


How to evaluate a summarization task



Shyamal Anadkat, Simón Fishman

Aug 15, 2023

 Open in Github

In this notebook we delve into the evaluation techniques for abstractive summarization tasks using a simple example. We explore traditional evaluation methods like **ROUGE** and **BERTScore**, in addition to showcasing a more novel approach using LLMs as evaluators.

Evaluating the quality of summaries is a time-consuming process, as it involves different quality metrics such as coherence, conciseness, readability and content. Traditional automatic evaluation metrics such as ROUGE and BERTScore and others are concrete and reliable, but they may not correlate well with the actual quality of summaries. They show relatively low correlation with human judgments, especially for open-ended generation tasks ([Liu et al., 2023](#)). There's a growing need to lean on human evaluations, user feedback, or model-based metrics while being vigilant about potential biases. While human judgment provides invaluable insights, it is often not scalable and can be cost-prohibitive.

In addition to these traditional metrics, we showcase a method (**G-Eval**) that leverages Large Language Models (LLMs) as a novel, reference-free metric for assessing abstractive summaries. In this case, we use `gpt-4` to score candidate outputs. `gpt-4` has effectively learned an internal model of language quality that allows it to differentiate between fluent, coherent text and low-quality text. Harnessing this internal scoring mechanism allows auto-evaluation of new candidate outputs generated by an LLM.

Setup

```
# Installing necessary packages for the evaluation
# rouge: For evaluating with ROUGE metric
# bert_score: For evaluating with BERTScore
# openai: To interact with OpenAI's API
!pip install rouge --quiet
```



```
/pip install bert_score --quiet  
/pip install openai --quiet
```

```
import openai  
import os  
import re  
import pandas as pd
```

```
# Python Implementation of the ROUGE Metric  
from rouge import Rouge
```

```
# BERTScore leverages the pre-trained contextual embeddings from BERT and matches words in candidate  
from bert_score import BERTScorer
```

```
openai.api_key = os.environ.get("OPENAI_API_KEY")
```

```
<IPython.core.display.Javascript object>
```

Example task

For the purposes of this notebook we'll use the example summarization below. Notice that we provide two generated summaries to compare, and a reference human-written summary, which evaluation metrics like ROUGE and BERTScore require.

Excerpt (excerpt):

"OpenAI's mission is to ensure that artificial general intelligence (AGI) benefits all of humanity. OpenAI will build safe and beneficial AGI directly, but will also consider its mission fulfilled if its work aids others to achieve this outcome. OpenAI follows several key principles for this purpose. First, broadly distributed benefits - any influence over AGI's deployment will be used for the benefit of all, and to avoid harmful uses or undue concentration of power. Second, long-term safety - OpenAI is committed to doing the research to make AGI safe, and to promote the adoption of such research across the AI community. Third, technical leadership - OpenAI aims to be at the forefront of AI capabilities. Fourth, a cooperative orientation - OpenAI actively cooperates with other research and policy institutions, and seeks to create a global community working together to address AGI's global challenges."

Summaries:

Reference Summary / ref_summary (human generated)	Eval Summary 1 / eval_summary_1 (system generated)	Eval Summary 2 / eval_summary_2 (system generated)
OpenAI aims to ensure artificial general intelligence (AGI) is used for everyone's benefit, avoiding harmful uses or undue power concentration. It is committed to researching AGI safety, promoting such studies among the AI community. OpenAI seeks to lead in AI capabilities and cooperates with global research and policy institutions to address AGI's challenges.	OpenAI aims to AGI benefits all humanity, avoiding harmful uses and power concentration. It pioneers research into safe and beneficial AGI and promotes adoption globally. OpenAI maintains technical leadership in AI while cooperating with global institutions to address AGI challenges. It seeks to lead a collaborative worldwide effort developing AGI for collective good.	OpenAI aims to ensure AGI is for everyone's use, totally avoiding harmful stuff or big power concentration. Committed to researching AGI's safe side, promoting these studies in AI folks. OpenAI wants to be top in AI things and works with worldwide research, policy groups to figure AGI's stuff.

Take a moment to figure out which summary you'd personally prefer and the one that captures OpenAI's mission really well.

```
excerpt = "OpenAI's mission is to ensure that artificial general intelligence (AGI) benefits all of humanity"
ref_summary = "OpenAI aims to ensure artificial general intelligence (AGI) is used for everyone's benefit, avoiding harmful uses or undue power concentration. It is committed to researching AGI safety, promoting such studies among the AI community. OpenAI seeks to lead in AI capabilities and cooperates with global research and policy institutions to address AGI's challenges."
eval_summary_1 = "OpenAI aims to AGI benefits all humanity, avoiding harmful uses and power concentration. It pioneers research into safe and beneficial AGI and promotes adoption globally. OpenAI maintains technical leadership in AI while cooperating with global institutions to address AGI challenges. It seeks to lead a collaborative worldwide effort developing AGI for collective good."
eval_summary_2 = "OpenAI aims to ensure AGI is for everyone's use, totally avoiding harmful stuff or big power concentration. Committed to researching AGI's safe side, promoting these studies in AI folks. OpenAI wants to be top in AI things and works with worldwide research, policy groups to figure AGI's stuff."
```

<IPython.core.display.Javascript object>

Evaluating using ROUGE

ROUGE, which stands for Recall-Oriented Understudy for Gisting Evaluation, primarily gauges the overlap of words between a generated output and a reference text. It's a prevalent metric for evaluating automatic summarization tasks. Among its variants, **ROUGE-L** offers insights into the longest contiguous match between system-generated and reference summaries, gauging how well the system retains the original summary's essence.



```
# function to calculate the Rouge score
def get_rouge_scores(text1, text2):
    rouge = Rouge()
    return rouge.get_scores(text1, text2)

rouge_scores_out = []

# Calculate the ROUGE scores for both summaries using reference
eval_1_rouge = get_rouge_scores(eval_summary_1, ref_summary)
eval_2_rouge = get_rouge_scores(eval_summary_2, ref_summary)

for metric in ["rouge-1", "rouge-2", "rouge-l"]:
    for label in ["F-Score"]:
        eval_1_score = eval_1_rouge[0][metric][label[0].lower()]
        eval_2_score = eval_2_rouge[0][metric][label[0].lower()]

        row = {
            "Metric": f"{metric} ({label})",
            "Summary 1": eval_1_score,
            "Summary 2": eval_2_score,
        }
        rouge_scores_out.append(row)

def highlight_max(s):
    is_max = s == s.max()
    return [
        "background-color: lightgreen" if v else "background-color: white"
        for v in is_max
    ]

rouge_scores_out = (
    pd.DataFrame(rouge_scores_out)
    .set_index("Metric")
    .style.apply(highlight_max, axis=1)
)

rouge_scores_out
```

	Summary 1	Summary 2
Metric		
rouge-1 (F-Score)	0.488889	0.511628
rouge-2 (F-Score)	0.230769	0.163265
rouge-l (F-Score)	0.488889	0.511628

```
<IPython.core.display.Javascript object>
```

The table shows the ROUGE scores for evaluating two different summaries against a reference text. In the case of rouge-1, Summary 2 outperforms Summary 1, indicating a better overlap of individual words and for rouge-l, Summary 2 has a higher score, implying a closer match in the longest common subsequences, and thus a potentially better overall summarization in capturing the main content and order of the original text. Since Summary 2 has many words and short phrases directly lifted from the excerpt, its overlap with the reference summary would likely be higher, leading to higher ROUGE scores.

While ROUGE and similar metrics, such as **BLEU** and **METEOR**, offer quantitative measures, they often fail to capture the true essence of a well-generated summary. They also correlate worse with human scores. Given the advancements in LLMs, which are adept at producing fluent and coherent summaries, traditional metrics like ROUGE may inadvertently penalize these models. This is especially true if the summaries are articulated differently but still encapsulate the core information accurately.

Evaluating using BERTScore

ROUGE relies on the exact presence of words in both the predicted and reference texts, failing to interpret the underlying semantics. This is where **BERTScore** comes in and leverages the contextual embeddings from the BERT model, aiming to evaluate the similarity between a predicted and a reference sentence in the context of machine-generated text. By comparing embeddings from both sentences, BERTScore captures semantic similarities that might be missed by traditional n-gram based metrics.

```
# Instantiate the BERTScorer object for English language
scorer = BERTScorer(lang="en")

# Calculate BERTScore for the summary 1 against the excerpt
# P1, R1, F1_1 represent Precision, Recall, and F1 Score respectively
P1, R1, F1_1 = scorer.score([eval_summary_1], [ref_summary])

# Calculate BERTScore for summary 2 against the excerpt
# P2, R2, F2_2 represent Precision, Recall, and F1 Score respectively
P2, R2, F2_2 = scorer.score([eval_summary_2], [ref_summary])
```

```
print("Summary 1 F1 Score:", F1_1.tolist()[0])
print("Summary 2 F1 Score:", F2_2.tolist()[0])
```

Some weights of the model checkpoint at roberta-large were not used when initializing RobertaModel
 - This IS expected if you are initializing RobertaModel from the checkpoint of a model trained on different data
 - This IS NOT expected if you are initializing RobertaModel from the checkpoint of a model trained on the same data

Summary 1 F1 Score: 0.9227314591407776
 Summary 2 F1 Score: 0.9189572930335999

<IPython.core.display.Javascript object>

The close F1 Scores between the summaries indicate that they may perform similarly in capturing the key information. However, this small difference should be interpreted with caution. Since `BERTScore` may not fully grasp subtleties and high-level concepts that a human evaluator might understand, reliance solely on this metric could lead to misinterpreting the actual quality and nuances of the summary. An integrated approach combining `BERTScore` with human judgment and other metrics could offer a more reliable evaluation.

Evaluating using GPT-4

Here we implement an example **reference-free** text evaluator using `gpt-4`, inspired by the [G-Eval](#) framework which evaluates the quality of generated text using large language models. Unlike metrics like `ROUGE` or `BERTScore` that rely on comparison to reference summaries, the `gpt-4` based evaluator assesses the quality of generated content based solely on the input prompt and text, without any ground truth references. This makes it applicable to new datasets and tasks where human references are sparse or unavailable.

Here's an overview of this method:

1. We define four distinct criteria:
 1. **Relevance:** Evaluates if the summary includes only important information and excludes redundancies.
 2. **Coherence:** Assesses the logical flow and organization of the summary.

3. **Consistency:** Checks if the summary aligns with the facts in the source document.
4. **Fluency:** Rates the grammar and readability of the summary.
2. We craft prompts for each of these criteria, taking the original document and the summary as inputs, and leveraging chain-of-thought generation and guiding the model to output a numeric score from 1-5 for each criteria.
3. We generate scores from `gpt-4` with the defined prompts, comparing them across summaries.

In this demonstration, we're using a direct scoring function where `gpt-4` generates a discrete score (1-5) for each metric. Normalizing the scores and taking a weighted sum could result in more robust, continuous scores that better reflect the quality and diversity of the summaries.



```
# Evaluation prompt template based on G-Eval
EVALUATION_PROMPT_TEMPLATE = """
You will be given one summary written for an article. Your task is to rate the summary on one metric
Please make sure you read and understand these instructions very carefully.
Please keep this document open while reviewing, and refer to it as needed.

Evaluation Criteria:

{criteria}

Evaluation Steps:

{steps}

Example:

Source Text:

{document}

Summary:

{summary}

Evaluation Form (scores ONLY):

- {metric_name}
"""

# Metric 1: Relevance

RELEVANCY_SCORE_CRITERIA = """
Relevance(1-5) - selection of important content from the source. \
The summary should include only important information from the source document. \
Annotators were instructed to penalize summaries which contained redundancies and excess information
"""

RELEVANCY_SCORE_STEPS = """
```

```

1. Read the summary and the source document carefully.
2. Compare the summary to the source document and identify the main points of the article.
3. Assess how well the summary covers the main points of the article, and how much irrelevant or redun
4. Assign a relevance score from 1 to 5.
"""

```

Metric 2: Coherence

```

COHERENCE_SCORE_CRITERIA = """
Coherence(1-5) - the collective quality of all sentences. \
We align this dimension with the DUC quality question of structure and coherence \
whereby "the summary should be well-structured and well-organized. \
The summary should not just be a heap of related information, but should build from sentence to a \
coherent body of information about a topic."
"""

```

```

COHERENCE_SCORE_STEPS = """
1. Read the article carefully and identify the main topic and key points.
2. Read the summary and compare it to the article. Check if the summary covers the main topic and key
and if it presents them in a clear and logical order.
3. Assign a score for coherence on a scale of 1 to 5, where 1 is the lowest and 5 is the highest base
"""

```

Metric 3: Consistency

```

CONSISTENCY_SCORE_CRITERIA = """
Consistency(1-5) - the factual alignment between the summary and the summarized source. \
A factually consistent summary contains only statements that are entailed by the source document. \
Annotators were also asked to penalize summaries that contained hallucinated facts.
"""

```

```

CONSISTENCY_SCORE_STEPS = """
1. Read the article carefully and identify the main facts and details it presents.
2. Read the summary and compare it to the article. Check if the summary contains any factual errors
3. Assign a score for consistency based on the Evaluation Criteria.
"""

```

Metric 4: Fluency

```

FLUENCY_SCORE_CRITERIA = """
Fluency(1-3): the quality of the summary in terms of grammar, spelling, punctuation, word choice, and
1: Poor. The summary has many errors that make it hard to understand or sound unnatural.
2: Fair. The summary has some errors that affect the clarity or smoothness of the text, but the main
3: Good. The summary has few or no errors and is easy to read and follow.
"""

```

```

FLUENCY_SCORE_STEPS = """
Read the summary and evaluate its fluency based on the given criteria. Assign a fluency score from 1
"""

```

```

def get_geval_score(
    criteria: str, steps: str, document: str, summary: str, metric_name: str
):
    prompt = EVALUATION_PROMPT_TEMPLATE.format(
        criteria=criteria,
        steps=steps,
        metric_name=metric_name,
        document=document,
        summary=summary,
    )

```



```

response = openai.ChatCompletion.create(
    model="gpt-4",
    messages=[{"role": "user", "content": prompt}],
    temperature=0,
    max_tokens=5,
    top_p=1,
    frequency_penalty=0,
    presence_penalty=0,
)
return response.choices[0].message.content

evaluation_metrics = {
    "Relevance": (RELEVANCY_SCORE_CRITERIA, RELEVANCY_SCORE_STEPS),
    "Coherence": (COHERENCE_SCORE_CRITERIA, COHERENCE_SCORE_STEPS),
    "Consistency": (CONSISTENCY_SCORE_CRITERIA, CONSISTENCY_SCORE_STEPS),
    "Fluency": (FLUENCY_SCORE_CRITERIA, FLUENCY_SCORE_STEPS),
}

summaries = {"Summary 1": eval_summary_1, "Summary 2": eval_summary_2}

data = {"Evaluation Type": [], "Summary Type": [], "Score": []}

for eval_type, (criteria, steps) in evaluation_metrics.items():
    for summ_type, summary in summaries.items():
        data["Evaluation Type"].append(eval_type)
        data["Summary Type"].append(summ_type)
        result = get_geval_score(criteria, steps, excerpt, summary, eval_type)
        score_num = int(result.strip())
        data["Score"].append(score_num)

pivot_df = pd.DataFrame(data, index=None).pivot(
    index="Evaluation Type", columns="Summary Type", values="Score"
)
styled_pivot_df = pivot_df.style.apply(highlight_max, axis=1)
display(styled_pivot_df)

```

Summary Type	Summary 1	Summary 2
Evaluation Type		
Coherence	5	3
Consistency	5	5
Fluency	3	2
Relevance	5	4

<IPython.core.display.Javascript object>

Overall, the Summary 1 appears to outperform Summary 2 in three of the four categories (Coherence, Relevance and Fluency). Both summaries are found to be consistent with each other. The result might suggest that Summary 1 is generally preferable based on the given evaluation criteria.

Limitations

Note that LLM-based metrics could have a bias towards preferring LLM-generated texts over human-written texts. Additionally LLM based metrics are sensitive to system messages/prompts. We recommend experimenting with other techniques that can help improve performance and/or get consistent scores, striking the right balance between high-quality expensive evaluation and automated evaluations. It is also worth noting that this scoring methodology is currently limited by `gpt-4` 's context window.

Conclusion

Evaluating abstractive summarization remains an open area for further improvement. Traditional metrics like `ROUGE` , `BLEU` , and `BERTScore` provide useful automatic evaluation but have limitations in capturing semantic similarity and nuanced aspects of summarization quality. Moreover, they require reference outputs which can be expensive to collect/label. LLM-based metrics offer promise as a reference-free method of evaluating coherence, fluency, and relevance. However, they too have potential biases favoring text generated by LLMs. Ultimately, a combination of automatic metrics and human evaluation is ideal for reliably assessing abstractive summarization systems. While human evaluation is indispensable for gaining a comprehensive understanding of summary quality, it should be complemented with automated evaluation to enable efficient, large-scale testing. The field will continue to evolve more robust evaluation techniques, balancing quality, scalability, and fairness. Advancing evaluation methods is crucial for driving progress in production applications.

References

- **G-EVAL: NLG Evaluation Using GPT-4 with Better Human Alignment** - Liu Y, Iter D, Xu Y, Wang S, Xu R, Zhu C. Published May, 2023.

- **BERTScore: Evaluating Text Generation with BERT** - Zhang T, Kishore V, Wu F, Weinberger KQ, Artzi Y. Published online February, 2020.
- **ROUGE: A Package for Automatic Evaluation of Summaries** - Lin CY. Published July, 2004.
- **SummEval: Re-evaluating Summarization Evaluation** - Fabbri et al. Published April, 2021.