Results

For facebook/bart-base model:

```
ROUGE Scores:
rouge-1:
  r: 0.4634
  p: 0.5938
  f: 0.5205
rouge-2:
  r: 0.2273
  p: 0.2857
  f: 0.2532
rouge-l:
  r: 0.2927
  p: 0.3750
  f: 0.3288
for facebook-bart model
BERT Precision: 0.8584
BERT Recall: 0.8532
BERT F1 Score: 0.8558
```

ROUGE Scores: The ROUGE-1 precision (0.5938) is higher than the recall (0.4634). ROUGE-2 scores are moderate, indicating some overlap for bigrams.

BERT Scores: High BERT precision (0.8584) and recall (0.8532), indicating the generated summaries are close in meaning to the reference texts.

For Google-T5 Model

```
ROUGE Scores:
rouge-1:
  r: 0.3659
  p: 0.6522
  f: 0.4687
rouge-2:
  r: 0.0909
  p: 0.1538
  f: 0.1143
rouge-l:
  r: 0.1951
  p: 0.3478
  f: 0.2500
for google-t5 model
BERT Precision: 0.8605
BERT Recall: 0.8468
BERT F1 Score: 0.8536
```

ROUGE Scores: The ROUGE-1 precision is high (0.6522), but the recall is a bit moderate (0.3659). ROUGE-2 scores are low, indicating less overlap for bigrams.
BERT Scores: High BERT precision (0.8605) and good recall (0.8468), results that the model generates summaries that are very close in meaning to the reference text.

For Bert2Bert model:

```
ROUGE Scores:
rouge-1:
  r: 1.0000
  p: 0.6721
  f: 0.8039
rouge-2:
  r: 1.0000
  p: 0.4681
  f: 0.6377
rouge-l:
  r: 1.0000
  p: 0.6721
  f: 0.8039
for bert2bert model
BERT Precision: 0.8077
BERT Recall: 0.9751
BERT F1 Score: 0.8836
```

ROUGE Scores: Perfect recall and high precision for ROUGE-1 and ROUGE-L, but lower precision for ROUGE-2.

BERT Scores: High precision (0.8077) and very high recall (0.9751), indicating excellent content overlap with reference summaries.

Evaluation Metrics

For topics like text summarization, this can evaluate the accuracy of generated text with reference text using ROUGE score, BERT score, etc.

1. ROUGE score

   Recall-Oriented Understudy for gist Evaluation (ROUGE). ROUGE-n compares n-grams of generated text with n-grams of reference text.

   What is N-Gram?

   It’s a chunk of n-words

   Ex: Deep Learning is awesome

   1-Gram: “Deep”, “Learning”, “is”, “awesome”.
   2-Gram: “Deep Learning”, “Learning is”, “is awesome”

   Let’s take an example on how to calculate the ROUGE score.

   Generated text: I really loved reading the hunger games

Reference text: I loved reading the hunger games

Unigram (1-gram) for generated text: “ I “,”really” ,”loved”,” reading”,” the”,” hunger”,” games”

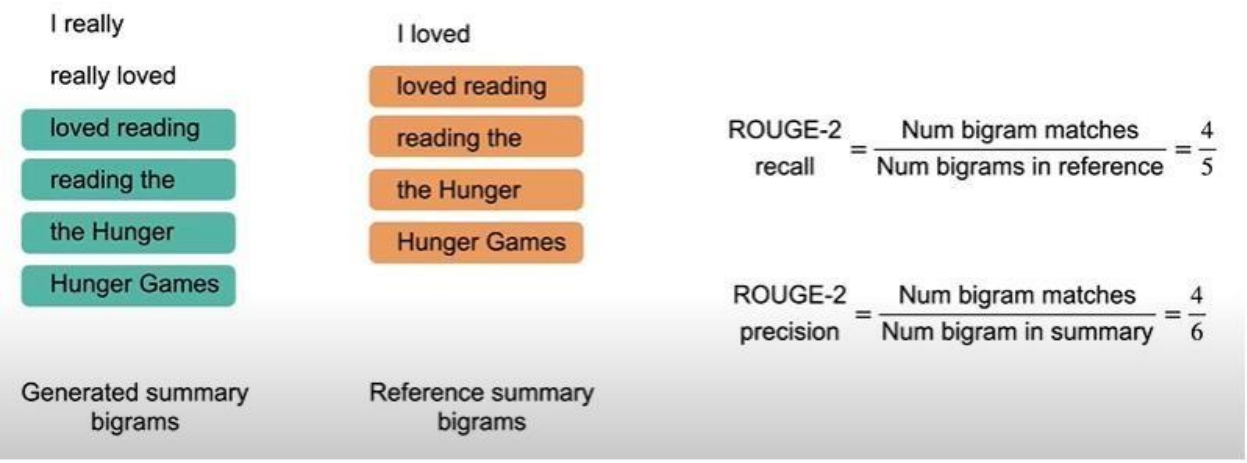
Unigram (1-gram) for reference text: “I”,”loved”,”reading”,”the”,”hunger”,”games”

$$\text{ROUGE-1 recall} = \frac{\text{Num word matches}}{\text{Num words in reference}} = \frac{6}{6}$$

$$\text{ROUGE-1 precision} = \frac{\text{Num word matches}}{\text{Num words in summary}} = \frac{6}{7}$$

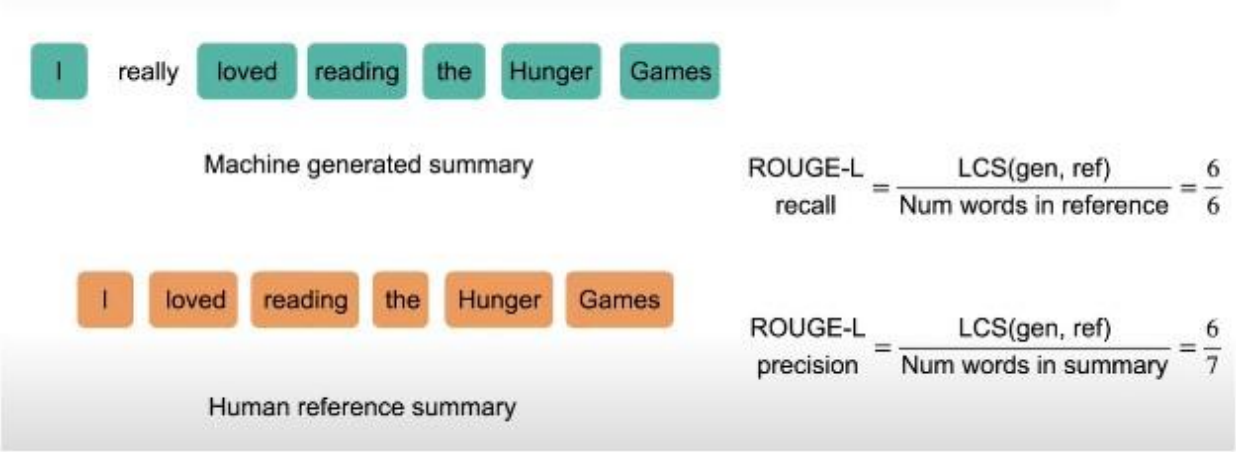
$$\text{ROUGE-1 F1-score} = 2\left(\frac{\text{precision} \cdot \text{recall}}{\text{precision} + \text{recall}}\right) = 0.92$$

Similarly, calculate ROUGE-2



And for ROUGE-L:

“L” stands for longest subsequence.



The main advantage of ROUGE-L over ROUGE-1 and ROUGE-2 is that it doesn't depend on the consecutive matches of n-gram, which means it tends to capture the sentence structure more accurately.

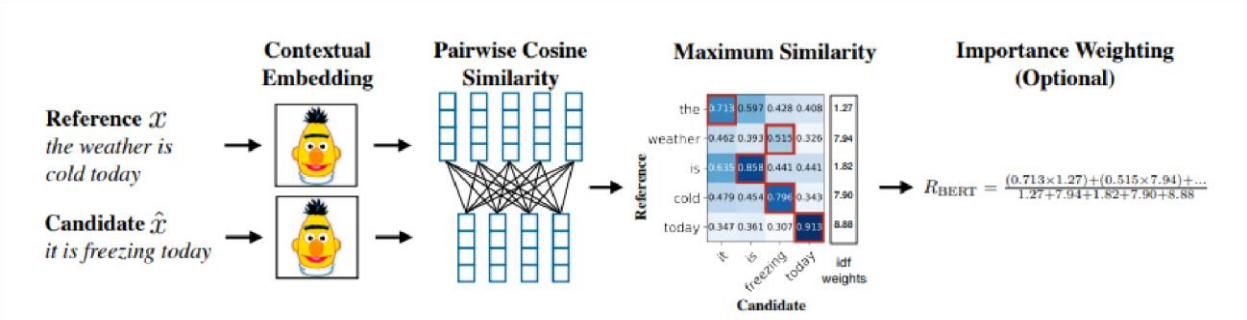
2. BERT score

Bi-directional Encoder Representation from Transformers is a context based model.

Step 1: Contextual Embedding: Models such as BERT, Roberta, XLNET, and XLM create contextual embedding based on surrounding words to represent reference and candidate phrases.

Step 2: Cosine Similarity: Cosine similarity is used to quantify how similar the contextual embedding of the reference and candidate phrases are to one other.

Step3: Token matching for precision and recall is done in step three. Recall and precision are computed by matching each token in the candidate sentence to the most comparable token in the reference sentence and vice versa. The results are pooled to determine the F1 score.



$$R_{\text{BERT}} = \frac{1}{|x|}\sum_{x_i \in x} \max_{\hat{x}_j \in \hat{x}} \mathbf{x}_i^\top \hat{\mathbf{x}}_j \quad , \quad P_{\text{BERT}} = \frac{1}{|\hat{x}|}\sum_{\hat{x}_j \in \hat{x}} \max_{x_i \in x} \mathbf{x}_i^\top \hat{\mathbf{x}}_j \quad , \quad F_{\text{BERT}} = 2\frac{P_{\text{BERT}} \cdot R_{\text{BERT}}}{P_{\text{BERT}} + R_{\text{BERT}}}$$

Step 4: Importance Weighting: Inverse Document Frequency (IDF), which can be optionally and domain-dependently added to BERTScore equations, is used to weigh the importance of rare words.

Example:

Reference Text: "the weather is cold today"

Candidate Text: "it is freezing today"

step1:Contextual Embedding

Every token in the two texts is transformed into high-dimensional vectors via BERT.

Step 2: In pairs Similarity of Cosines

Compute the reference and candidate text tokens' cosine similarity matrix.

Step 3: Maximum similarity

Determine the highest cosine similarity between any token in the candidate text and any token in the reference text for each token.

Let’s consider an example values (not exact calculated values)

Candidate \ Reference	the	weather	is	cold	today
it	0.713	0.369	0.428	0.404	0.347
is	0.360	0.369	0.515	0.441	0.316
freezing	0.428	0.515	0.342	0.433	0.347
today	0.360	0.316	0.428	0.404	0.878

Recall calculation: How well the reference text tokens are represented in the candidate text (generated text). For each token from reference text, find maximum similarity with all the tokens from the candidate text.

- the: max(0.713, 0.360, 0.428, 0.360) = 0.713
- weather: max(0.369, 0.369, 0.515, 0.316) = 0.515

- is: max(0.428, 0.515, 0.342, 0.428) = 0.515
- cold: max(0.404, 0.441, 0.433, 0.404) = 0.441
- today: max(0.347, 0.316, 0.347, 0.878) = 0.878

For simpler calculations, let's keep the weighted scores as optional.

Average of Maximum Similarities: (0.713+0.515+0.515+0.441+0.878)/5=0.6124

Precision calculation: How well the candidate text tokens are represented in the reference text

For each token from candidate text, find maximum similarity with all the tokens from reference text.

- it: max(0.713, 0.369, 0.428, 0.404, 0.347) = 0.713
- is: max(0.360, 0.369, 0.515, 0.441, 0.316) = 0.515
- freezing: max(0.428, 0.515, 0.342, 0.433, 0.347) = 0.515
- today: max(0.360, 0.316, 0.428, 0.404, 0.878) = 0.878

Average of Maximum Similarities: (0.713+0.515+0.515+0.878)/4=0.65525