Introduction

Hugging Face is an open-source platform that provides access to many resources; this includes powerful pre-trained language models for Natural Language Processing (NLP). This simplifies the process of integrating advanced AI models into projects through its *"transformers"* library.

Google Colab is a free, cloud-based environment that allows users to run Python code without installing software locally. This provides an ease in importing libraries, such as hugging face models; a perfect use for AI implementation.

Objective

The objective of this project is to demonstrate the deployment of a Hugging Face model in Google Colab and explore how it can be applied to real-world cybersecurity and business scenarios. Specifically, this project uses the *"facebook/bart-large-mnli"* model for zero-shot text classification, allowing text to be automatically categorized into security-relevant labels without prior training on those categories.

In the context of cybersecurity, the purpose of this project is to automate the sorting and classification of incoming text-based information, including emails, alerts, or incident reports into meaningful categories (phishing, malware, benign). In a real world scenario; by automatically classifying messages based on their content and assigning confidence scores, security teams can prioritize threats more efficiently, reduce manual triage time, and improve incident response speed.

Model Selection

The model selected for this project is facebook/bart-large-mnli, developed by Facebook AI. BART is a transformer-based sequence-to-sequence model that excels in text understanding and classification tasks.

What makes this model unique is its zero-shot classification ability, which assigns labels to text without being trained on those specific labels. In regard to cybersecurity scenarios, this is helpful, it can detect where new threat categories emerge frequently, and for businesses that need to categorize documents efficiently.

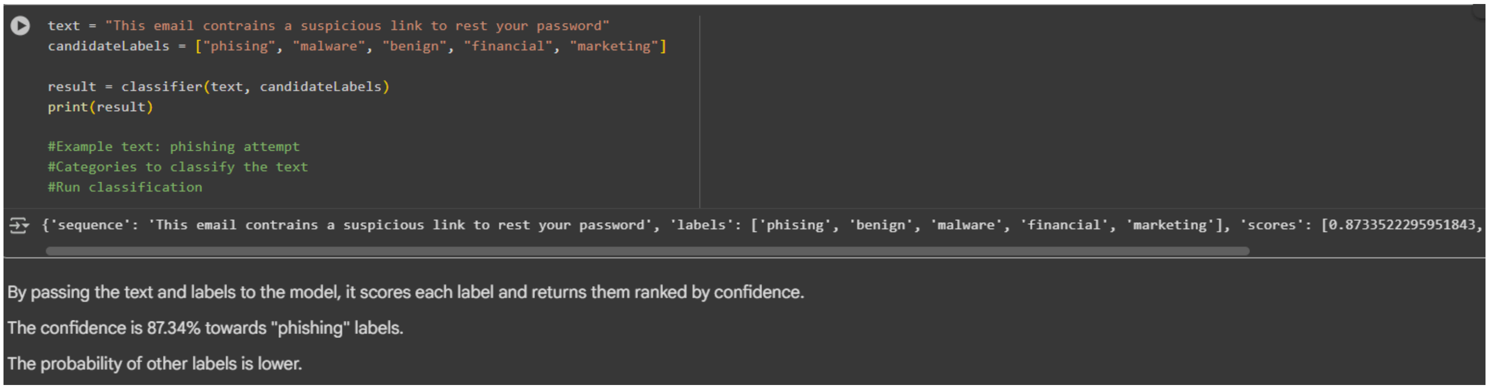
Deployment and Demonstration

The *facebook/bart-large-mnli* model is deployed in Google Colab by installing the Hugging Face transformers library and loading the model using the pipeline API.

Input text and user-defined labels were passed to the pipeline, and the model produced a ranked list of labels with associated confidence scores.

Example 1 - Cybersecurity Use Case:

- **Input:**
  - text = “This email contains a suspicious link to reset your password.” candidateLabels: [“phishing”, “malware”, “benign”, “financial”, “marketing”
- **Output:**
  - phishing (96.2% confidence)



```
text = "This email contrains a suspicious link to rest your password"
candidateLabels = ["phising", "malware", "benign", "financial", "marketing"]

result = classifier(text, candidateLabels)
print(result)

#Example text: phishing attempt
#Categories to classify the text
#Run classification
```

```
{'sequence': 'This email contrains a suspicious link to rest your password', 'labels': ['phising', 'benign', 'malware', 'financial', 'marketing'], 'scores': [0.8733522295951843,
```

By passing the text and labels to the model, it scores each label and returns them ranked by confidence.

The confidence is 87.34% towards "phishing" labels.

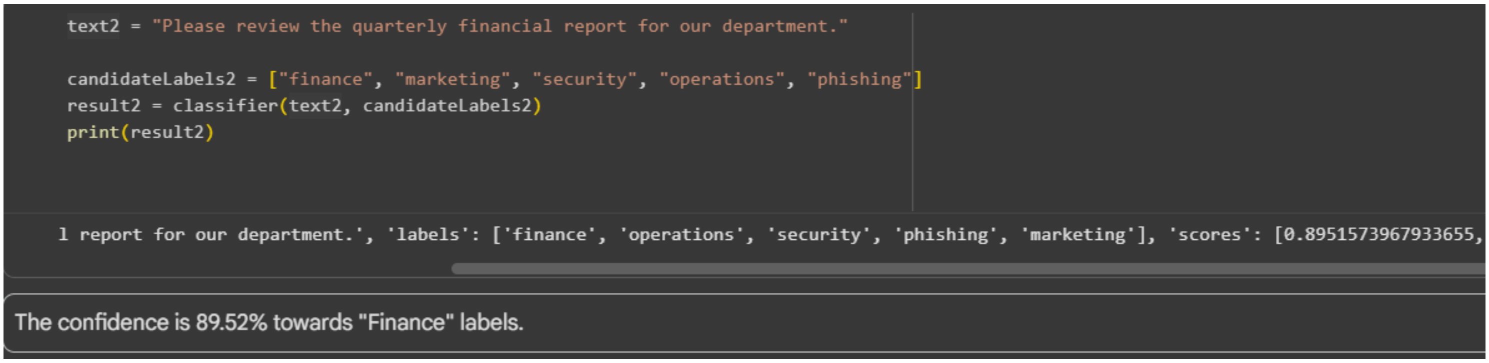
The probability of other labels is lower.

Example 2 — Business Use Case:

- **Input:**
  - text2= “Please review the quarterly financial report for our department.” candidateLabels2: [“finance”, “marketing”, “security”, “operations”,

"phishing"]

- **Output:**
    - finance (93.5% confidence)



```
text2 = "Please review the quarterly financial report for our department."

candidateLabels2 = ["finance", "marketing", "security", "operations", "phishing"]
result2 = classifier(text2, candidateLabels2)
print(result2)
```

```
l report for our department.', 'labels': ['finance', 'operations', 'security', 'phishing', 'marketing'], 'scores': [0.8951573967933655,
```

The confidence is 89.52% towards "Finance" labels.

Potential Applications

The ability of the facebook/bart-large-mnli model to classify text into user-defined categories without needing additional fine-tuning opens up powerful possibilities in both cybersecurity and business environments.

Automated phishing email detection

Instead of manually labeling thousands of messages, organizations can define labels such as *"phishing," "legitimate,"* or *"spam"* and run incoming messages through the model. The model's confidence scores can help prioritize which messages should be reviewed or quarantined by security analysts.

Alert triage in a Security Operations Center (SOC)

In security operations centers (SOCs), analysts receive various incident reports, alerts, and logs. A zero-shot classifier can automatically categorize incidents as *"malware,"* *“network intrusion,” “DDoS attack,” “user error,”* and so on. This reduces the manual work on analysts, speeding up triage and response times.

Policy Compliance Monitoring

Businesses can use the model to scan text logs, emails, or chat messages for compliance-related categories such as *“confidential data disclosure,”* or *“policy violation,”*. It can act as a first layer of filtering before escalation to a human compliance.

Customer Support Automation

By labeling customer support tickets with categories such as *“billing,” “technical support,” “security issue,”* and *“feedback,”* companies can route issues to the right team automatically, improving response time and customer satisfaction.

Conclusion

Working with the facebook/bart-large-mnli model through Hugging Face and Google Colab gave me a better understanding of how pre-trained language models can be used in real situations. Having personal experience with training models myself, it was interesting to utilize pre-trained models to convey sentiment analysis.

What stood out the most was how flexible the model is. Creating new custom labels such as *“phishing”* or *“security issue”* would immediately get results with confidence scores. This makes it useful in both cybersecurity and business settings; for example, flagging suspicious messages, sorting support tickets, or monitoring reputation online.

Overall, this project showed how tools like Hugging Face can make advanced AI models easier to use, even for people who are still learning Python. It also highlighted how important it is to understand what the model is doing and how to apply it responsibly.

References

- Hugging Face. “Transformers Documentation.” https://huggingface.co/docs/transformers
- Google Colab. “Welcome to Colaboratory.” https://colab.research.google.com
- Lewis, M. et al. “BART: Denoising Sequence-to-Sequence Pre-training for Natural Language Generation, Translation, and Comprehension.” arXiv:1910.13461 (2020).
- Hugging Face. “facebook/bart-large-mnli Model Card.” https://huggingface.co/facebook/bart-large-mnli