Survey on KG assisted Text Summarization

Sairam Mogili, Pushpak Bhattacharyya

{sairam, pb}@cse.iitb.ac.in

CFILT, Indian Institute of Technology Bombay, India

August 7, 2021

Abstract

Text summarization is a task of summarizing the documents, which is one of the difficult problems in NLP. There is no precise metric evaluating the summary which tells whether the summary is good or not. There are different machine learning approaches that label sentences for selecting the summary sentences. Also, there are different deep sequence-to-sequence neural networks models to generate summaries. Knowledge Graphs are used to generate a good quality summary. Nowadays, various deep neural network-based approaches are getting proposed for generating extractive as well as abstractive summaries. This report will give a brief idea about types of summary, summary evaluation measures and various ways to get summary.

1 Introduction

With increase in the available textual information day by day it is helpful to get only the important information from the document called summary. For humans to go through long documents will take more time. Text summarization is used to give the condensed version of the documents with all the important information and without any redundancy.

Text Summarization is compressing the input text into shorter version, with useful information inplace and removing the redundant information. In general the ideal compression of good summary is one third of given input text. There are different summarization techniques used for different usecases. There are different summary evaluation metrics, the details are given in following sections.

1.1 Types of Summarization[4]

Broadly summarization approaches are categorized as **abstractive** and extractive. In an **extractive** type of summarization sentences from the input, texts are presented as it is as part of summary whereas in case abstractive summarization new sentences depicting gist of a topic are formed. Summarization approaches based on the number of documents are classified as a **single document** and **multi-document**. When only one document is used to generate a condensed form of text then it is termed as single document summary and when more that one documents are searched for desired information then it is termed as multi-document summarization.

Sr. No	Type of Summary	Factors
1	Single and Multi-document	Number of Document
2	Extractive and Abstractive	Output(if exact or abstract is required)
3	Generic and Query-focused	Purpose (whether general or query related data is required)
4	Supervised and Unsupervised	Availability of training data
5	Mono, Multi and Cross-lingual	Language
6	Personalized	Information specific to user's need
7	Sentiment-based	Opinions are detected
8	Update	Current Updates regarding topic
9	E-mail bases	For summarizing e-mails
10	web-based	For summarizing web pages

Table 1: Various Types of Summarization Techniques[4]

Purpose of summary leads to **generic** and **query-focused** summarization. In a generic type of summarization entire document(s) is searched for various information contents, unlike query-focused where the document(s) are searched for only the topic mentioned in the query. The task of summarization can be applied to and sentiment from the document and such type of summarization is called as **sentiment-base** summarization. In the **update** type of summary, it is assumed that the user is aware of basic information related to the topic and only need to know recent updates regarding the topic.

If generated summary language is same as input document(s) then it is called as **mono-lingual** summarization and when the language of summary varies with that of input document(s) summary then it is called as **multi-lingual**

summarization. Sometimes based on pro file of user nature of summarization gets varied such type of summarization is termed as **personalized** summarization. Apart from these, there are **web-based**, **e-mail based** type of summarization as shown in the table 1.

1.2 Summary Evaluation Techniques[4]

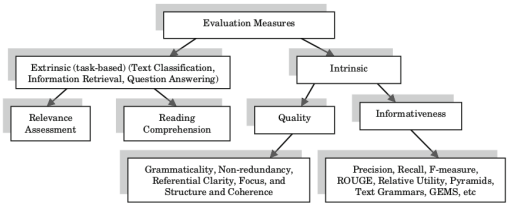

Figure 1: Summary Evaluation Techniques[4]

Automatic generation of the summary is a hard task since we don't know what part of the information should contribute to the summary. The varying perspective of summary makes it harder to evaluate automatically generated summary even from the trained human. Someone may see a certain point important while others may think that point less important. Purpose of summary can help to evaluate automatically generated summary. As described in the survey paper [4] evaluation of summary can be broadly categorized as follows,

1.2.1 Extrinsic Evaluation

There are various tasks that help to generate a summary of the text. In the extrinsic type of evaluation approach of summary gets tested for its usefulness to these various supporting tasks. Sometimes this type of evaluation is gets termed as **task-based** evaluation. Extrinsic evaluation is further categorised as follows

1. Relevance assessment: Generated summary is tested against relevance to the topic. This method is mostly used to topic/query-focused types of summarization.

2. Reading comprehension: It tests weather generated summary can be used to answer multiple choice tests.

1.2.2 Intrinsic Evaluation

Generally, reference summaries are used to evaluate generated summary mostly on the basis of informativeness and coverage. The relevance of summary to the input document(s) has an issue of finding a relevant topic from the document(s) as relevance has not a rigid definition. As shown in figure1 intrinsic evaluation is categorised as follows:

1. Quality evaluation: Quality of text in summary is checked on the basis of linguistic parameters like grammatically, structures and coherence, vocabulary, non-redundancy etc.

2. Informativeness evaluation: This is the most used type of summary evaluation techniques. There are two ways in which informativeness of summary is evaluated, they are as follows,

   Automatic: don't need human annotation

   Semi-automatic: needs human annotation

Following session explains some of the informativeness intrinsic evaluation techniques.

- ROUGE

  ROUGE (Recall-Oriented Understudy for Gisting Evaluation) makes use of reference summary for evaluation. It looks for co-occurrences of various levels grams in the generated summary and reference summary. Five different metrics are available to capture ROUGE.

  ROUGE-N: checks for overlap of N gram

  ROUGE-L: checks for longest common sub-sequences(LCS)

  ROUGE-W: weighted LCS, favours longest LCS

  ROUGE-S: skip-bigram based co-occurrence check

  ROUGE-SU: checks of co-occurrence except bi-gram and uni-gram.

- BLEU (Bilingual Evaluation Understudy)

  It is a modified form of precision. The modification comes from overlap between candidate summary and reference summary. Here overlap of words in summary is calculated with respect to the maximum count of that word from all reference summaries. It can be written in the equation as follows,

  $$P = m_{max}/w_t \tag{1}$$

  where m_{max} is maximum time occurrence of word from all reference summaries and w_t is total number of words present in generated summary.

- Basic Element(BE)
  Sentences are expressed in the form of using three word namely head, modifier/argument and relation(between head and modifier). Then these are mapped against various equivalence expressions.

- DEPEVAL
  This evaluation method is similar to BE method wherein parsers are used in this method unlike minipar in BE. Dependency triplets (head |modifier| relation) are from the automatically generated text are checked against the ones from reference summaries.

1.3 Outline

Rest of the document is organised as per chronological approaches applied and suggested by the community to provide a solution for Text Summarizing. section 2 suggests machine learning based approaches which are further categorised into sequence labelling task and statistical approaches. section 3 briefs about recent summarization approaches

2 ML based Summarization Approaches

Machine Learning based approaches need data to learn from. The summaries by human are given as reference summaries. Summarization can be considered as sequence labeling task where the sentences from source documents are given a binary label to indicate whether it is to be present in summary nor not for Extractive summarization. SVM[3] based or SVR[7] based ranking summarization approach are used for Multi-document summarization.

2.1 SVM based Ensemble approach to Multi-Document Summarization [3]

This[3] Abstractive multi-document summarization approach is used for the problem by DUC-2007 (Document Understanding Conference) which requires query based summarization for multi-document.

- Problem Definition: For a given question (description of topic) and a set of documents that are relevant, the task is to generate 250-word summary that answers the question.

- Data Labelling: To label automatically the DUC-2006 dataset is used. In each topic for sentences ROUGE score is calculated against the available reference summaries and top scored sentences are selected.

- Feature Extraction: Features related to query and some sentences features are extracted. some of the features are LCS, Weighted LCS, exact word overlap, skip-bigram, sentence length, sentence position, title match.

- SVM Ensemble: Even though with SVM good generalization is done some of the predictions which are false may degrade the performance, to overcome this ensemble is applied with majority voting. Leaving 25% of data 4 different models are trained on the remaining 75% of data. Sentences are ranked using the ensemble of these models and top N sentences are selected till the required summary length is satisfied.

- Results: Different ROUGE measures of baseline and SVM model are compared with this model in table 2.

Systems	R-1	R-L	R-W	R-SU
Precision	0.4081	0.3359	0.1791	0.1621
Recall	0.3705	0.3051	0.0877	0.1334
F-score	0.4081	0.3197	0.1177	0.1463

Table 2: ROUGE measures for SVM Ensemble [3]

2.2 Multi-document Summarization Using Support Vector Regression [7]

This approach[7] is for summarization of multi-document, which majorly contains three steps: Preprocessing the text, scoring the sentences, and post-processing.

- Text Processing: sentences are formed by segmenting the documents and query. From documents news heads are removed.

- Scoring: sentences are ranked using the features like word-based features, semantic based WordNet feature, sentence position, Named Entity Number feature etc. Combination of these features are used by the SVR to rank the sentences.

- Post-processing: sentences with high score is considered for summary. To make summary non-redundant simple rule based

method is used for removing phrases which are redundant.

DUC-2006 dataset is used for training. DUC-2006 contains 25 documents for each of the 50 topics and 4 reference for each topic. Authors proposed two different ways for scoring the sentence based on its similarity with the reference summary. For the given summary S and sentence s, the similarity is defined as eq 2.

$$Sim(s,S) = \frac{\sum_{t_i}\sum_{t_j} same(t_i,t_j)}{|s|} \quad (2)$$

Two sentence scoring strategies which uses the reference summaries given by humans are as follows

1. Average: sum of similarity of this sentence with reference summaries.

$$Score(s) = \sum_i sim(s,S_i) \quad (3)$$

2. Maximum: maximum similarity of the sentence with one of the 4 reference summaries.

$$Score(s) = \max_i \{sim(s,S_i)\} \quad (4)$$

After scoring the sentences, the formation of training data is done i.e D = $\{< V_s, Score(s) >\}$ and the regression function is learned by the SVR model. Performance of SVR model is given below compared to baseline and best submitted models. where baseline is SVR but its weights are assigned manually and best submitted is the one performed best in DUC-2006.

Systems	ROUGE-2
Best submitted system	0.09558
SVR-based system	0.09057
Baseline system	0.08012

Table 3: Performance of SVR-based system [7]

3 Neural Network based Summarization approaches

Lately Neural networks gained a lot of popularity, because of its ability to generalize for the given data. In the prev section we have seen the traditional ML approaches, in this section we will see different Neural network models for both extractive summarization models, SummaRuNNer [10] and abstractive summarization models, Pointer-Generator [11]. we will see the approaches and results in the following sections.

3.1 SummaRuNNer: RNN based Sequence Model for Extractive Summarization of Documents

SummaRuNNer is an extractive summarizer. It is a RNN based sequence classifier which makes a decision to select a sentence or not to include in summary. In this abstractive summaries are used for training. This model has two bi-directional GRU-RNN, architecture is present in the below figure. first layer takes the input and outputs the hidden state representations of words and these are concatenated and average-pooled output is passed as input to second layer, which outputs the sentence representations. These hidden states are then concatenated and average pooling of them and tanh (non-linear transformation) function is applied on them to get document representation as given at eq 5.

$$d = tanh(W_d \frac{1}{N_d}\sum_{j=1}^{N^d}[h_j^f, h_j^b] + b) \quad (5)$$

where h^f, h^b are the hidden states of jth sentence and N_d is the number of sentences

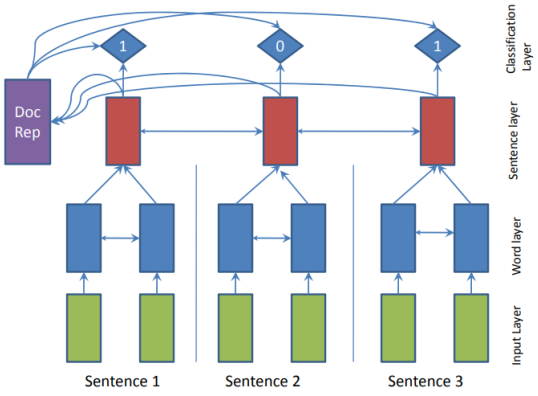

Figure 2: SummaRuNNer System Architecture

The task of deciding whether to add the sentence to summary or not i.e classification is done by logistic layer. This is done in the second pass, in which sequentially every sentence is visited. Paper [10] mentions the mathematical formulation of logistic layer as bellow

$$P(y_j = 1|h_j, s_j, d) = \sigma(W_c h \quad \#(content) + h_j^T W_s d \quad \#(salience) - h_j^T W_r tanh(s_j) \quad \#(novelty) + W_{ap} p_j^a \quad \#(abs.pos.imp.) + W_{rp} p_j^r \quad \#(rel.pos.imp.) + b) \quad \#(biasterm) \tag{6}$$

In the above equation the term $W_c h_j$ denotes content of the sentence, $h_j^T W_s d$ represent salience, $-h_j^T W_r tanh(s_j)$ is for redundancy, $p^a and p^r$ are absolute and relative positional embedding.

3.1.1 Extractive Training

To train a Extractive model gold labels should be present at sentence level for each document representing sentence included in summary. Labels are generated using the available abstractive summaries, using a greedy approach for increasing ROUGE score for selected sentences compared to gold summary. The approach intuition is that the selected sentences should given the high score. If adding the sentences doesn't increase the score this process is stopped and these set of sentences are given as true labels using which training is done.

We have added Triple features i.e frequency if triples is a sentence, which tells that there is triple and number of triples if there are multiple in the sentence while classifying. As we thought the results were increased i.e the ROUGE score value from 35 to 36.5, as shown in the table 4

3.1.2 Implementation and Result

Corpus used is CNN/DailyMail, It contains 286,722 training documents, 13,362 validation documents and 11,480 test documents and on average 28 sentences for document and 3-4 sentences of reference summary. The performance of this model compared with baseline (LEAD-3, first three sentences in doc) is given in table 4.

System	ROUGE-1	ROUGE-2	ROUGE-L
LEAD-3	39.2	15.7	35.5
SummaRuNNer	39.6 $\pm$0.2	16.2 $\pm$0.2	35.3 $\pm$0.2

Table 4: SummaRuNNer and Lead-3 Result Comparison on ROUGE score using Extractive Summary as Reference Summary on CNN/Daily Mail dataset

3.2 Summarization with Pointer-Generator Network[11]

Abstractive summarization proposed in the paper [11] tries to overcome the shortcomings of handling the OOVs. The author discusses three approaches (1)Baseline model (subsection 3.2.1) (2)Pointer generator model (subsection 3.2.2) and (3)Coverage mechanism (subsection 3.2.3).

3.2.1 Sequence-to-Sequence Attention Model

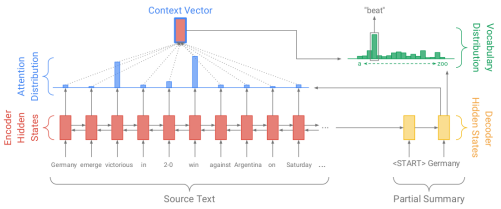

Figure 3: Baseline Sequence-to-Sequence Attention Model for Abstractive Text Summarization [11]

The paper [11] discussed baseline model which similar to the one used in this model. Base line model uses one Bi-directional as encoder and one Uni-directional LSTM as decoder The baseline model is shown in figure 3 from which it gets clear that the word *beat* gets generate based on present context of sentence. Let encoder hidden states be h_i and decoder hidden states be s_i then attention distribution at time-step t can be formulated as shown in equation 7 and 8 where v, W_h, W_s and b_{atten} are learnable parameters.

$$e_i^t = v^T tanh(W_h h_i + W_s s_t + b_{attn}) \tag{7}$$

$$a^t = softmax(e_t) \tag{8}$$

Attention mechanism is used which calculates the weighted average sum of hidden states and gives overall hidden state h^*. This hidden state along with hidden state of decoder then used to probability distribution P_{vocab} over all words in vocabulary. Probability Distribution over all vocab words is calculated as per the equation 9

$$P_{vocab} = softmax(V^{'}(V[s_t, h_t^*] + b) + b^{'}) \tag{9}$$

where, $V, V^{'}, b, b^{'}$ are learnable parameters. Negative log-likelihood is used to train and learn the parameters.

3.2.2 Pointer-Generator Network

This is hybrid model which combines the baseline model and the model of pointer network proposed in[12]. Pointer generation model tries to handle OOVs either by copying from input text or by generating from decoder vocabulary. Figure 4 describes use of working of pointer-generator model. Generation probability is calculated as shown in equation 10 where w's and b_{ptr} are learnable parameters.

$$p_{gen} = \sigma(w_h^T h_t^* + w_s^T s_t + w_x^T x_t + b_{ptr}) \quad (10)$$

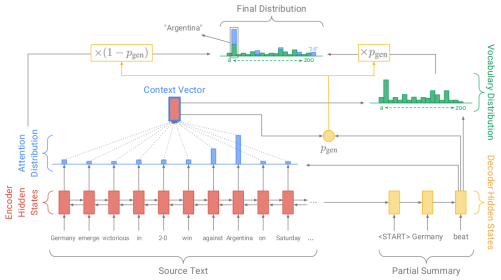

Figure 4: Pointer-Generator Model for Abstractive Text Summarization [11]

There is a *soft switch* and σ function is used to decide between generation and pointer mechanism. The notion of extended vocabulary which is a combination of vocabulary and all words appearing in the input text. The equation 11 give probability distribution of vocabulary words.

$$P(w) = p_{gen} P_{vocab}(w) + (1 - p_{gen}) \sum_{i:w_i=w} a_i^t \quad (11)$$

When w happens to be OOV, P_{vocab} becomes zero and in case of non-appearance in source document, attention term becomes zero. negative log-likelihood is used as loss function to train the model and learn the parameters.

3.2.3 Coverage Mechanism

The main purpose of coverage mechanism is to avoid repetition in the generated summary. To achieve this, paper [11] suggest maintaining coverage vector c^t which is attention distribution over all previous decoder time-steps. The equation 12

$$c^t = \sum_{t'=0}^{t-1} a_t \quad (12)$$

Updated equation form of the equation 13 after considering coverage vector is as shown bellow,

$$e_i^t = v^T tanh(W_h h_i + W_s s_t + w_c c_i^t + b_{attn}) \quad (13)$$

The author also suggests to add coverage loss to negative log likelihood, then equation 14 describe overall loss for learning parameters, where λ also gets learnt.

$$loss_t = -log\, P(w_t^*) + \lambda \sum_i min(a_i^t, c_i^t) \quad (14)$$

min of attention and coverage is useful for penalizing only overlapping part.

For some of the LG inputs the decoded summary contains the repeated segments. TO overcome this problem we incorporated the no repeated n-grams mechanism, this discards the hypothesis which has repeated n-grams and discards that hypothesis during Beam Search decoding.

We have applied Coreference resolution before passing to the model but the inputs contain unusually more number of mentions and replacing all the mentions with entity name has made the input contain lot of repeated words. This degraded the output performance so we discarded the step of applying coreference resolution.

3.2.4 Implementation and Result

The authors compare their model with abstractive model presented in subsection **??** and then the combination of sequence-to-sequence(s2s) with baseline by training on 150k vocabulary words and 50k vocabulary words. Table 7 shows results of the evaluation on the basis of ROUGE measure on CNN/Daily Mail training dataset where the proposed model makes use of 256-dimensional hidden states and 128-dimensional word embedding.

Model	Rouge-1	Rouge-2	Rouge-L
Abstractive Model (Nallapati et al., 2016)	35.46	13.3	32.65
s2s 150k vocab	30.49	11.17	28.08
s2s 50k vocab	31.33	11.81	28.83
Pointer Generator	36.44	15.66	33.42
Pointer Generator + Coverage	39.53	17.28	**36.38**

Table 5: Comparison of Results of Models Suggested in Paper[11] with Basic Sequence-to-Sequence Model Proposed in Paper [1]

4 DNN based Approach: BART

BART is a denoising autoencoder for pretraining sequence-to-sequence models. In the paper [6] it is described as , BART is trained by (1) corrupting text with an arbitrary noising function, and (2) learning a model to reconstruct the original text. Beacause of this, BART performs well on different tasks like Abstractive summarization, question answering etc. For Summarization, it gains of up to 6 ROUGE score.

4.1 Masked Language Modeling

MLM models such as BERT are pre-trained to predict masked tokens. BERT does two following steps. Random subset of input is replaced with a mask token [MASK]. (Adding corruption/noise). It predicts the original tokens for each of the [MASK] tokens.
BERT can see the whole input at once (some tokens replaces with MASK), it can see the input in forward and backward direction and therefore can get the neighbouring tokens of masked tokens.

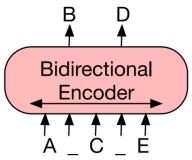

Figure 5: Encoding Part of BART[6]

It is suited for classification tasks but less suited for generation (text) tasks, where the words need to be generated seeing the previous available/generated words.

4.2 Autoregressive Models

Abstractive summarization is a text generation task, where tokens are predicted given the previous words. GPT-2 models are pre-trained to predict the next token.

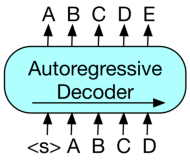

Figure 6: Decoding part of BART[6]

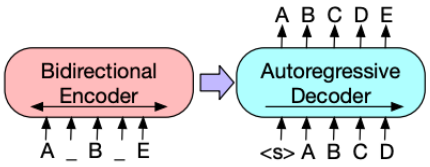

Figure 7: BART Architecture[6]

4.3 BART Sequence-to-Sequence

BART brings the best of both worlds, it has a encoder and decoder, like BERT as a encoder and GPT as a decoder. In BART model at decoder step, Beam search is used and even though we are generating summary the outputs doesn't look like Abstractive but look like Extractive summaries. To overcome this sampling methods were introduced to get abstractive outputs. The following figure depicts the probability of next word generation for whole sentence of beam search decoding. To overcome that sampling methods like Top-P and Top-K used to make decoding outputs human like.

4.4 Implementation and Results

Pre-trained a huge model with 12 layers in each of the encoder and decoder, and a 1024 hidden size. Following RoBERTa[8] we use a batch size of 8000, and train the model for 500000 steps. To help the model better fit the data, we disabled dropout for the final 10steps. The same pretraining data as [8], consisting of 160Gb of news, books, stories, and web text. The following table gives the results of this model on CNN datasets.

Model	R1	R2	RL
Lead-3	40.42	17.62	36.67
PTGEN+COV	39.53	17.28	36.38
BART	44.16	21.28	40.90

Table 6: BART resutls comapred to other models

5 KG Augmented Summarization

This model[5] uses two encoders one is sequential document encoder and other is a graph structured encoder. The second encoder is used to maintain the global context and first encoder to give the local characteristics of the entities. Another downstream task is modeled i.e a reward based on a multiple choice test to capture the entity interactions better.

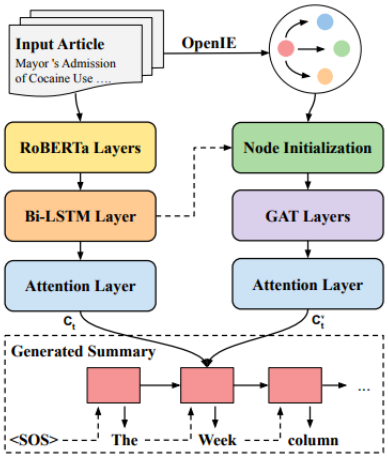

Figure 8: KG augmented summarization [5]

5.1 KG creation

For the given input document a knowledge graph is constructed. To construct the knowledge graph Stanford CoreNLP[9] is used to apply Coreference Resolution first and then OpenIE to extract triples. Linking accross different documents is not done but in the document linking is performed. From the generated triples they are discarded if the argument (subject or object) is more than threshold (10 words). Largest triple is considered if there are overlapping triples.

5.2 Encodering and Decoding

A graph is constructed with directed edges from predicate to object and subject to predicate. Reverse edges and self-loops are added to enhance the flow of information.

Node Initialization: The extracted triples contains multi word subject and objects. Thus node initilization is done by taking embedding average of the tokens. It leverages the document encoder hidden states as the contextual representation of tokens. Number of mentions in the node is added as an extra encoding to V, to signify entity salience.

Contextualized Node Encoding: Each node vi is represented by a weighted average of its neighbors:

$$\hat{\mathbf{v}}_i = \mathbf{v}_i + \|_{n=1}^{N} \sum_{v_j \in \mathcal{N}(v_i)} \alpha_{i,j}^{n} \mathbf{W}_{0,n} \mathbf{v}_j$$
$$\alpha_{i,j}^{n} = \text{softmax}\left((\mathbf{W}_{1,n}\mathbf{v}_i)^T (\mathbf{W}_{2,n}\mathbf{v}_j) \right) \quad (15)$$

N = 4 in used in experiments with two layers of GATs. N(vi) denotes the neighbors of vi in graph G. W∗ are trainable parameters.

The graph encoder described above encodes document-level global context by merging entity mentions throughout the document and capturing their interactions with graph paths. It is henceforth denoted as DOCGRAGH

5.3 DOC Encoder

First input is feed to RoBERTa[8] and token embeddings are taken from last layer output. These are passed to a single-layer bidirectional LSTM (BiLSTM) over token embeddings, producing encoder hidden states.

5.4 Summary Decoder

Output decoder consists of unidirectional sing-layer LSTM. It attends to both input doc and graph and generated the summary tokens recurrently.

Attending the Graph: Context vector to graph is computed at decoding step t, with attention mechanism (Bahdanau et al., 2014):

$$\mathbf{c}_t^v = \sum_i a_{i,t}^v \hat{\mathbf{v}}_i$$
$$a_{i,t}^v = \text{softmax}\left(\mathbf{u}_0^T \tanh\left(\mathbf{W}_3 \mathbf{s}_t + \mathbf{W}_4 \hat{\mathbf{v}}_i \right) \right) \quad (16)$$

Attending the Document: Similarly, the document context vector is computed over input tokens by additionally considering the graph context vector.

$$\mathbf{c}_t = \sum_k a_{k,t} \mathbf{h}_k$$
$$a_{k,t} = \text{softmax}(\mathbf{u}_1^T \tanh\left(\mathbf{W}_5 \mathbf{s}_t + \mathbf{W}_6 \mathbf{h}_k + \mathbf{W}_7 \mathbf{c}_t^v \right)) \quad (17)$$

Both the Doc and Graph context vectors, are taken from both sources and are concatenated with hidden state of decoder to produce the Vocab distribution. Copy mechanism proposed by (see et.al) is used to copy words from input instead of OOV tokens to the output summary

5.5 Reinforcement Learning with Cloze

To get more informative summaries, in the second stage Reinforcement Learning with self-critical policy gradient algorithm is used. While training two summaries will be generated: first is by sampling tokens based on distribution P(y|X) at decoding step: and other is greedy approach which selects the high probability token at each step.

$$\mathcal{L}_{\mathrm{rl}} = -\frac{1}{|D|} \sum_{(\mathbf{y}^s, \mathbf{x}) \in D} \left(R\left(\mathbf{y}^s\right) - R(\hat{\mathbf{y}})\right) \log p\left(\mathbf{y}\right) \tag{18}$$

where y_s is sampled summary and y is greedily generated summary

The reward function uses the combination of ROUGE and the multiple choice cloze score introduced below, i.e., R(y) = Rrouge(y) + clozeRcloze. For ROUGE, it considers weighted F1 scores of different rouge scores like ROUGE-1, ROUGE-2, and ROUGE-L calculated against the reference summary.

Multiple choice Cloze question construction

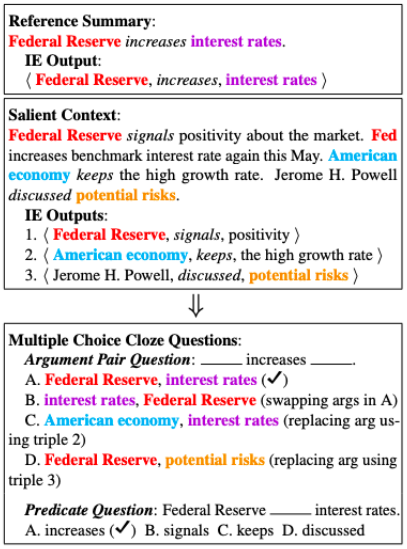

Figure 9: MCQs construction [5]

Question Construction. OpenIE[2] is used on the gold summaires available and taking only triples with arguments less than 5 words. For each triple of subject, predicate, objecti, two types of questions are created: (1) argument pair questions, by removing the subject and object, and (2) predicate questions, by removing the predicate. Candidate Answer Construction. Because fill-in the-blank style cloze may incorrectly penalize QA systems with answers paraphrased from the ground truth, Three candidates are constructed from the sentences which are summary-worth sentences from input.

5.6 Implementation and Results

To train our cloze QA model for CNN/DM,1,361,175 question-answer samples are collected from the training set. 20, 000 samples as a validation set and 20,000 samples as a test set.
RoBERTa model is used to extract token features for all experiments. Input articles to 512 (CNN/DM) BPEs. In this LSTM models with 256-dim and hidden states for the document encoder (128 each direction) and the decoder. For the residual connection of the graph encoder, 4 heads are used, each with a dimension of 72. α_1 = 0.33, α_2 = 0.33 on CNN/DM after tuning on the validation set. For both datasets, α_{cloze} = 0.05

Model	Rouge-1	Rouge-2	Rouge-L
PoiGen + Cov	0.3953	0.1728	0.3638
BART	**0.4416**	**0.2128**	**0.4090**
This model	0.4393	0.2037	0.4048

Table 7: Comparison of this model with SOTA models

This models is reliable because it makes use of the triples and the output summaries are more informative according to the human evaluation. Even though the score are not better than the BART model this models outputs are rated better by the human evaluators.

6 Conclusion

In this survey we have categorized ways of summarization as traditional approaches, machine learning based approaches and recent approaches which uses notion of deep neural network for generating summary. We have also described various of types of summarization like abstractive-extractive, multi-lingual, monolingual, supervised-unsupervised etc. Some of summary evaluation measures like ROUGE, BLEU, DEPVAL etc. are also described. Recently, due to advances in computational power, sophisticated models based on neural networks, joint learning, reinforcement learning etc. are getting proposed and year by year more accurate and acceptable summaries are getting produced. We can conclude that Text Summarization is vastly studied topic in the field of AI-NLP and research is still going on to achieve human-level excellence for producing summaries. As there is not exact measure to declare a summary as good or bad and as the readers perception changes as per domain knowledge, topic of text summarization remains open for researchers.

References

[1] Abstractive text summarization using sequence-to-sequence rnns and beyond.

arXiv preprint arXiv:1602.06023, 2016.

[2] Gabor Angeli, Melvin Jose Johnson Premkumar, and Christopher D Manning. Leveraging linguistic structure for open domain information extraction. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 344–354, 2015.

[3] Yllias Chali, Sadid A Hasan, and Shafiq R Joty. A svm-based ensemble approach to multi-document summarization. In *Canadian Conference on Artificial Intelligence*, pages 199–202. Springer, 2009.

[4] Mahak Gambhir and Vishal Gupta. Recent automatic text summarization techniques: a survey. *Artificial Intelligence Review*, 47(1):1–66, 2017.

[5] Luyang Huang, Lingfei Wu, and Lu Wang. Knowledge graph-augmented abstractive summarization with semantic-driven cloze reward. *arXiv preprint arXiv:2005.01159*, 2020.

[6] Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Ves Stoyanov, and Luke Zettlemoyer. Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. *arXiv preprint arXiv:1910.13461*, 2019.

[7] Sujian Li, You Ouyang, Wei Wang, and Bin Sun. Multi-document summarization using support vector regression. In *Proceedings of DUC*. Citeseer, 2007.

[8] Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*, 2019.

[9] Christopher D Manning, Mihai Surdeanu, John Bauer, Jenny Rose Finkel, Steven Bethard, and David McClosky. The stanford corenlp natural language processing toolkit. In *Proceedings of 52nd annual meeting of the association for computational linguistics: system demonstrations*, pages 55–60, 2014.

[10] Ramesh Nallapati, Feifei Zhai, and Bowen Zhou. Summarunner: A recurrent neural network based sequence model for extractive summarization of documents. In *Thirty-First AAAI Conference on Artificial Intelligence*, 2017.

[11] Abigail See, Peter J Liu, and Christopher D Manning. Get to the point: Summarization with pointer-generator networks. *arXiv preprint arXiv:1704.04368*, 2017.

[12] Oriol Vinyals, Alexander Toshev, Samy Bengio, and Dumitru Erhan. Show and tell: A neural image caption generator. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3156–3164, 2015.