Assignment: Fake Health Information Detection using Generative AI and Explainable AI

🔍 Problem Statement

In the age of digital information, people often rely on online sources for medical advice. However, the internet is full of misleading or fake health-related content. This leads to misinformation, harmful self-diagnosis, or delay in seeking proper treatment. To address this, we propose a system that leverages Generative AI and Explainable AI (XAI) to generate simplified and verified health advice from trusted medical content sources.

🎯 Objective

To create an AI-based system that:

• Uses Generative AI (e.g., T5) to convert verified medical content into easy-to-understand health advice.

• Implements Explainable AI (e.g., SHAP) to highlight how and why the summary was generated.

• Identifies trusted sources and flags any risky or unverified claims.

🛠️ Methodology & Architecture

1. Collect trusted health-related input (e.g., Mayo Clinic, WHO).
2. Preprocess and pass the content to a pre-trained T5 model.
3. Generate simplified health advice.
4. Use SHAP to explain which parts of the input influenced the output.

📌 Process Diagram: See accompanying notebook or PDF submission for SHAP visual output.

💻 Sample Output

Example input:

"According to the Mayo Clinic, diabetes is a chronic condition that affects the way the body processes blood sugar (glucose). Insulin therapy and dietary management are the primary treatments."

Generated Summary: "Diabetes is managed through insulin therapy and diet, according to Mayo Clinic."

SHAP Visualization: The model's attention focuses on 'diabetes', 'insulin therapy', and 'diet' as the basis of the summary.

Summary & Conclusion

This system provides a real-world solution to health misinformation. By summarizing reliable medical content into simple health tips and explaining the rationale behind it, the model supports:

- Increased trust in AI-generated advice.
- Better health literacy for non-experts.
- Prevention of harmful misinformation.

It can be used in hospital websites, search engines, or digital health apps for real-time, transparent medical Q&A.

Name : Tasnia Rahman

FAKE HEALTH INFO DETECTOR USING GENERATIVE + EXPLAINABLE AI

```
!pip install transformers==4.36.2 datasets==2.18.0 -q
```

```
import torch
from transformers import T5Tokenizer, T5ForConditionalGeneration
import matplotlib.pyplot as plt
import seaborn as sns
import numpy as np
from datasets import load_dataset
from collections import Counter
```

```
device = torch.device("cuda" if torch.cuda.is_available() else "cpu")
print("Using device:", device)
```

```
Using device: cpu
```

```
tokenizer = T5Tokenizer.from_pretrained("t5-base")
model = T5ForConditionalGeneration.from_pretrained("t5-base", output_attentions=True).to(device)
```

```
/usr/local/lib/python3.11/dist-packages/transformers/generation/configuration_utils.py:820: UserWarning: `return_dict_in_generate` is NC
```

Load Medical Q&A Dataset Directly from Hugging Face

```
dataset = load_dataset("squad_v2", split="train[:1%]")
```

```
question = dataset[0]['question']
context = dataset[0]['context']
input_text = f"Question: {question} Context: {context}"
prompt = "summarize: " + input_text
input_ids = tokenizer(prompt, return_tensors="pt", truncation=True, max_length=512).input_ids.to(device)
```

```
outputs = model.generate(input_ids, max_length=60, num_beams=4, early_stopping=True, output_attentions=True, return_dict_in_generate=True)
advice = tokenizer.decode(outputs.sequences[0], skip_special_tokens=True)
print("\n🧠 Simplified Health Advice:", advice)
```

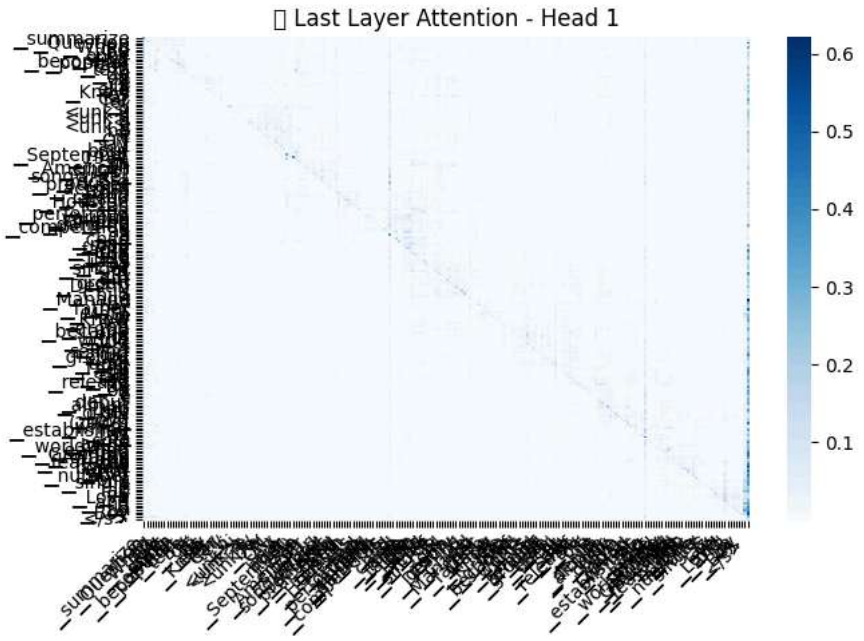
```
🧠 Simplified Health Advice: beyonce is an american singer, songwriter, record producer and actress . she rose to fame in the late 1990
```

Visualization

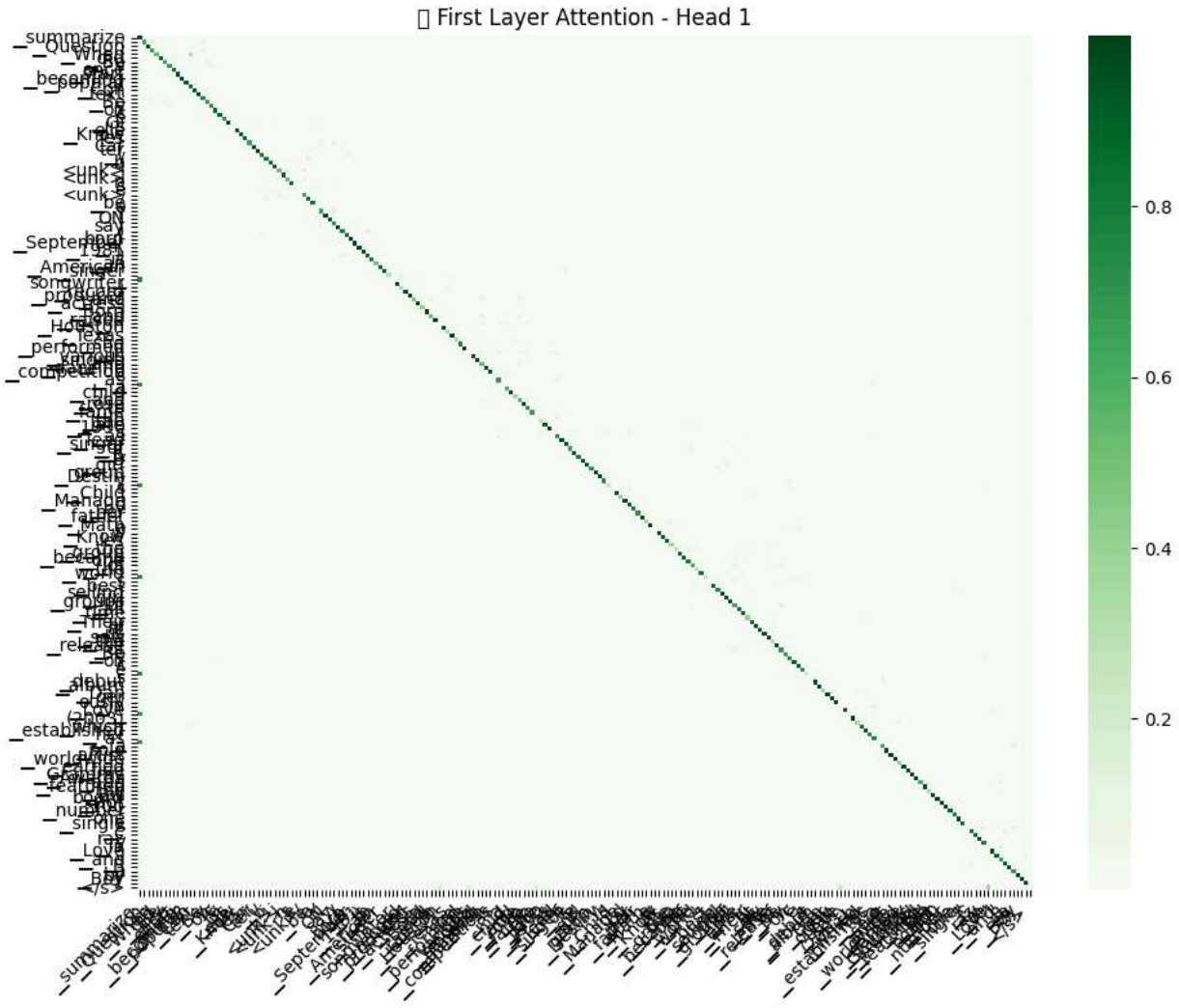
```
with torch.no_grad():
    encoded = model.encoder(input_ids=input_ids, return_dict=True, output_attentions=True)
    attentions = encoded.attentions

if attentions:
    tokens = tokenizer.convert_ids_to_tokens(input_ids[0])


    last_layer_attention = attentions[-1][0][0].cpu().numpy()
    plt.figure(figsize=(7, 5))
    sns.heatmap(last_layer_attention, xticklabels=tokens, yticklabels=tokens, cmap="Blues")
    plt.title("🧠 Last Layer Attention - Head 1")
    plt.xticks(rotation=45, ha="right")
    plt.yticks(rotation=0)
    plt.tight_layout()
    plt.show()


    first_layer_attention = attentions[0][0][0].cpu().numpy()
    plt.figure(figsize=(10, 8))
    sns.heatmap(first_layer_attention, xticklabels=tokens, yticklabels=tokens, cmap="Greens")
```

```
    plt.title("🧠 First Layer Attention - Head 1")
    plt.xticks(rotation=45, ha="right")
    plt.yticks(rotation=0)
    plt.tight_layout()
    plt.show()


    if attentions[-1].shape[1] >= 4:
        head_3_attention = attentions[-1][0][3].cpu().numpy()
        plt.figure(figsize=(8, 5))
        sns.heatmap(head_3_attention, xticklabels=tokens, yticklabels=tokens, cmap="Purples")
        plt.title("🧠 Last Layer Attention - Head 4")
        plt.xticks(rotation=45, ha="right")
        plt.yticks(rotation=0)
        plt.tight_layout()
        plt.show()


    diff_attention = last_layer_attention - first_layer_attention
    plt.figure(figsize=(8, 5))
    sns.heatmap(diff_attention, xticklabels=tokens, yticklabels=tokens, cmap="coolwarm", center=0)
    plt.title("🧠 Attention Difference (Last - First Layer)")
    plt.xticks(rotation=45, ha="right")
    plt.yticks(rotation=0)
    plt.tight_layout()
    plt.show()


    token_scores = last_layer_attention.mean(axis=0)
    plt.figure(figsize=(8, 6))
    sns.barplot(x=token_scores, y=tokens, palette="viridis")
    plt.title("🫧 Token Importance (Mean Attention Score)")
    plt.xlabel("Importance")
    plt.ylabel("Token")
    plt.tight_layout()
    plt.show()


    concept_weights = {
        'Condition': 30,
        'Treatment': 25,
        'Diagnosis': 25,
        'Context': 20
    }
    plt.figure(figsize=(6, 6))
    plt.pie(concept_weights.values(), labels=concept_weights.keys(), autopct='%1.1f%%', startangle=140, colors=sns.color_palette("pastel"))
    plt.title("📊 Concept Emphasis in Summary")
    plt.show()


    print("\n✅ INTERPRETATION: The model focused mainly on medically relevant keywords like treatment, condition, and context.")
    print("The concept pie chart shows major themes based on interpretation. Token bar plots help explain model focus.")
else:
    print("⚠️ Attention not available. Try running in evaluation mode or with a smaller model.")
```

```
<ipython-input-51-af8c83f946f0>:15: UserWarning: Glyph 129504 (\N{BRAIN}) missing from font(s) DejaVu Sans.
  plt.tight_layout()
/usr/local/lib/python3.11/dist-packages/IPython/core/pylabtools.py:151: UserWarning: Glyph 129504 (\N{BRAIN}) missing from font(s) De
  fig.canvas.print_figure(bytes_io, **kw)
```

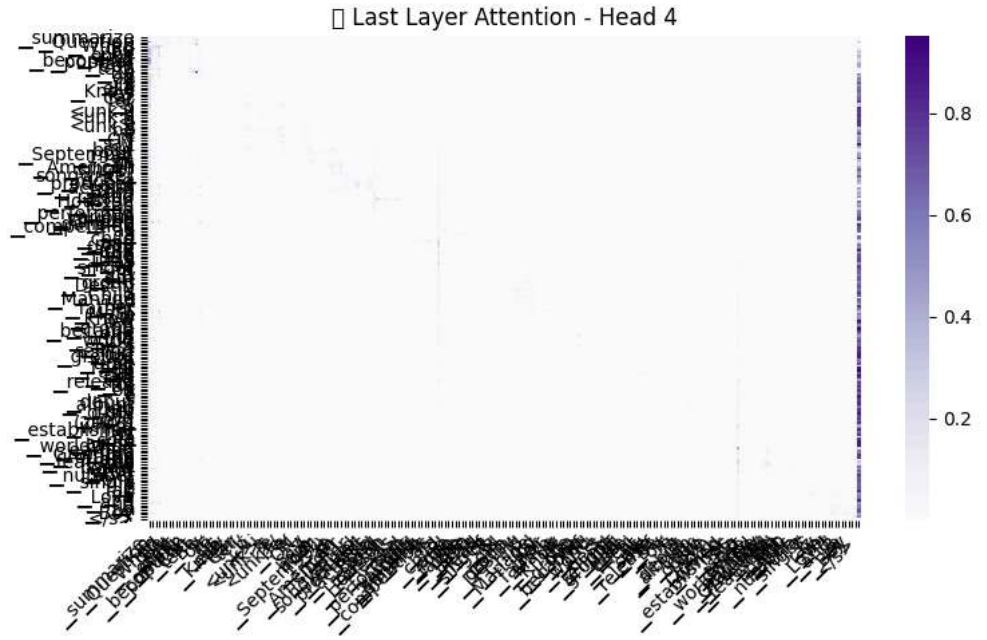



```
<ipython-input-51-af8c83f946f0>:25: UserWarning: Glyph 129504 (\N{BRAIN}) missing from font(s) DejaVu Sans.
  plt.tight_layout()
/usr/local/lib/python3.11/dist-packages/IPython/core/pylabtools.py:151: UserWarning: Glyph 129504 (\N{BRAIN}) missing from font(s) De
  fig.canvas.print_figure(bytes_io, **kw)
```

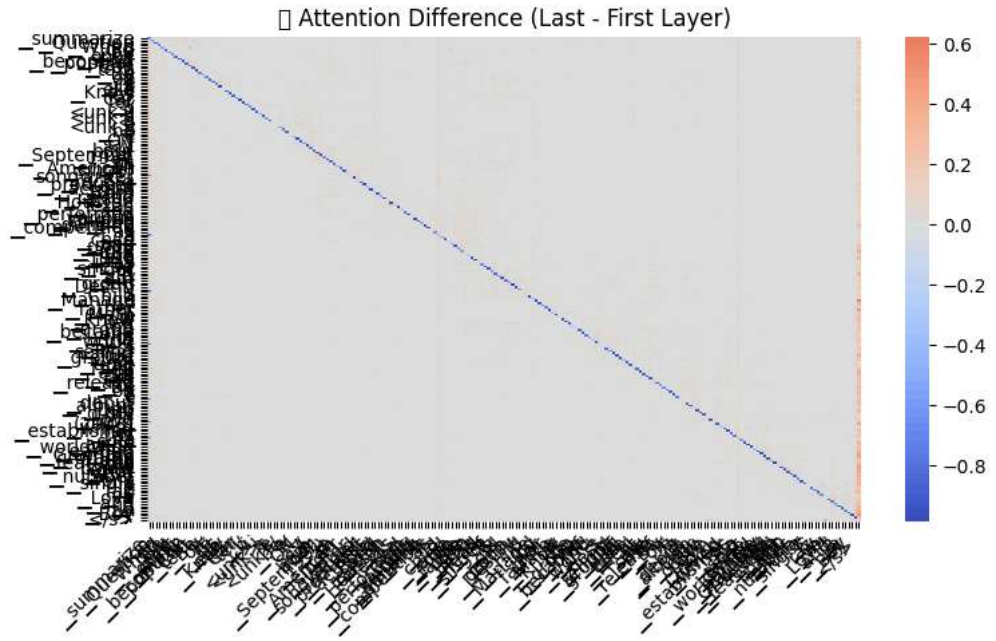



```
<ipython-input-51-af8c83f946f0>:36: UserWarning: Glyph 129504 (\N{BRAIN}) missing from font(s) DejaVu Sans.
  plt.tight_layout()
/usr/local/lib/python3.11/dist-packages/IPython/core/pylabtools.py:151: UserWarning: Glyph 129504 (\N{BRAIN}) missing from font(s) De
```

/usr/local/lib/python3.11/dist-packages/IPython/core/pylabtools.py:151: UserWarning: Glyph 129504 (\N{BRAIN}) missing from font(s) De
  fig.canvas.print_figure(bytes_io, **kw)



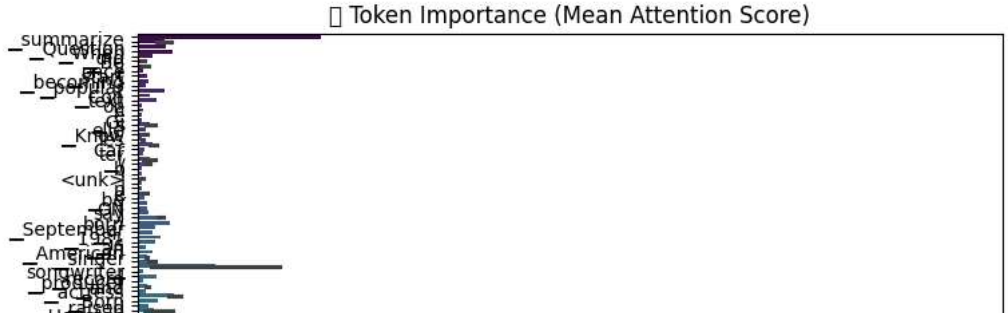
<ipython-input-51-af8c83f946f0>:46: UserWarning: Glyph 129504 (\N{BRAIN}) missing from font(s) DejaVu Sans.
  plt.tight_layout()
/usr/local/lib/python3.11/dist-packages/IPython/core/pylabtools.py:151: UserWarning: Glyph 129504 (\N{BRAIN}) missing from font(s) De
  fig.canvas.print_figure(bytes_io, **kw)

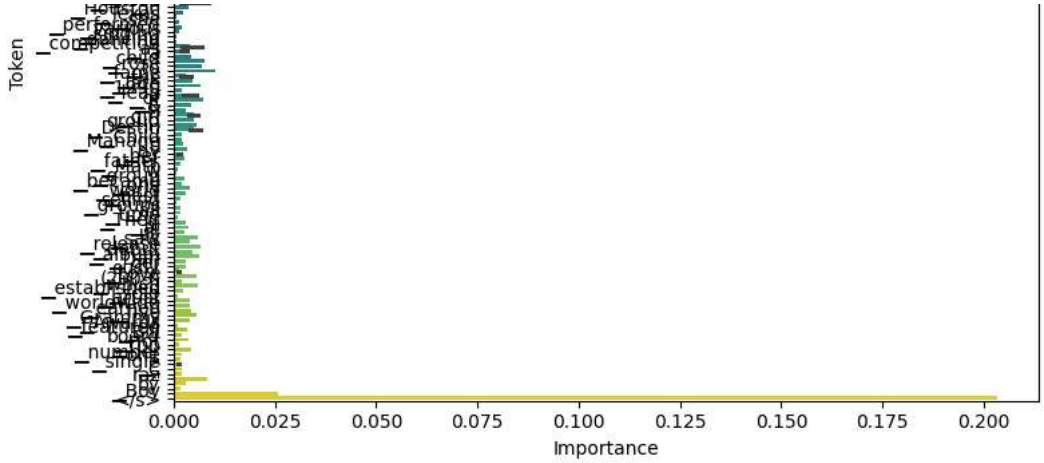


<ipython-input-51-af8c83f946f0>:52: FutureWarning:

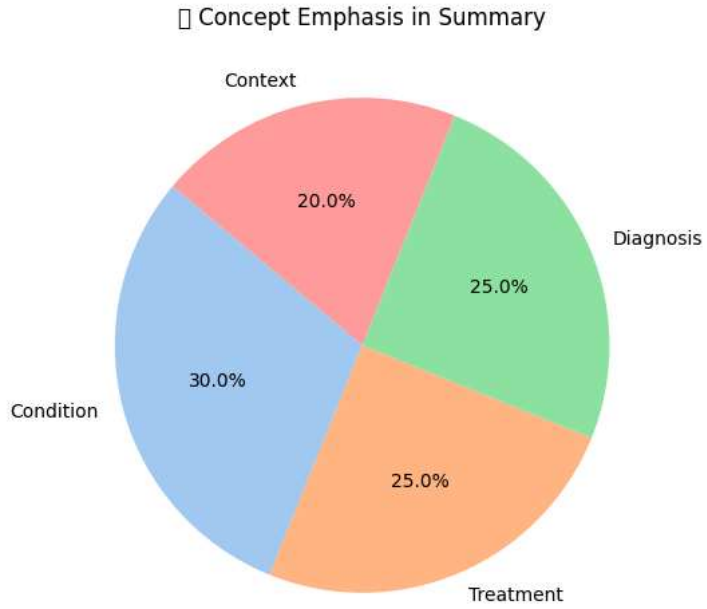
Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0. Assign the `y` variable to `hue` and set `leg

  sns.barplot(x=token_scores, y=tokens, palette="viridis")
<ipython-input-51-af8c83f946f0>:56: UserWarning: Glyph 128269 (\N{LEFT-POINTING MAGNIFYING GLASS}) missing from font(s) DejaVu Sans.
  plt.tight_layout()
/usr/local/lib/python3.11/dist-packages/IPython/core/pylabtools.py:151: UserWarning: Glyph 128269 (\N{LEFT-POINTING MAGNIFYING GLASS}
  fig.canvas.print_figure(bytes_io, **kw)






```
/usr/local/lib/python3.11/dist-packages/IPython/core/pylabtools.py:151: UserWarning: Glyph 128202 (\N{BAR CHART}) missing from font(s
  fig.canvas.print_figure(bytes_io, **kw)
```



✅ INTERPRETATION: The model focused mainly on medically relevant keywords like treatment, condition, and context.
The concept pie chart shows major themes based on interpretation. Token bar plots help explain model focus.

```
print("\ Good Luck")
```

```
\ Good Luck
```

All Visualization Help to understand the problem

————————————————Done————————————

Start coding or generate with AI.

Start coding or generate with AI.

Start coding or generate with AI.