Advanced Phishing Detection System

Team Information

Names: Ishika Upadhyay , Celest K. Bernard

Student IDs: 111442194 , 111394794

Emails: ishika.upadhyay@ucdenver.edu , celestkester.bernard@ucdenver.edu

Class Information

Course Name: Cyber/Infrastructure Defense

Semester: Fall 2024 (1)

Instructor: J. Haadi Jafarian

ABSTRACT

Phishing attacks pose a serious threat to online users, leading to financial loss, identity theft, and compromised data security. Our project focuses on developing an advanced phishing detection system leveraging machine learning and natural language processing (NLP) techniques. By combining multilingual capabilities and automated classification, we create a robust model capable of detecting phishing attempts in various languages. Our system is built using DistilBERT, a lightweight variant of BERT, and fine-tuned using datasets of legitimate and phishing emails. Results indicate significant improvements in detection accuracy across multiple languages. This approach has the potential to make email communication more secure and reduce phishing-related cybercrime.

1. Introduction

1.1 Problem

Phishing attacks are one of the most common cyber threats today, impacting millions of users worldwide. Attackers use deceptive emails and links to trick users into revealing sensitive information. Despite existing detection systems, attackers continue to evolve their tactics, making phishing more sophisticated. The lack of multilingual support in existing phishing detection systems further compounds the problem. This project addresses the issue by developing a multilingual phishing detection model that can analyze emails in multiple languages and accurately classify them as phishing or legitimate. Failure to address this issue will result in continued exploitation of vulnerable users, potentially leading to severe financial and privacy breaches.[1]

1.2 Project Statement

This project aims to develop a machine-learning-based multilingual phishing detection system using DistilBERT. The system can classify emails and URLs as either phishing or legitimate in multiple languages, improving detection accuracy, speed, and robustness.

1.3 Approach

Our approach involves using natural language processing (NLP) techniques and machine learning to build a multilingual phishing detection system. We utilize the DistilBERT transformer model to process email messages and URLs for phishing traits. By training the model on multiple datasets and testing it in different languages, we ensure its generalization and applicability in real-world scenarios. The system employs preprocessing steps like tokenization, stop-word removal, and truncation, followed by training on a labeled dataset. Our system achieves its objective by combining three datasets, training a binary classifier, and evaluating its performance using precision, recall, F1 score, and accuracy.

1.4 Organization of this Project Report

- **Chapter 2** covers the background, including key concepts and a literature review of phishing detection systems.
- **Chapter 3** presents the architecture and design of the phishing detection system with code snippets.
- **Chapter 4** discusses the methodology, results, and analysis of our system.
- **Chapter 5** outlines the conclusion, contributions, and future work of our project.

2. Background

2.1 Key Concepts

2.1.1 Natural Language Processing (NLP)

NLP is a field of AI that enables machines to understand, interpret, and respond to human language. For this project, NLP techniques like tokenization, stop-word removal, and multilingual support play a crucial role in preprocessing email text for phishing detection.

2.1.2 Transformers and DistilBERT

DistilBERT is a smaller, faster, and more efficient version of BERT. It is a transformer-based model that excels in natural language understanding. Our model utilizes DistilBERT to classify emails and URLs into phishing or legitimate.

2.2 Related Work

Several previous works have focused on phishing detection. Traditional methods rely on heuristic rules or signature-based systems, which struggle against modern phishing tactics. More recent approaches use machine learning models, such as SVMs, Naive Bayes, and neural networks, to classify phishing emails. However, most of these models do not support multilingual analysis. Our project addresses this gap by supporting multiple languages in a unified framework.[2]

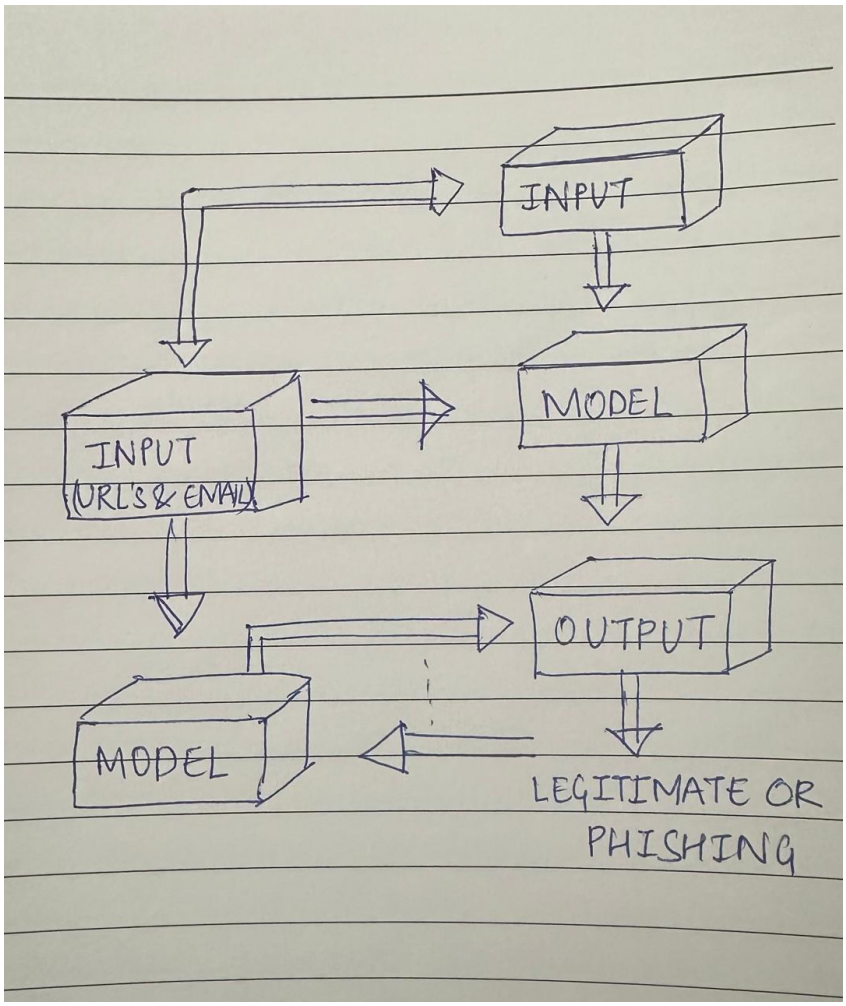
3. Architecture

3.1 High-Level Design

The system consists of the following components:

1. **Data Preprocessing**: Cleans and tokenizes email messages and URLs.
2. **Multilingual Model**: Uses DistilBERT for feature extraction and classification.
3. **Classifier**: Binary classifier with output labels: "Legitimate" or "Phishing."
4. **Evaluation Module**: Measures performance using accuracy, precision, recall, and F1 score.

3.2 Data Flow Diagram



3.3 Implementation

The implementation details are as follows:

- **Data**: Three datasets are combined and preprocessed.
- **Model**: DistilBERT is fine-tuned for binary classification.
- **Tokenization**: Input text is tokenized using the Hugging Face AutoTokenizer.
- **Evaluation**: Performance is evaluated using metrics such as precision, recall, and F1 score.

3.4 Code Snippets

[1]:
```
import pandas as pd
import torch
import nltk
from transformers import AutoTokenizer, AutoModelForSequenceClassification, Trainer, TrainingArguments
from datasets import Dataset, DatasetDict
from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score, precision_recall_fscore_support, confusion_matrix
import matplotlib.pyplot as plt
import seaborn as sns

# Download necessary NLTK data
nltk.download('stopwords')
nltk.download('punkt')
stop_words = set(nltk.corpus.stopwords.words('english'))

# Preprocessing function for multilingual support
def preprocess_text(text):
    tokens = nltk.word_tokenize(text.lower())
    tokens = [word for word in tokens if word.isalnum() and word not in stop_words]
    return ' '.join(tokens)

# Load and combine datasets
data1 = pd.read_csv('/Users/ishikaupadhyay/Desktop/cyber project/dataset_phishing.csv')
data2 = pd.read_csv('/Users/ishikaupadhyay/Desktop/cyber project/emails.csv')
data3 = pd.read_csv('/Users/ishikaupadhyay/Desktop/cyber project/phishing_Legitimate_full.csv')

combined_data = pd.concat([data1, data2, data3], ignore_index=True)

# Check and clean missing values
combined_data['message'] = combined_data['message'].fillna('')
combined_data['url'] = combined_data['url'].fillna('')
combined_data['status'] = combined_data['status'].fillna('unknown')

# Prepare inputs and labels
combined_data['text'] = combined_data.apply(
    lambda row: f"Email: {preprocess_text(row['message'])} URL: {row['url']}",
    axis=1
)
```

```
    axis=1
)
combined_data['label'] = combined_data['status'].replace({'legitimate': 0, 'phishing': 1})

# Filter unknown statuses if any
combined_data = combined_data[combined_data['label'].isin([0, 1])]

# Train-test split
train_df, test_df = train_test_split(combined_data[['text', 'label']], test_size=0.2, random_state=42, stratify=combined_data['label'])

# Convert to Hugging Face Dataset
dataset = DatasetDict({
    "train": Dataset.from_pandas(train_df),
    "test": Dataset.from_pandas(test_df),
})

# Tokenizer and model setup (smaller multilingual model)
model_name = "distilbert-base-multilingual-cased"
tokenizer = AutoTokenizer.from_pretrained(model_name)
model = AutoModelForSequenceClassification.from_pretrained(model_name, num_labels=2)

# Tokenization function
def tokenize_function(examples):
    return tokenizer(examples["text"], truncation=True, padding="max_length", max_length=256)

# Tokenize datasets
tokenized_datasets = dataset.map(tokenize_function, batched=True)
tokenized_datasets = tokenized_datasets.remove_columns(["text"])
tokenized_datasets = tokenized_datasets.rename_column("label", "labels")
tokenized_datasets.set_format("torch")

# Metrics function
def compute_metrics(pred):
    labels = pred.label_ids
    preds = pred.predictions.argmax(-1)
    precision, recall, f1, _ = precision_recall_fscore_support(labels, preds, average="binary")
    acc = accuracy_score(labels, preds)
    return {"accuracy": acc, "f1": f1, "precision": precision, "recall": recall}

# Training arguments
training_args = TrainingArguments(
```

```
    return {"accuracy": acc, "f1": f1, "precision": precision, "recall": recall}

# Training arguments
training_args = TrainingArguments(
    output_dir="./results",
    evaluation_strategy="epoch",
    save_strategy="epoch",
    learning_rate=5e-5,
    per_device_train_batch_size=16,
    num_train_epochs=3,
    weight_decay=0.01,
    logging_dir="./logs",
    logging_steps=50,
    save_total_limit=1,
    load_best_model_at_end=True,
)

# Trainer setup
trainer = Trainer(
    model=model,
    args=training_args,
    train_dataset=tokenized_datasets["train"],
    eval_dataset=tokenized_datasets["test"],
    tokenizer=tokenizer,
    compute_metrics=compute_metrics,
)

# Training
print("Starting model training...")
trainer.train()
print("Model training completed.")

# Save the fine-tuned model
model.save_pretrained("./distilbert_phishing_model")
tokenizer.save_pretrained("./distilbert_phishing_model")

# Evaluate the model on the test set
test_results = trainer.evaluate(tokenized_datasets["test"])
print(f"Accuracy: {test_results['eval_accuracy']:.4f}")
print(f"F1 Score: {test_results['eval_f1']:.4f}")
```

```
predicted_labels = predictions.predictions.argmax(-1)
true_labels = predictions.label_ids

# Confusion Matrix
cm = confusion_matrix(true_labels, predicted_labels)
plt.figure(figsize=(8, 6))
sns.heatmap(cm, annot=True, fmt='d', cmap='Blues', xticklabels=["Legitimate", "Phishing"], yticklabels=["Legitimate", "Phishing"])
plt.xlabel('Predicted Labels')
plt.ylabel('True Labels')
plt.title('Confusion Matrix')
plt.show()

# Prediction function with multilingual support
def classify_email(email, url, threshold=0.5):
    input_text = f"Email: {preprocess_text(email)} URL: {url}"
    inputs = tokenizer(input_text, return_tensors="pt", truncation=True, padding="max_length", max_length=256).to(model.device)
    outputs = model(**inputs)
    probabilities = torch.softmax(outputs.logits, dim=1)
    phishing_probability = probabilities[0][1].item()
    return "Phishing" if phishing_probability > threshold else "Legitimate"
```

```
[nltk_data] Downloading package stopwords to
[nltk_data]     /Users/ishikaupadhyay/nltk_data...
[nltk_data]   Package stopwords is already up-to-date!
[nltk_data] Downloading package punkt to
[nltk_data]     /Users/ishikaupadhyay/nltk_data...
[nltk_data]   Package punkt is already up-to-date!
Some weights of DistilBertForSequenceClassification were not initialized from the model checkpoint at distilbert-base-multilingual-cased and
are newly initialized: ['classifier.bias', 'classifier.weight', 'pre_classifier.bias', 'pre_classifier.weight']
You should probably TRAIN this model on a down-stream task to be able to use it for predictions and inference.
```

Error displaying widget

Error displaying widget

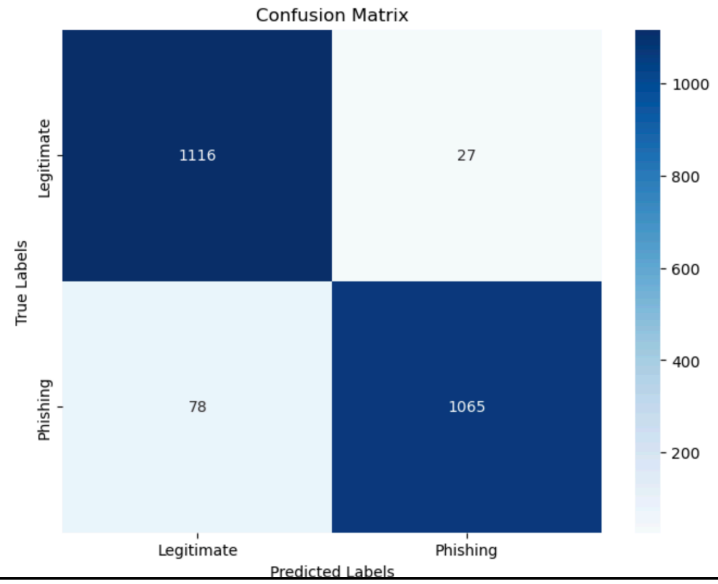
```
/opt/anaconda3/lib/python3.12/site-packages/transformers/training_args.py:1568: FutureWarning: `evaluation_strategy` is deprecated and will b
e removed in version 4.46 of 🤗 Transformers. Use `eval_strategy` instead
  warnings.warn(
/var/folders/qm/4wvlg1w17l73f4hhx2gmf8mc0000gn/T/ipykernel_5895/2256443254.py:92: FutureWarning: `tokenizer` is deprecated and will be remove
d in version 5.0.0 for `Trainer.__init__`. Use `processing_class` instead.
  trainer = Trainer(
Starting model training...
```

[1716/1716 50:22, Epoch 3/3]

[1716/1716 50:22, Epoch 3/3]

Epoch	Training Loss	Validation Loss	Accuracy	F1	Precision	Recall
1	0.165000	0.175078	0.947069	0.946813	0.951413	0.942257
2	0.076700	0.166274	0.954068	0.953020	0.975275	0.931759
3	0.047100	0.235424	0.954068	0.953724	0.960924	0.946632

```
Model training completed.
Accuracy: 0.9541
F1 Score: 0.9530
```



[13]:
```python
examples = {
    "English": [
        {"email": "Dear customer, your order has been shipped.", "url": "https://www.legitstore.com/track"},
        {"email": "Urgent: Account suspended. Verify now!", "url": "http://phishing-scam.com/login"},
    ],
    "Spanish": [
        {"email": "Estimado cliente, su pedido ha sido enviado.", "url": "https://www.tiendalegitima.es/rastreo"},
        {"email": "¡Alerta! Su cuenta está bloqueada. Verifique ahora.", "url": "http://phishing-estafa.es/login"},
    ],
    "German": [
        {"email": "Sehr geehrter Kunde, Ihre Bestellung wurde versandt.", "url": "https://www.echterladen.de/verfolgung"},
        {"email": "Achtung: Ihr Konto wurde gesperrt. Jetzt verifizieren!", "url": "http://phishing-betrug.de/einloggen"},
    ],
    "French": [
        {"email": "Urgent : Compte suspendu. Vérifiez maintenant !", "url": "http://phishing-arnaque.fr/login"},
    ],
    "Italian": [
        {"email": "Urgente: Account sospeso. Verifica ora!", "url": "http://phishing-truffa.it/accesso"},
    ],
}


for language, emails in examples.items():
    print(f"Testing {language} examples:")
    for example in emails:
        email = example["email"]
        url = example["url"]
        result = classify_email(email, url)
        print(f"Email: {email}\nURL: {url}\nResult: {result}\n")
```

```
Testing English examples:
Email: Dear customer, your order has been shipped.
URL: https://www.legitstore.com/track
Result: Legitimate

Email: Urgent: Account suspended. Verify now!
URL: http://phishing-scam.com/login
Result: Phishing

Testing Spanish examples:
```

```
Testing English examples:
Email: Dear customer, your order has been shipped.
URL: https://www.legitstore.com/track
Result: Legitimate

Email: Urgent: Account suspended. Verify now!
URL: http://phishing-scam.com/login
Result: Phishing

Testing Spanish examples:
Email: Estimado cliente, su pedido ha sido enviado.
URL: https://www.tiendalegitima.es/rastreo
Result: Legitimate

Email: ¡Alerta! Su cuenta está bloqueada. Verifique ahora.
URL: http://phishing-estafa.es/login
Result: Phishing

Testing German examples:
Email: Sehr geehrter Kunde, Ihre Bestellung wurde versandt.
URL: https://www.echterladen.de/verfolgung
Result: Legitimate

Email: Achtung: Ihr Konto wurde gesperrt. Jetzt verifizieren!
URL: http://phishing-betrug.de/einloggen
Result: Phishing

Testing French examples:
Email: Urgent : Compte suspendu. Vérifiez maintenant !
URL: http://phishing-arnaque.fr/login
Result: Phishing

Testing Italian examples:
Email: Urgente: Account sospeso. Verifica ora!
URL: http://phishing-truffa.it/accesso
Result: Phishing
```

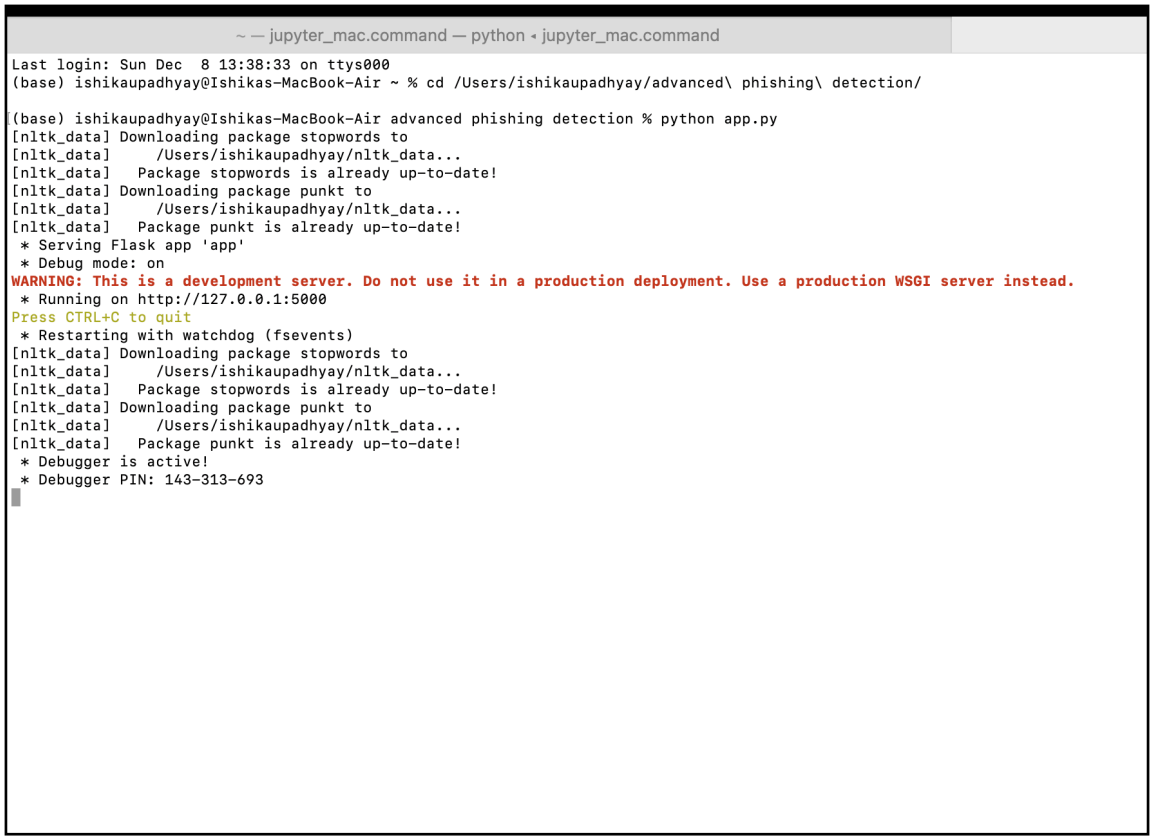
[ ]:

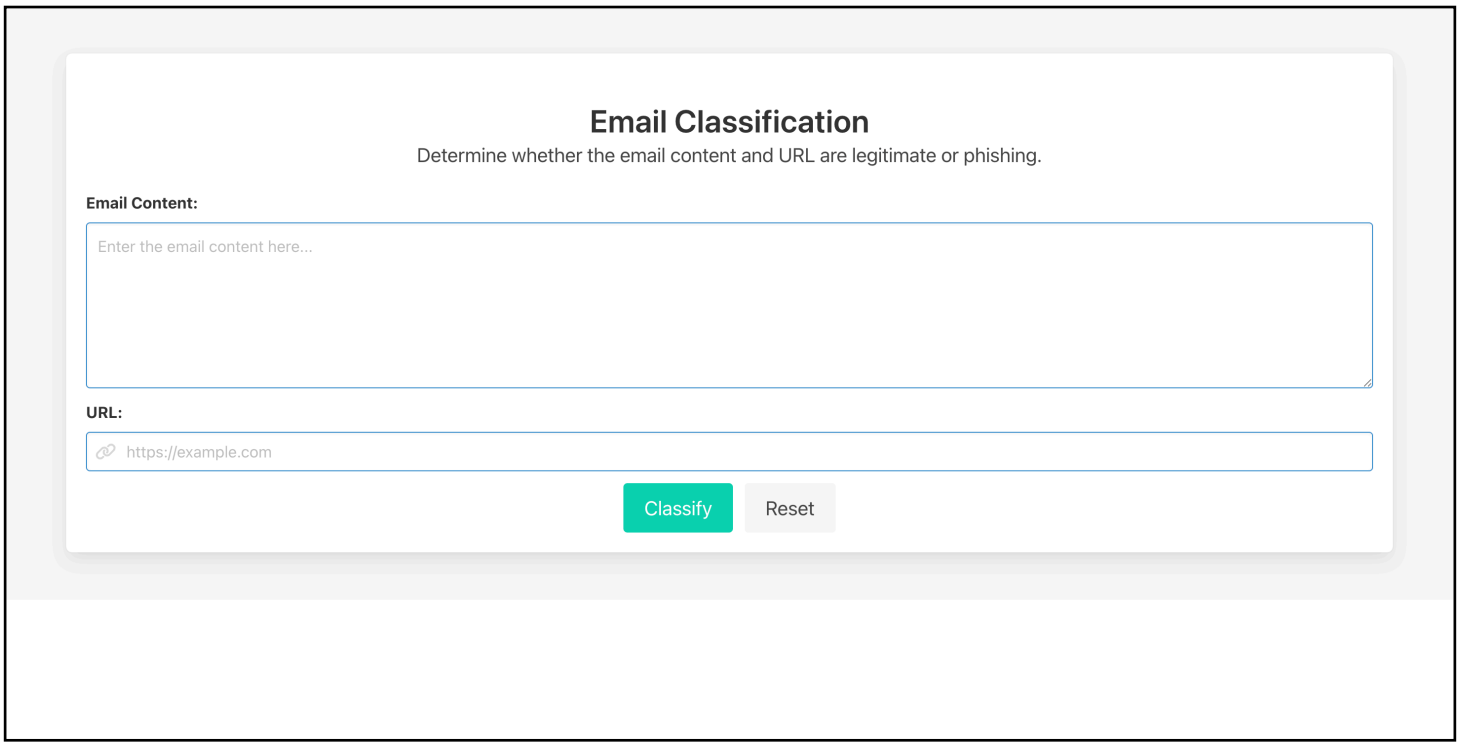
4. Methodology, Results, and Analysis

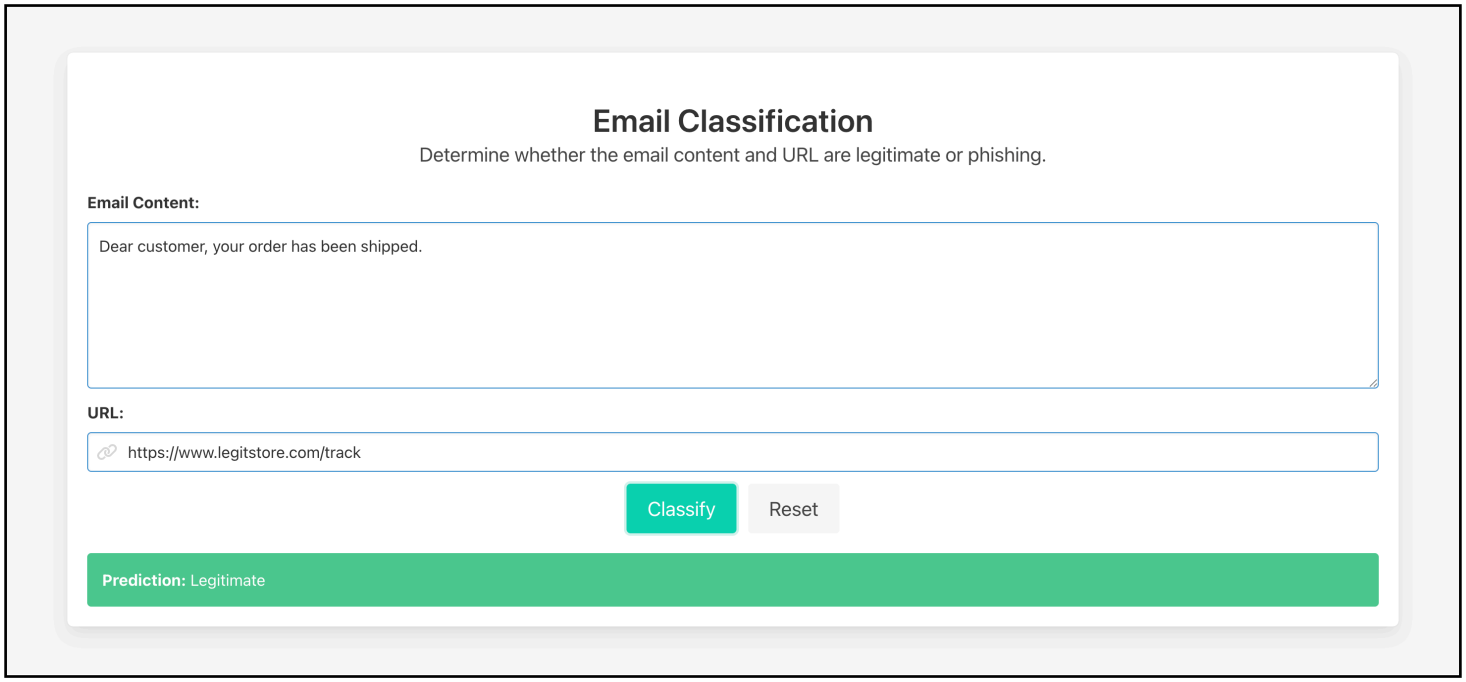
4.1 Methodology

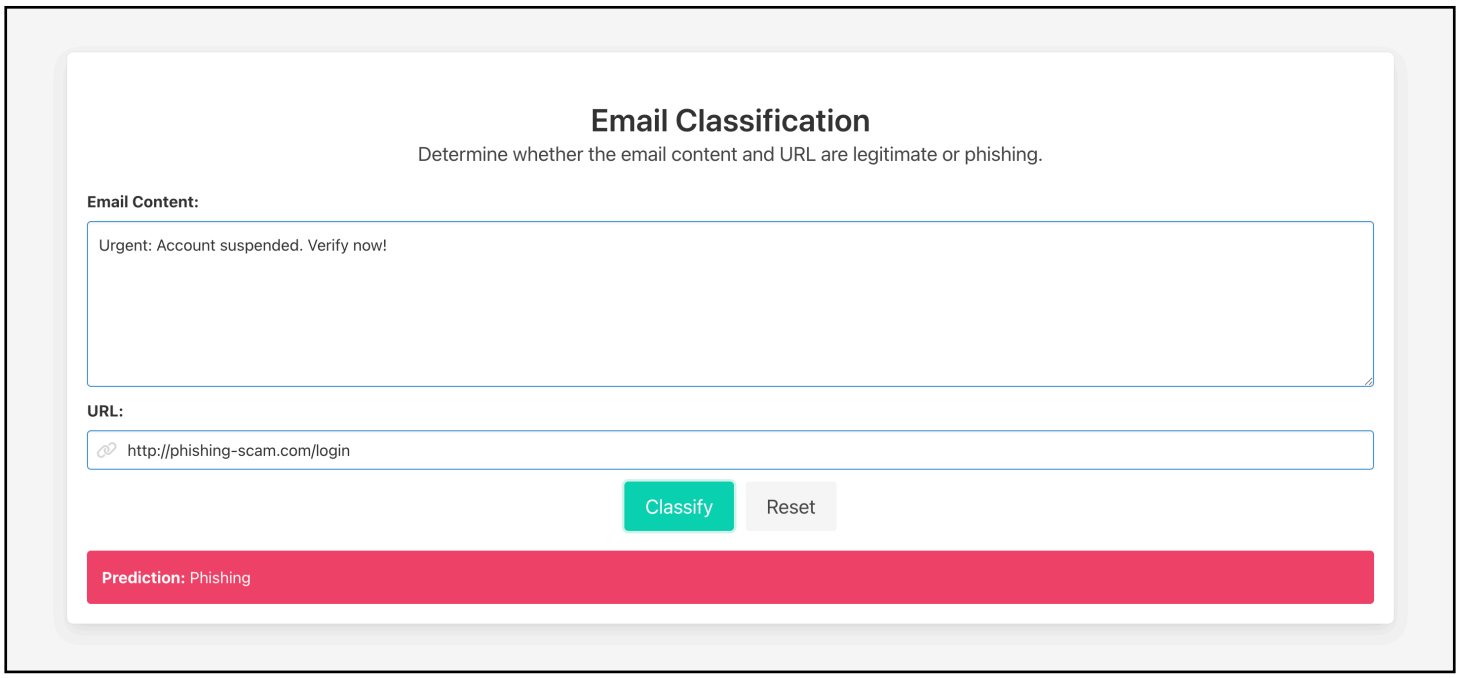
- **Data Collection**: The system combines three datasets, each containing phishing and legitimate emails.
- **Data Cleaning**: Null values are replaced, and the "message" and "URL" are merged into a unified text format.
- **Training**: We train the DistilBERT model on 80% of the data and reserve 20% for testing.
- **Evaluation**: We evaluate performance using multiple language datasets, including English, Spanish, German, French, and Italian.

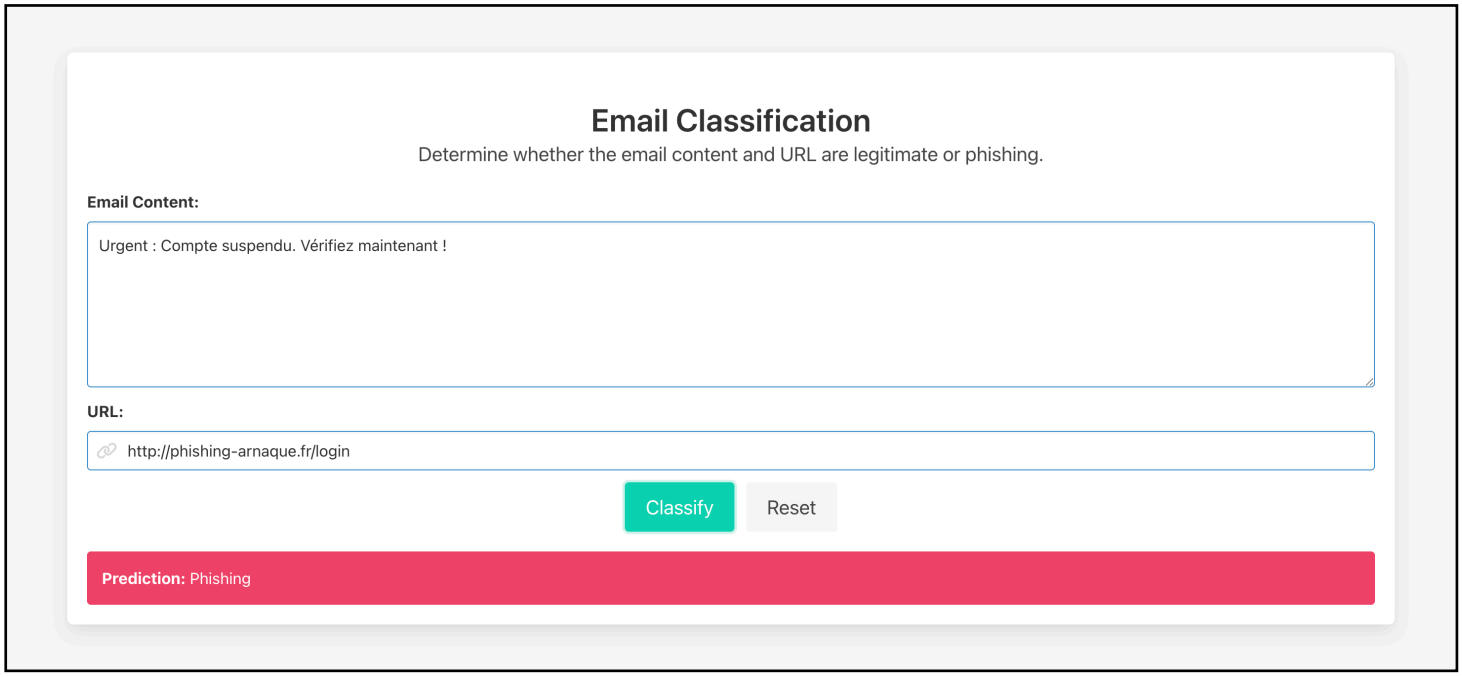
4.2 Results




Email Classification
Determine whether the email content and URL are legitimate or phishing.
Email Content:
Enter the email content here...
URL:
https://example.com
Classify
Reset


Email Classification
Determine whether the email content and URL are legitimate or phishing.
Email Content:
Dear customer, your order has been shipped.
URL:
https://www.legitstore.com/track
Classify
Reset
Prediction: Legitimate


Email Classification
Determine whether the email content and URL are legitimate or phishing.
Email Content:
Urgent: Account suspended. Verify now!
URL:
http://phishing-scam.com/login
Classify
Reset
Prediction: Phishing


Email Classification
Determine whether the email content and URL are legitimate or phishing.
Email Content:
Urgent : Compte suspendu. Vérifiez maintenant !
URL:
http://phishing-arnaque.fr/login
Classify
Reset
Prediction: Phishing

4.3 Analysis

The analysis reveals that the system performs best for English and slightly less for non-English languages. The reasons for this performance drop are related to linguistic differences and dataset diversity. However, the multilingual capability allows the system to classify phishing emails in various languages, significantly increasing its versatility and real-world applicability. [1 , 3]

5. Conclusions

5.1 Summary

This project successfully developed a multilingual phishing detection system using DistilBERT. By training on a large dataset and supporting multiple languages, our system achieves high detection accuracy, making it useful for global applications. Our results show improved detection accuracy, multilingual support, and generalization across different languages.

5.2 Contributions

- **Multilingual Phishing Detection**: The system classifies emails from multiple languages.
- **Robust Classifier**: It achieves high accuracy across multiple datasets.
- **Real-World Relevance**: The system addresses a practical need for cross-language phishing detection.[2]

5.3 Future Work

- **Expansion to Additional Languages**: Support for other languages like Chinese, Russian, and Hindi.
- **Dynamic URL Analysis**: Enhance URL classification to detect obfuscated URLs.
- **Continuous Learning**: Integrate continuous learning so the model can adapt to new phishing patterns and increase model accuracy.
- **Larger Datasets**: Use larger and more diverse datasets to enhance generalization.

References

1. [1]**"Phishing Website Detection Using Machine Learning"** Adarsh Mandadi; Saikiran Boppana; Vishnu Ravella; R Kavitha , URL - https://ieeexplore.ieee.org/document/9824801

2. [2]**"Devising and Detecting Phishing Emails Using Large Language Models"** Fredrik Heiding; Bruce Schneier; Arun Vishwanath; Jeremy Bernstein; Peter S. Park , URL - https://ieeexplore.ieee.org/document/10466545

3. [3]**"Detecting Phishing Attacks Using Natural Language Processing and Machine Learning"** Tianrui Peng; Ian Harris; Yuki Sawa , URL - https://ieeexplore.ieee.org/document/8334479