Measuring AI Quality, Evaluation & Creation of Automatic Summaries

January 20, 2026

This project leverages GPT4 to build an AI model that generates automatic summaries. The model is trained on the classic CNN/DailyMail dataset and evaluated using automatic metrics (BLEU, ROUGE, METEOR, BERTScore) by comparing model outputs against human references.

Additionally, a POS tagging comparative analysis is also performed to assess linguistic quality.

1 Input and Cleaning

[ ]:
```
# Charging and Extracting of the dataset:
```

[ ]:
```
from datasets import load_dataset
dataset = load_dataset("ccdv/cnn_dailymail", "3.0.0")
```

[3]:
```
print(dataset["train"][0]["article"][:300])
print(dataset["train"][0]["highlights"])
```

```
It's official: U.S. President Barack Obama wants lawmakers to weigh in on
whether to use military force in Syria. Obama sent a letter to the heads of the
House and Senate on Saturday night, hours after announcing that he believes
military action against Syrian targets is the right step to take over
Syrian official: Obama climbed to the top of the tree, "doesn't know how to get
down"
Obama sends a letter to the heads of the House and Senate .
Obama to seek congressional approval on military action against Syria .
Aim is to determine whether CW were used, not by whom, says U.N. spokesman .
```

[ ]:
```

```

Since this dataset contains thousands of entries, we prepared a subset that only includes 5% of the data. This approach simplifies and accelerates model training while preserving diversity and ensuring that evaluation metrics remain reliable and free from noise.

[19]:
```
train = dataset["train"].shuffle(seed=42)

subset_size = int(0.05 * len(train))
train_subset = train.select(range(subset_size))
```

```
print("The lenght of the subset: ")
print(len(train_subset))  # debería ser ~14,000 ejemplos

print("Original entry number 300: ")
print(train_subset[0]["article"][:300])

print("Highlights of the article:"  )
print(train_subset[0]["highlights"])
```

```
The lenght of the subset:
14355
Original entry number 300:
A protester in Ferguson was arrested during a demonstration on Thursday night -
and live-tweeted her entire experience. Brittany Ferrell, a nursing student at
the University of Missouri-Saint Louis, was one of 13 people detained by
officers in the conflicted Missouri city for 'noise disruption'. The
Highlights of the article:
Brittany Ferrell, nursing student, was arrested with 12 people on Thursday .
They were calling on police take responsibility for Michael Brown's death .
Ms Ferrell tweeted as she was arrested, piled in a small wagon with 7 others .
They were accused of 'noise disruption', put in orange jumpsuits and cuffed .
Officers now being investigated, lawyers claim they 'overstretched powers'
```

[34]:
```
# Saving the subset to access anytime later to it without the necessity of␣
↪downloading and extracting all the dataset again.

train_subset.save_to_disk("train_subset_5pct")
```

```
Saving the dataset (0/1 shards):   0%|          | 0/14355 [00:00<?, ? examples/s]
```

2 Generation of the Large Lenguague Model

Now we're going to define the LLM using GPT4ALL, generate a prompt of tree sentences to test it, build the chain connecting the LLM with the prompt and finally generate the automatic resume, wich we wil compare with the human one.

[1]:
```
from datasets import load_from_disk

train_subset = load_from_disk("train_subset_5pct")

print(len(train_subset))
print(train_subset[0]["article"][:200])
print(train_subset[0]["highlights"])
```

```
14355
A protester in Ferguson was arrested during a demonstration on Thursday night -
and live-tweeted her entire experience. Brittany Ferrell, a nursing student at
the University of Missouri-Saint Louis, w
```

```
Brittany Ferrell, nursing student, was arrested with 12 people on Thursday .
They were calling on police take responsibility for Michael Brown's death .
Ms Ferrell tweeted as she was arrested, piled in a small wagon with 7 others .
They were accused of 'noise disruption', put in orange jumpsuits and cuffed .
Officers now being investigated, lawyers claim they 'overstretched powers'
```

To define the LLM we are going to use the Nomic AI Orca's model:

[11]:
```
# To generate the prompt and resume we are going to use Orca model from GPT4ALL
from gpt4all import GPT4All

# We are going to use a light model to test the response
model = GPT4All("orca-mini-3b-gguf2-q4_0.gguf")

with model.chat_session() as session:
    response = session.generate("Summarize in one sentence what machine learning␣
 ↪is.")
    print(response)
```

```
Downloading: 100%|| 1.98G/1.98G [07:39<00:00, 4.30MiB/s]
Verifying: 100%|| 1.98G/1.98G [00:03<00:00, 537MiB/s]
```

```
 "The ability to learn automatically is the power of artificial intelligence."
```

Having verified that the local instance has been successfully created and that the model has correctly generated the text according to the instructions we gave it, we will create a specific prompt to summarize the news in our dataset and test that the expected response is generated correctly.

[49]:
```
from gpt4all import GPT4All as G4

m = G4("orca-mini-3b-gguf2-q4_0.gguf")
prompt_text = "Summarize the following news article in 3 concise sentences.
 ↪\n\nArticle:\n" + sample_article[:4000] + "\n\nSummary:"

with m.chat_session() as sess:
    out = sess.generate(prompt_text, max_tokens=100, temp=0.7)  # no 'stop'
print("DIRECT OUTPUT:\n", repr(out))
```

```
DIRECT OUTPUT:
 " A protester in Ferguson was arrested during a demonstration on Thursday
night. She live-tweeted her experience and reported that nine women and four men
were arrested, including herself, while the group was protesting Michael Brown's
death. The arrest was reportedly related to noise disruption. Her bail was set
at $2,300 but she was released after several hours. Lawyers are investigating
whether the arrests were necessary and whether officers overstretched their
powers."
```

We are going to use the same sesion for time efficiency.

We will also adjust some of the variables in order to obtain the best possible output in the most efficient way for the model.

[55]:
```
import pandas as pd
from tqdm import tqdm
import time
```

[57]:
```
g4 = G4("orca-mini-3b-gguf2-q4_0.gguf")  # We inicializa the Orca Model

TRUNCATE_CHARS = 3500                    # Limiting article lenght prevents the␣
↪model from "getting lost,"
                                         # reduces memory, and increases␣
↪inference time.

MAX_TOKENS = 120                         # Limiting the lenght of the resume␣
↪helps to not waist time and resources.

TEMP = 0.7                               # Set generation temperature to 0.7 is␣
↪the ideal value for resume generation

RETRIES = 3500                           # Mitigates transient model failures or␣
↪I/O problems without aborting the entire process.

SLEEP_BETWEEN_RETRIES = 1.0              # Avoids overloading and allows time to␣
↪free up resources if the model fails due to temporary saturation.


PROMPT_PREFIX = (
    "Summarize the following news article in exactly 3 concise sentences.\n\n" #␣
↪Clear and direct prompt
    "Article:\n"
)
PROMPT_SUFFIX = "\n\nSummary:"
```

[59]:
```
# We will section the dataset into 50 values to increase the speed of summary␣
↪generation.

df = pd.DataFrame(train_subset[:50])
df["machine_summary"] = ""
df["status"] = ""
df.head()
```

[59]:

	article	highlights
0	A protester in Ferguson was arrested during a ...	
1	A day after confirming it had lost the ability...	
2	By . Jason Groves . PUBLISHED: . 19:31 EST, 14...	
3	By . Alexandra Klausner . PUBLISHED: . 01:19 E...	
4	By . Daily Mail Reporter . PUBLISHED: . 10:21 ...	

```
0  Brittany Ferrell, nursing student, was arreste...
1  Twitter has added photo filters to its Android...
2  Judge initially suggested that Carine Patry Ho...
3  A child in the park peed on a swing .\n'It's f...
4  Black bear broke through the screen door of a ...

                                         id machine_summary status
0  1e01f238418c31d4e9093f6334e0232babeb639a
1  6f89645bff243fe9ce2a0509e5ca01912abf0d10
2  966f243b6c64fa239239d8ffb7311827eb4dc50e
3  09db16eb2ffa5304737e385a39c7a2d69e8fecdb
4  6fd02a5e9b7f7f803e735448f8126925a8cfcd7c
```

[61]:
```python
def generate_direct_summary(article_text):
    short = article_text[:TRUNCATE_CHARS]
    prompt_text = PROMPT_PREFIX + short + PROMPT_SUFFIX
    # If the output is empty
    for attempt in range(1, RETRIES + 1):
        try:
            with g4.chat_session() as sess:
                out = sess.generate(prompt_text, max_tokens=MAX_TOKENS,␣
↪temp=TEMP)
            if out and out.strip():
                return out.strip()
        except Exception as e:
            # log internally and retry
            last_err = str(e)
        time.sleep(SLEEP_BETWEEN_RETRIES)
    # if it fails after retries, return empty and the error
    return ""
```

[79]:
```python
%%time

# We now apply our function to the dataframe

for i, row in tqdm(df.iterrows(), total=len(df)):
    art = row["article"]
    summary = generate_direct_summary(art)
    if summary:
        df.at[i, "machine_summary"] = summary
        df.at[i, "status"] = "ok"
    else:
        df.at[i, "machine_summary"] = ""
        df.at[i, "status"] = "failed"
```

```
100%|| 50/50 [2:24:23<00:00, 173.27s/it]

CPU times: total: 9h 28min 32s
```

```
Wall time: 2h 24min 23s
```

Now we verify that all of the articles were correctly summarized.

[86]:
```
df['status'].value_counts()
```

[86]:
```
status
ok    50
Name: count, dtype: int64
```

This means the model produced an output for all the entries

We save the results of the generated data for further use

[ ]:
```
df.to_csv("subset_50_with_summaries.csv", index=False)
```

Now let's do a quick check of the first three articles to verify the consistency of the discourse with which the summary was generated.

[93]:
```
for idx in range(min(3, len(df))):
    print("---- ARTICLE", idx, "----")
    print("Human summary:\n", df.loc[idx, "highlights"] if "highlights" in df.
 ↪columns else "(no human summary)")
    print("Machine summary:\n", df.loc[idx, "machine_summary"])
    print()
```

```
---- ARTICLE 0 ----
Human summary:
 Brittany Ferrell, nursing student, was arrested with 12 people on Thursday .
They were calling on police take responsibility for Michael Brown's death .
Ms Ferrell tweeted as she was arrested, piled in a small wagon with 7 others .
They were accused of 'noise disruption', put in orange jumpsuits and cuffed .
Officers now being investigated, lawyers claim they 'overstretched powers'
Machine summary:
 A protester in Ferguson was arrested during a demonstration on Thursday night
while live tweeting her experience. Brittany Ferrell, a nursing student at the
University of Missouri-Saint Louis, was one of 13 people detained by officers in
the conflicted Missouri city for "noise disruption". The detention has sparked
an investigation by the American Civil Liberties Union as lawyers accuse
officers of overstretching their powers. Scroll down for video .

---- ARTICLE 1 ----
Human summary:
 Twitter has added photo filters to its Android and iOS mobile apps .
The addition will help Twitter compete against Facebook-owned Instagram .
This is the first time the social network has offered image editing tools .
Machine summary:
 Twitter has rolled out its own library of retro filters for its Android and
iPhone apps. The eight filters are the usual suspects we've come to expect from
```

```
mobile photo apps, including desaturated, black and white, and high contrast.
There are auto-adjust and cropping options, as well as a helpful grid view that
lets you see what each filter will look like at once.

---- ARTICLE 2 ----
Human summary:
 Judge initially suggested that Carine Patry Hoskins played only a minor role .
She was paid £218,000 for her work between July 2011 and November 2012 .
Hoskins had an affair with Hacked Off lawyer David Sherbourne .
Leveson now says she had also played a key role in drawing up lines of .
questioning used by the inquiry's lead counsel Robert Jay .
Machine summary:
 Carine Patry Hoskins played a significant role in the Leveson Inquiry,
including providing legal advice and assisting with witness interviews and
assessments of evidence. This relationship with Hugh Grant's barrister was
revealed after revelations about her payment of £218,000 for her work between
July 2011 and November 2012. The judge initially suggested that her role was
minor, but under pressure to reveal more following the revelation, he admitted
she played a key role in drawing up lines of questioning used by lead counsel
Robert Jay and reviewing witness statements and researching
```

The model is generating legible understandable an coherent summaries, therefore since we obtained the expected outputs let's proceed to evaluate the LLM model.

3 Evaluation Metrics

3.1 Recall-Oriented Understudy for Gisting Evaluation (ROUGE)

The ROUGE metric is a set of metrics used in natural language processing (NLP) to evaluate the quality of automatic summaries or machine translations.

To evaluate the summaries generated by our model, we will evaluate:

- Recall: What percentage of the important words in the human summary are in the automatic summary? (ROUGE's main priority).
- Precision: What percentage of the words in the automatic summary were actually relevant?
- F1-Score: Combines recall and precision to give a balanced score, this balance score is the ROUGE metric we will use to evaluate de model.

[1]:
```python
import pandas as pd

# loading the outputs of the model we saved previusly
df = pd.read_csv("subset_50_with_summaries.csv")
```

[7]:
```python
import evaluate

rouge = evaluate.load("rouge")
```

```
references = df["highlights"].tolist()
generated = df["machine_summary"].tolist()

results = rouge.compute(
    predictions = generated,
    references = references
)


print("ROUGE:")
for key, value in results.items():
    print(f"{key}: {value:.4f}")
```

```
ROUGE:
rouge1: 0.3474
rouge2: 0.1314
rougeL: 0.2278
rougeLsum: 0.2799
```

ROUGE-1 - Evaluates the overlap of unigrams (individual words) between the generated summary and the reference summary. - Measures how much basic vocabulary matches.

ROUGE-2 - Evaluates the overlap of bigrams (pairs of consecutive words).

ROUGE-L - It is based on the Longest Common Subsequence (LCS), that is, the longest sequence of words in the same order that appears in both texts (not necessarily consecutive). - Captures the fluency and structure of the summary, not just exact n-gram matches.

Conclusions:

- The model captures key vocabulary (decent ROUGE-1), but fails to reproduce complete sentences (low ROUGE-2):
  - This indicates that it understands the topic, but does not always construct human-like sentences.
- Structural coherence is moderate (ROUGE-L ~0.23–0.28).
  - the generated summaries are somewhat aligned with humans, but there are still differences in order and style.
- Practical comparison:
  - In research, ROUGE-1 values between 0.35–0.45 are considered good for basic models ROUGE: *(A Package for Automatic Evaluation of Summaries Chin-Yew Lin Information Sciences Institute University of Southern California)*

3.2 Bilingual Evaluation Understudy (BLEU)

BLEU is an automatic algorithm used to evaluate the quality of machine-translated texts (machine translation) by comparing them with one or more human translation references. It is fast, computationally inexpensive, and correlates well with human judgment.

- Funcioning: BLEU measures the accuracy of "n-grams" in machine translation compared to human reference translations.

- Scoring: The result is a number between 0 and 1 (or 0 to 100), where a value closer to 1 indicates greater similarity to the human reference and, therefore, higher quality.
- Brevity Penalty: BLEU applies a penalty if the machine translation is too short compared to the reference, preventing the model from obtaining a high score simply by omitting difficult words.

[52]:
```
bleu = evaluate.load("bleu")

references = [[ref] for ref in df["highlights"].tolist()]
generated = df["machine_summary"].tolist()

# Calcular BLEU
results_bleu = bleu.compute(
    predictions = generated,
    references = references
)

rb = results_bleu['bleu']
print(f"BLEU score: {rb:.4f}")
```

```
BLEU score: 0.0927
```

Comparison with ROUGE:

- ROUGE-1 0.35: good keyword recall.
- BLEU 0.09: low accuracy of exact sentences.
- This means that the model understands the topic but does not replicate human writing.

Practical conclusions:

- The model captures vocabulary (decent ROUGE-1), but does not construct identical sentences (low BLEU).
  - the summaries are freer and less "copied" from the reference.
- Low BLEU does not mean perce that the model is bad.
  - In automatic summarization, BLEU is usually much lower than ROUGE. That's why this analysis will be complemented by the semantic metrics BERTScore.

3.3 BERTscore (Bidirectional Encoder Representations from Transformers)

BERTScore is an automatic metric for evaluating the quality of text generated by language models (such as chatbots or machine translators), based on semantic similarity rather than simply exact word matching.

Unlike traditional metrics such as BLEU or ROUGE, which look for identical words, BERTScore uses contextual embeddings from the BERT model to determine whether the meaning of two sentences is similar, even if they use different vocabulary.

- Funcioning: It calculates the similarity between the tokens of a generated text ("candidate") and a reference text ("ideal") using the cosine similarity between their embeddings.

- Precision, Recall, and F1: BERTScore provides metrics for precision (how much of the generated text makes sense relative to the reference), recall (how much of the reference was covered by the generated text), and an F1 score, which is the harmonic mean of the two.
- Contextual Awareness: Because it uses BERT, it understands the context of words, allowing it to recognize synonyms and paraphrases.

[116]:
```
bertscore = evaluate.load("bertscore")

references = df["highlights"].tolist()
generated = df["machine_summary"].tolist()

results_bert = bertscore.compute(
    predictions = generated,
    references = references,
    lang = "en"
)

bert_P = sum(results_bert["precision"]) / len(results_bert["precision"])
bert_R = sum(results_bert["recall"]) / len(results_bert["recall"])
bert_F1 = sum(results_bert["f1"]) / len(results_bert["f1"])

print("BERTScore (mean):")
print(f"Precision: {bert_P:.4f}")
print(f"Recall {bert_R:.4f}")
print(f"F1: {bert_F1:.4f}")
```

```
BERTScore (mean):
Precision: 0.8681
Recall 0.8721
F1: 0.8699
```

CONCLUSIONS:

- Low BLEU + high BERTScore: the model does not reproduce identical phrases, but it does maintain the meaning.
- Intermediate ROUGE: captures key vocabulary, although the structure does not always match.
- Although BLEU was low (0.09) and ROUGE showed moderate recall (0.35 in unigrams), BERTScore confirms that the model does convey the same meaning, even if it uses different words or paraphrases. This shows that the model does understand the content, even if it does not copy it word for word.

3.4 POS tagging analysis (Part-of-Speech)

Part-of-speech (POS) tagging involves identifying and classifying words in a text according to their grammatical category: nouns, verbs, adjectives, adverbs, determiners, etc.

This technique makes it possible to evaluate not only the semantic content of the summaries generated by a model, but also their linguistic structure and the way in which they construct sentences.

In the context of evaluating these automatic summaries, POS tagging offers a complementary per-

spective to the ROUGE, BLEU, and BERTScore metrics. These metrics quantify the lexical or semantic similarity between the generated text and the reference text, while grammatical analysis reveals style patterns and possible biases in the model.

This analysis will therefore allow us to answer key questions: - Does the model use a grammatical variety similar to that of human summaries? - Does it tend to overuse certain categories (e.g., nouns) and underuse others (e.g., action verbs)? - What implications do these patterns have for the quality and readability of the summaries?

[71]:
```
import nltk
import pandas as pd
from collections import Counter
import matplotlib.pyplot as plt
```

[73]:
```
# Sentence and word tokenizer & grammatical tagging model
nltk.download("punkt")
nltk.download("averaged_perceptron_tagger")
```

```
[nltk_data] Downloading package punkt to
[nltk_data]     C:\Users\Usuario\AppData\Roaming\nltk_data...
[nltk_data]   Package punkt is already up-to-date!
[nltk_data] Downloading package averaged_perceptron_tagger to
[nltk_data]     C:\Users\Usuario\AppData\Roaming\nltk_data...
[nltk_data]   Package averaged_perceptron_tagger is already up-to-
[nltk_data]       date!
```

[73]:
```
True
```

[75]:
```
# First let's tag the summaries generated by the model (POS-tagging)
pos_generated = []
for text in df["machine_summary"].tolist():
    tokens = nltk.word_tokenize(text)
    tags = nltk.pos_tag(tokens)
    pos_generated.extend([tag for _, tag in tags])
```

[77]:
```
# POS tagging for human summaries
pos_reference = []
for text in df["highlights"].tolist():
    tokens = nltk.word_tokenize(text)
    tags = nltk.pos_tag(tokens)
    pos_reference.extend([tag for _, tag in tags])
```

[83]:
```
# Frecuency
gen_counts = Counter(pos_generated)
ref_counts = Counter(pos_reference)

# comparative DataFrame
pos_df = pd.DataFrame({
    "Generated": gen_counts,
```

```
    "Reference": ref_counts
}).fillna(0)

pos_df.head()
```

[83]:

	Generated	Reference
DT	365.0	188.0
NN	599.0	396.0
IN	531.0	296.0
NNP	399.0	340.0
VBD	149.0	121.0

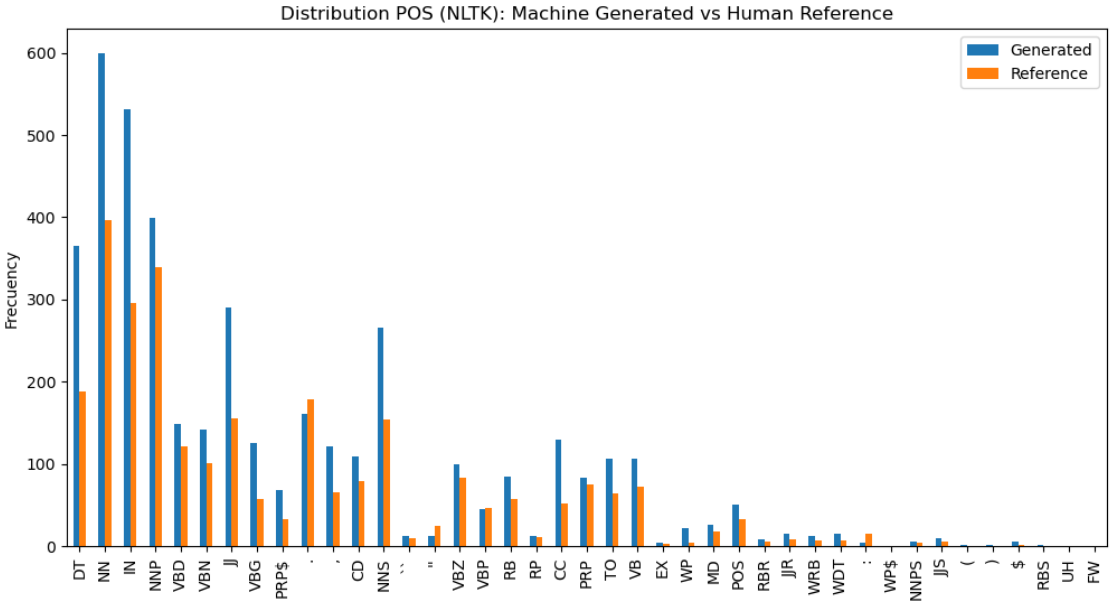
The NLKT library uses the following tags for text classification. These tags were taken from the Penn Treebank Project: https://www.ling.upenn.edu/courses/Fall_2003/ling001/penn_treebank_pos.html

POS Tags (NLTK)

Tag	Grammatical Category	Example
NN	Noun, singular or mass	cat, book
NNS	Noun, plural	cats, books
NNP	Proper noun, singular	London, Maria
NNPS	Proper noun, plural	Americans
VB	Verb, base form	run, eat
VBD	Verb, past tense	ran, ate
VBG	Verb, gerund/present participle	running, eating
VBN	Verb, past participle	eaten, driven
VBP	Verb, non-3rd person singular present	run, eat
VBZ	Verb, 3rd person singular present	runs, eats
JJ	Adjective	big, fast
JJR	Adjective, comparative	bigger, faster
JJS	Adjective, superlative	biggest, fastest
RB	Adverb	quickly, silently
RBR	Adverb, comparative	faster, sooner
RBS	Adverb, superlative	fastest, soonest
DT	Determiner	the, a, an
IN	Preposition or subordinating conjunction	in, on, because
PRP	Personal pronoun	he, she, it
PRP$	Possessive pronoun	his, her, its
CC	Coordinating conjunction	and, but, or
CD	Cardinal number	one, two, 100
EX	Existential “there”	there is, there are
FW	Foreign word	bonjour, gracias
MD	Modal verb	can, should, will
PDT	Predeterminer	all, both
POS	Possessive ending	’s
TO	“to” as infinitive marker	to go, to eat
UH	Interjection	oh, wow, hello

Tag	Grammatical Category	Example
WDT	Wh-determiner	which, that
WP	Wh-pronoun	who, what
WP$	Possessive wh-pronoun	whose
WRB	Wh-adverb	where, when

[88]:
```
# Comparative visualization
pos_df.plot(kind="bar", figsize=(12,6))
plt.title("Distribution POS (NLTK): Machine Generated vs Human Reference")
plt.ylabel("Frecuency")
plt.show()
```



Main observations from POS tagging analysis

Category	Generated vs Reference	Observation
Determiners (DT)	365 vs. 188	The model uses many more determiners, suggesting longer, more explanatory sentences.
Nouns (NN, NNS)	599/266 vs. 396/154	Greater use of nouns, more nominal style, and focus on entities.
Prepositions (IN)	531 vs 296	Excessive use of prepositions, dense and subordinate sentences.

Category	Generated vs Reference	Observation
Proper nouns (NNP)	399 vs 340	Similar, although the model mentions more specific entities.
Verbs (VB, VBD, VBN, VBZ, VBG, VBP)	Generally higher in generated text (VB: 107 vs 72, VBG: 125 vs 58)	Includes a good amount of actions, even more than humans in some cases.
Adjectives (JJ, JJR, JJS)	291/16/10 vs 156/9/6	More adjectives, descriptive summaries but potentially redundant.
Possessive pronouns (PRP$)	69 vs 33	Doubles the use of possessives, may sound less natural.
Coordinating conjunctions (CC)	130 vs. 52	Much greater use of conjunctions, long sentences with multiple clauses.
Punctuation (. ,)	Fewer periods, more commas	Long, subordinate clauses, less concise than humans.

POS tagging analysis reveals that the model tends to generate more nominal and descriptive summaries, with a high use of determiners, nouns, and adjectives, as well as long sentences with many prepositions and conjunctions. In contrast, human summaries are more concise and direct. This finding complements the automatic metrics and provides linguistic evidence about the model's style.

[ ]: