowledge to predict summaries. “How to select relevant information” is the transferable component whereas the “style of writing” is learnable only from in-domain training samples. We have implemented [paper 1] as our generic summariser.

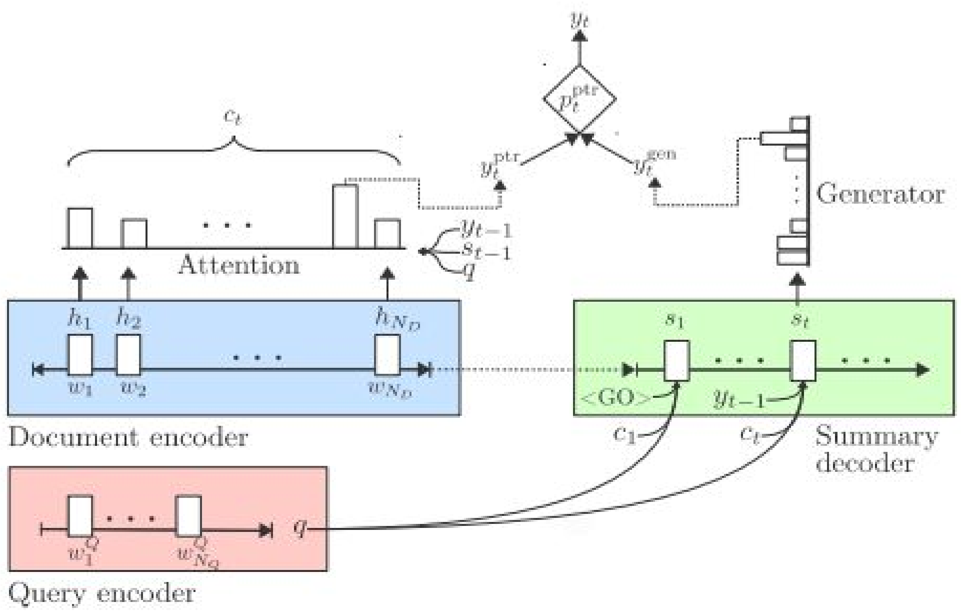


Query focused summarisation : We read [paper 5] and [paper 6] for understanding two approaches to query focused summarisation. In [paper 5], the authors use an extractive + abstractive approach. They first extract the sentences which are relevant to the query, and then employ a simple abstractive summariser with the extracted sentences as the input. Here, relevance is calculated using simple methods - TF-IDF cosine similarity and unigram overlap between query and sentence. The relevance score of the sentence is assigned to each of its words, and is multiplied with the attention scores right before applying softmax, so include relevance as a parameter in the summarisation process. For handling multi document summarisation and controlling the length, the paper discusses a simple approach. They rank all documents in their order of relevance and keep iteratively summarising till the word limit has reached. This model achieves good ROUGE scores, and changes the content of the summaries in accordance with the query. But, there are some concerns we have :

1. Handling multiple documents through ranking and summarising till a word limit is reached does not ensure coverage. If cases where one document may have very important information about the query, but only in one paragraph, it would be ranked lower than a document that has multiple sparse sentences related to the query. Length of a summary should be an indicator of the broadness/ level of detail. It should not compromise on the coverage. Hence, this approach to controlling length might not be a good idea.
2. Unigram overlap and TF-IDF cosine similarity often aren't able to extract the exact meaning of the query - especially since the query is a sentence here, not just a keyword. If I want to find "reasons for xyz", there might be no explicit mention of the word "reason" in the document, but the meaning needs to be captured. The focus of this paper is to "assess whether the mechanism we propose in order to combine relevance and

abstractive capabilities is capable of producing fluent and relevant summaries given a good relevance model". The relevance model in itself hasn't been given much attention to. The models are just compared for fluency as opposed to extractive ones.

Paper[6] explores a different approach to query-focused summarisation. In this paper, the authors use a neural attention model with a pointer generator network. Since the queries being considered here are a couple of words at maximum, a unidirectional RNN encoder encodes the query into a fixed length vector. This input is fed to the decoder's attention layer at every time step. The idea is that the updation of state S(t) is dependent on previous state S(t-1), last predicted word Y(t-1), context vector C(t) and query Q, so that the model learns to give more "attention" to the words relevant to the query. The model eventually learns to predict a sentence that leads up to include the query words. [figure below gives architecture]



We see some concerns in this approach :

1. The evaluation criteria is not clear. The authors test whether the ROUGE scores go down if a (document D1, query Q1) pair's output is compared with (Document D1, Query Q2) pair's answer. This proves that Query Q1 produces a "different" output than Query Q2, but this doesn't prove that it's better.
2. We understand that there's a need for a task-oriented dataset for QFC. The paper makes do with generating cloze-style questions out of news highlights and using entities in those highlights as "queries". We can see that the qualitative results are not as impressive, because the variation of queries is not much (all highlights of an article have more or less the same entities) and highlights are only 1-2 sentences long.

Architecture

Document Encoder : The document encoder processes an input document, generating a state for each input word. To get a representation of the context around a word, we use a bidirectional RNNencoder, so both the context before and after contribute to the representation. The RNN hidden state at time step i, h_i, and the intermediate states, $\overrightarrow{h}_i$ and $\overleftarrow{h}_i$, from the forward reader and backward reader respectively, are computed as

$$\overrightarrow{h}_i = \overrightarrow{GRU}_{doc}(\overrightarrow{h}_{i-1}, E(w_i))$$

$$\overleftarrow{h}_i = \overleftarrow{GRU}_{doc}(\overleftarrow{h}_{i-1} E(\overleftarrow{w}_i))$$

$$h_i = [\overrightarrow{h}_i, \overleftarrow{h}_i],$$

where $w_i \in V$, for the vocabulary V, is word i in the input document; $\overleftarrow{w}_i$ is word i in the reversed input; and $E(w_i)$ is the word embedding of w_i. The initial states $\overrightarrow{h}_0$ and $\overleftarrow{h}_0$ are zero vectors. The document encoder state dimensionality is denoted d_{doc} and the word embedding dimensionality d_{emb}.

Query Encoder : The query encoder is responsible for creating a fixed-size internal representation of the input query. Unlike the document encoder, the query encoder is a unidirectional RNN encoder since queries are relatively short compared to documents and we only use the final state to represent the whole query. The RNN state h_{Qi} at query word i, is updated according to $h_{Qi}=GRU_{que}(h_{Qi-1},E(w_{Qi})),q=h_{QN_Q}$, where w_Q is the input query and N_Q is the length of the query. The initial state h_{Q0} is the zero vector. The query encoder state dimensionality is denoted d_{que}.

Query Decoder : The decoder is a unidirectional RNN for constructing a summary of the input document by depending on the final state of the input encoder, the query. It utilizes soft attention, in combination with a pointer mechanism, as well as a *generator* part similar to Bahdanau et al. (2015). The query embedding q is fed as input at each decoder time step. In our model, the RNN state is updated according to $s_t=GRU_{dec}(s_{t-1},[c_t,q,E(y_{t-1})])$, where $s_0=h_{N_D}$, the final document encoder state, N_D being the number of input words; y_0 corresponds to a special <GO> token, used at the initial time step when no previous word has been predicted; c_t is the *context vector* at time step t from the attention mechanism, defined subsequently; and $y_{t-1} \in V$ is the predicted output word at time step t−1. This is either from the generator mechanism, or the pointer mechanism, also defined subsequently. The word embeddings are the same as are used in the encoder.

Pointer-Generator network : The generator outputs a word from a subset of the vocabulary $V_{gen} \subseteq V$ at each time step. The selection of the output words is done through a distribution of words in V_{gen}, computed through a softmax as $p_{gen_{tj}} = \frac{\exp(z_{tj})}{\sum_k \exp(z_{tk})}$, for $j \leq |V_{gen}|$, an index uniquely mapped to a word $w \in V_{gen}$, and z_{tj} as defined subsequently. Defining this as the probability $P_{gen_t}(w)$, we then select output word y_{gen_t} with the highest probability by $y_{gen_t} = \text{argmax}_{w \in V_{gen}} P_{gen_t}(w)$. The softmax probability depends on z_{tj}, the output from two linear transformations on the decoder state and context vector, defined as $z_t = W^{(2)}_{gen}(W^{(1)}_{gen}[s_t, c_t] + b^{(1)}_{gen}) + b^{(2)}_{gen}$, where $W^{(1)}_{gen} \in R^{d_{gen} \times (d_{dec} + d_{doc})}$, $b^{(1)}_{gen} \in R^{d_{gen}}$, $W^{(2)}_{gen} \in R^{|V_{gen}| \times d_{gen}}$ and $b^{(2)}_{gen} \in R^{|V_{gen}|}$ are trainable hyperparameters, in which d_{gen} is the dimensionality of the hidden layer. The main function of this layer is to reduce the dimensionality of the input, for reducing computation time for the final layer with size $|V_{gen}|$.

A general issue is that with a generator mechanism limited to frequent words, infrequent words cannot be generated. The pointer mechanism adds a *switch*, $p_{ptr} \in (0,1)$, at each decoder time step t, to the model. It is computed as the output of a linear transformation fed through a sigmoid activation function, as $p_{ptr_t} = \sigma(v^{\top}_{ptr}[s_t, E(y_{t-1}), c_t] + b_{ptr})$, where $v_{ptr} \in R^{d_{dec} + d_{emb} + d_{doc}}$ and b_{ptr} are vectors, all of which are trained together with the rest of the network.

Experiments

We performed the following experiments :

1. Implemented [paper 1] : attention model with pointer generator networks on CNN Daily Mail dataset and calculated the relevant scores.
2. Implemented [paper 6] : QFS that builds on top of the model we coded earlier. We trained this on CNN Daily Mail (subset of the data) as well.
3. We qualitatively compared the summaries formed by the generic summariser and the QF summariser to analyse how the query changes the summary content.
4. We qualitatively compared the summaries formed by the QF summariser for the same article on two different queries to note the change in content and to understand what is causing the difference.

Results and Error Analysis

For the experiments we performed, we observed the following results :

Generic summarisation : The [Get to the Point] paper yields quite accurate and precise summaries for the CNN dataset. However, in some cases, the model fails to capture the essence of the article. We can see that the model has learned to give attention to the names "gabriel salvadore", "craig" and "manny pacquito". But, since the model is a seqToSeq model, it highly relies on the sequence of the input for generating summaries. In the article, the whole "craig walking into the restaurant" event happens first, after which "salvador, the lucky waiter lands a meeting with tv exec". In the reference summary, these events are inverted, hence the model performs poorly.

Reference summary *: gabriel salvador set up an initial meeting between a tv exec and manny pacquiao 's trainer . salvador is an actor and waiter at craig 's restaurant in los angeles .*

Pointer Generator Summary without coverage *: floyd mayweather vs. manny pacquiao in the fight in west hollywood last year . the president and ceo of cbs corp. includes showtime , moonves runs a powerful television network in a town where nearly every waiter also wants to be an actor*

Pointer Generator Summary with coverage *: les moonves sits down at a restaurant in hollywood , it 's usually the waiter 's lucky night . but when he walked into craig 's in west hollywood last year , those roles were reversed . this time , moonves ' waiter , gabriel salvador , was the one serving up a mouthwatering opportunity .*

We can see examples of very well formed summaries as well - although most well formed summaries are more extractive than abstractive. This is highly due to the fact that in news articles, the first few sentences is often a good elaboration of the highlight. Although, we can see the absence of any unresolved anaphora, given that the article used pronouns in most places. The model is successfully able to learn the relevant content out of the article, and include the important named entities involved (and include them using the pointer network"). Since the word "prosecutor" was not present in the article, the model was not able to generate this word on the basis of the context provided. The summaries are decently abstractive, although with abstraction, we see some factual details being misinterpreted due to replacement of complex words with in-vocabulary words (since the generated word's probability increases).

Reference Summary *: prosecutor : carlos colina , 32 , will be arraigned on the murder charge next week . he 's already been arraigned for alleged assault and battery , improper disposal of a body .body parts were found in a duffel bag and a common area of an apartment building .*

Pointer Generator Summary without coverage *:carlos colina , 32 , will be arraigned april 14 for murder in connection with the remains of cambridge . colina was arraigned on charges of assault and battery causing serious bodily injury and improper disposal of a body .*

Pointer Generator Summary with coverage : *carlos colina , 32 , will be arraigned the morning of april 14 for murder .colina was arraigned on charges of assault and battery causing serious bodily injury and improper disposal of a body .*

Query Focused Summarisation : The architecture we have used is able to give attention to the queries asked. Since the queries are 1-2 keywords, we can almost always see the summaries including sentences which contain these keywords. But they are not often capturing the gist of the article, they are just producing coherent sentences which contain the given query. Also, we observed that if the query entity occurs somewhere in the beginning of the article, it's summaries are captured better. An example query and summary is given below.

Document (cnn) – *the united states have named former germany captain jurgen klinsmann as their new national coach , just a day after sacking bob bradley . bradley , who took over as coach in january 2007 , was relieved of his duties on thursday , and u.s. soccer federation president sunil gulati confirmed in a statement on friday that his replacement has already been appointed . [...]*

Query *united states*

Reference summary *: jurgen klinsmann is named as coach of the united states national side*

Output summary : *klinsmann appointed as the new coach of united states*

Another observation is that since we give the query as an input into the decoder, the model "hopes to generate sentences that lead up to the entity" - as mentioned by the author. In the example below, there are two queries on the same article : Helmand and Afghanistan, both names of locations. So the model is learning to make sentences that are likely to include a location, but as we can see, they fail to capture the relevance of these queries in the articles (they carry the overall topic of the article - for example, the article used below was about rising military deaths).

We also see that the generated summaries mostly match the topic of the input documents, but they rarely succeed in generating summaries rephrasing something actually stated in the article. It might even present factual information incorrectly.

Reference Summary : *drive in helmand is part of effort to secure country before august elections*

Generated summary for query "helmand" *: new : death toll rises in helmand province*

Reference Summary : *uk military deaths in afghanistan now at 184 , five more than in iraq*

Generated summary for query “afghanistan” *: new : death toll rises in afghanistan*

In most other cases though, when the nature of the queries is different (both are not locations/ both are not names), we can notice that the queries are affecting the content of the summaries in a positive manner. For the same article and the same reference summary, if one entity is a name and the other is a location, one summary focuses on Hatoyama’s views (even though it’s not there in the reference summary) and the other on his visit to the given location

Reference *: hatoyama and his wife visited the nanjing memorial for victims of 1937 massacre*

Hatoyama : *new : president yukio hatoyama says he 's not to be a `` good ''*

Nanjing *: it was the first to visit to the nanjing*

We can frequently see repetitions of the same phrases, especially those including the query entities, and an extreme example is shown below. The model appears to get stuck trying to begin a summary. Additionally, we observe that the repetition can be observed in the attention distribution as well. The same problem has been seen by Nallapati et al. (2016), who make an addition, *temporal attention* Sankaran et al. (2016), to their model for alleviating the issue of repetitions. The example below shows that the model gets stuck while generating the summary for query “tsunami”, and has issues with starting the summary.

Generated Summary *: the earthquake tsunami tsunami tsunami tsunami tsunami*

Conclusion

Abstractive summaries are still lack a lot when pitched against extractive ones and even farther off when compared against manually generated ones. One would expect abstractive summarization models to actually be able to capture the context and generate words and phrases on their own. Even though, adding the concept of attention does improve results, it still remains very restrictive and focuses on words at the beginning of the document(gives preference to those words). Repetitions occur very often because the attention for words repeated are high enough. Temporal attention was proposed as a solution for the same, and while it does help in reducing the repetitions, it still does not help in generating better quality summaries. Sentence formation is very poor in most pointer-generator based models. Similarities in syntax can often be directly seen from the training set, but the notion of capturing information ios missing even after specifying the queries.