Détéction des adresses URL malveillantes avec Machine Learning

PAR :AZIZ AYA

ALAOUI HIBA

PROFESSEUR:M.YOUNES TABII

Introduction

La prolifération des cyberattaques, notamment à travers le phishing et la distribution de malwares, représente une menace croissante pour la cybersécurité. Dans ce contexte, la détection précoce et fiable des URLs malveillantes est devenue une priorité stratégique. Les cybercriminels exploitent souvent des techniques d’obfuscation complexes pour dissimuler leurs intentions malicieuses, rendant les méthodes classiques de filtrage insuffisantes.

Problématique

Les URLs malveillantes peuvent imiter des sites légitimes en adoptant des structures très similaires, ce qui rend leur détection difficile.

01. Intérêt du Machine Learning

02. Exploration et Préparation des Données

03. APPROCHE HYBRIDE

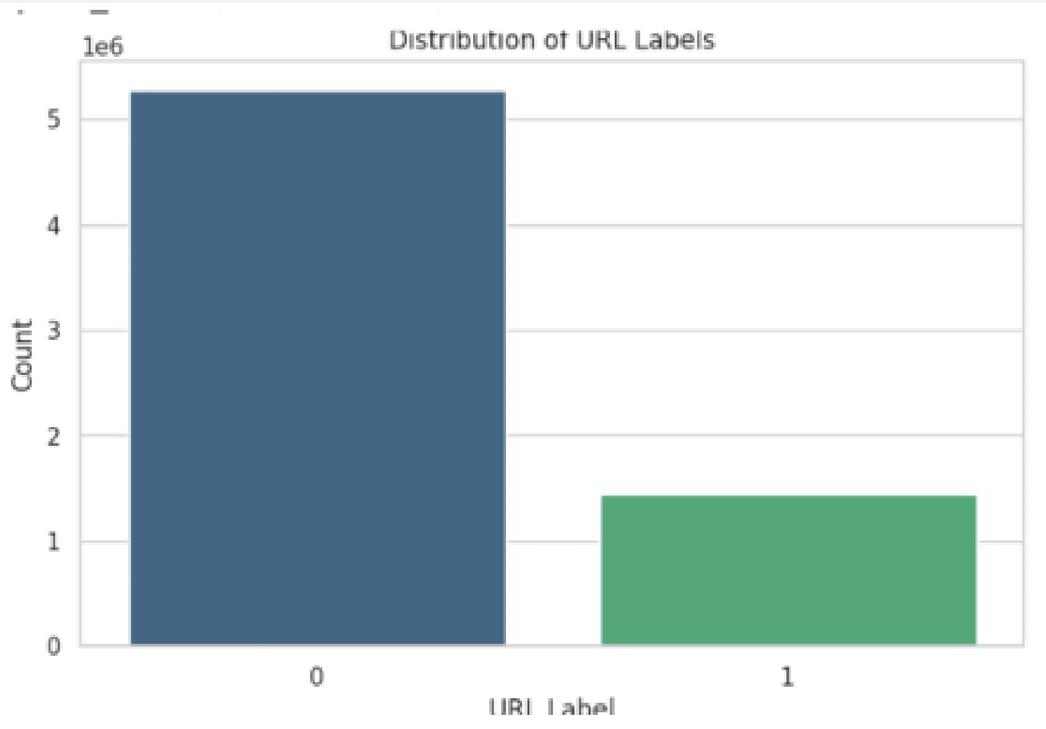
Exploration et Préparation des Données

Les premières étapes de "feature engineering" ont permis de visualiser et mieux comprendre la distribution des variables. Par exemple, certaines caractéristiques comme url_length, count_dash ou la présence d'IP ont montré des valeurs très élevées dans les cas d'URLs malveillantes.

```
data.head()
```

	url	label	source	url_has_login	url_has_client	url_has_server	url_has_admin	url_has_ip	url_isshorted	url
	irs-profilepaymentservice.com/home	1	phishtank	0	0	0	0	0	0	
	cpuggsukabumi.id	0	majestic_million	0	0	0	0	0	0	
	members.tripod.com/~don_rc/ring.htm	0	data_clean_test_mendel	0	0	0	0	0	0	
	optuswebmailadminprovider.weebly.com/	1	phishtank	0	0	0	1	0	0	
	topoz.com.pl	0	dmoz_harvard	0	0	0	0	0	0	

ows × 60 columns



Évaluation des Modèles

01	02	03
RANDOM FOREST	Régression Logistique	Modèle CNN 1D

RANDOM FOREST

```
Accuracy: 0.9319014393246988
Classification Report:
              precision    recall  f1-score   support

           0       0.94      0.98      0.96   1056595
           1       0.91      0.76      0.83    289175

    accuracy                           0.93   1345770
   macro avg       0.92      0.87      0.89   1345770
weighted avg       0.93      0.93      0.93   1345770
```

Ce modèle s’avère performant, notamment pour la détection des URLs légitimes. Le rappel plus faible pour la classe 1 indique que certains cas malveillants échappent à la détection.

Régression Logistique

```
Logistic Regression Accuracy: 0.8715746375680837
Logistic Regression Classification Report:
              precision    recall  f1-score   support

           0       0.89      0.96      0.92   1056595
           1       0.79      0.55      0.65    289175

    accuracy                           0.87   1345770
   macro avg       0.84      0.75      0.78   1345770
weighted avg       0.87      0.87      0.86   1345770
```

La régression logistique est moins performante que la forêt aléatoire, en particulier pour détecter les URLs malveillantes (rappel = 55%)

Modèle CNN 1D

Pour l’analyse séquentielle, nous nous sommes concentrés sur les colonnes url et label.

	url_isshorted	url_len	...	pdomain_count_hyphen	pdomain_count_atrate	\
0	0	34	...	0	0	
1	0	16	...	0	0	
2	0	35	...	0	0	
3	0	37	...	0	0	
4	0	12	...	0	0	

	pdomain_count_non_alphanum	pdomain_count_digit	tld_len	tld	\
0	0	0	3	com	
1	0	0	2	id	
2	0	0	3	com	
3	0	0	3	com	
4	0	0	6	com.pl	

	tld_is_sus	pdomain_min_distance	subdomain_len	subdomain_count_dot
0	0	17	0	0
1	1	10	0	0
2	0	2	7	0
3	0	3	25	0
4	0	3	0	0

[5 rows x 60 columns]

URLs converties en minuscules

```
--- Nettoyage et Préparation des URLs ---
Valeurs manquantes dans 'url' avant nettoyage : 0
Valeurs manquantes dans 'label' avant nettoyage : 0

URLs converties en minuscules.

Aperçu des URLs après traitement minimal :
                                  url  \
0     irs-profilepaymentservice.com/home
1                         cpuggsukabumi.id
2    members.tripod.com/~don_rc/ring.htm
3  optuswebmailadminprovider.weebly.com/
4                            topoz.com.pl

                           url_processed  label
0     irs-profilepaymentservice.com/home      1
1                         cpuggsukabumi.id      0
2    members.tripod.com/~don_rc/ring.htm      0
3  optuswebmailadminprovider.weebly.com/      1
4                            topoz.com.pl      0
```

Tokenisation Caractère par Caractère et Padding

```
Exemple 3:
  URL Originale (traitée)    : 'members.tripod.com/~don_rc/ring.htm' (longueur: 35)
  Séquence Paddée (premiers 30 tokens) : [11  2 11 20  2  8  9  4 10  8  7 15  3 14  4  6  3 11 17
46 14  3 12 42
  8  6 17  8  7 12]...
  URL Reconstruite (sans padding): 'members.tripod.com/~don_rc/ring.htm'

Exemple d'URL courte (index 1) : 'cpuggsukabumi.id'
  Sa séquence paddée : [ 6 15 16 19 19  9 16 22  5 20 16 11  7  4  7 14  0  0  0  0  0  0  0  0
  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
  0  0  0  0  0  0  0  0  0  0  0  0]
  Nombre de zéros (padding) dans cette séquence : 44

L'indice du token <UNK> est : 1
```

Division des Données

L'ensemble de données prétraitées a été divisé en 80\% pour l'entraînement (5,383,078 échantillons) et 20\% pour le test (1,345,770 échantillons), avec stratification pour maintenir la distribution des classes.

```
--- Division des Données en Ensembles d'Entraînement et de Test ---
Forme de X_train : (5383078, 60)
Forme de y_train : (5383078,)
Forme de X_test  : (1345770, 60)
Forme de y_test  : (1345770,)

Distribution des labels dans l'ensemble d'entraînement :
{0: 4226540, 1: 1156538}

Distribution des labels dans l'ensemble de test :
{0: 1056635, 1: 289135}
```

Architecture du Modèle Convolutional Neural Network 1D

Le modèle CNN 1D traite des séquences de 60 caractères via une couche d'Embedding (vocabulaire 157, dimension 100). Il applique ensuite une couche Conv1D (128 filtres, noyau 5) suivie d'un GlobalMaxPooling, d'une couche Dense (64 unités), d'un Dropout (0.5) et d'une sortie sigmoïde. Ce modèle est conçu pour capter efficacement les motifs locaux dans les séquences de texte.

```
--- Construction et Entraînement du Modèle CNN 1D ---
Utilisation de vocab_size: 157
Utilisation de MAX_SEQUENCE_LENGTH: 60
Utilisation de EMBEDDING_DIM: 100


/usr/local/lib/python3.11/dist-packages/keras/src/layers/core/embedding.py:90: UserWarning: Argume
nt `input_length` is deprecated. Just remove it.
  warnings.warn(
2025-05-07 15:36:01.827955: E external/local_xla/xla/stream_executor/cuda/cuda_driver.cc:152] fail
ed call to cuInit: INTERNAL: CUDA error: Failed call to cuInit: UNKNOWN ERROR (303)
```

CNN 1D

L'entraînement a été effectué sur 10 époques.

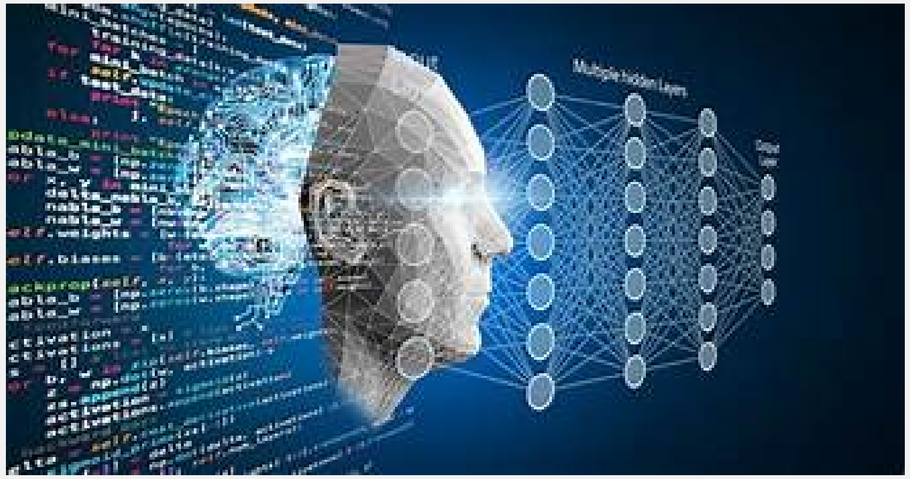
Une accuracy de 96.13\% sur l'ensemble de test est un résultat très encourageant, suggérant que le modèle CNN 1D a réussi à apprendre des motifs discriminants à partir des séquences de caractères pour distinguer les URLs malveillantes des URLs légitimes, et ce, malgré les contraintes.

```
Entraînement du modèle CNN terminé.

Évaluation préliminaire sur l'ensemble de test :
Loss sur le test (CNN) : 0.1268
Accuracy sur le test (CNN) : 0.9613
```

CONCLUSION

Ce projet a permis d’explorer différentes approches pour la détection automatique d’URLs malveillantes, en combinant des techniques classiques de machine learning avec des méthodes plus avancées de deep learning.



Remerciements