Artificial intelligence-based spam filtering using a neuro-linguistic approach with PyTorch framework

Balázs Tóth
John von Neumann Faculty of Informatics
Óbuda University
Budapest, Hungary
mwzx0d@stud.uni-obuda.hu

Valéria Póser
John von Neumann Faculty of Informatics
Óbuda University
Budapest, Hungary
poser.valeria@nik.uni-obuda.hu

Szandra Anna Laczi
John von Neumann Faculty of Informatics
Óbuda University
Budapest, Hungary
laczi.szandra@nik.uni-obuda.hu

***Abstract*—The paper presents the development of a PyTorch-based artificial intelligence spam filter based on neuro-linguistic approaches, i.e. natural language processing (NLP). A model has been developed to easily filter out messages that appear suspicious.**

***Keywords*—PyTorch, model, development, NLP, spam**

I. Introduction

Digital communication has become almost indispensable in people's daily lives. Unfortunately, spam is growing exponentially at the same time, challenging the systems that filter out unwanted content. It is critical that the software we use to send and receive message filters them reliably.

This paper shows how the PyTorch framework and natural language processing approaches can be used in concert to design an intelligent spam filtering system. It is able to recognise if the given data is general or suspicious message.

A. Typical patterns in email spam

- Phishing emails are designed to impersonate a trusted organisation and lure recipients into revealing sensitive information such as usernames, passwords or any valuable data.
- Malware emails send attachments or links to malicious software designed to infect the recipient's device with viruses, ransomware or other malicious programs.
- The advance payment scam, these emails promise large sums of money in exchange for a small advance.
- Fake lottery or prize scams, which falsely claim that recipients have won a lottery prize and often ask for personal details or payment to claim the alleged prize.
- Survey emails designed to collect personal data for fraudulent purposes.

These are the most common types of email spam but the list could be endless.

II. Target of the project

There can be found various of spam types, especially e-mail, SMS spamming, social media spamming and others. The project focuses on e-mail messages spam precisely in terms of data.

III. Why PyTorch?

In 2016, Facebook's AI research group, now known as Meta, took the lead in creating the PyTorch framework and generously shared it with the global community as an open-source resource. PyTorch has gained recognition for its outstanding qualities, being praised for its exceptional simplicity, impressive flexibility, and inherent efficiency. These remarkable features have solidified PyTorch's position as a fundamental and highly regarded tool in the fields of artificial intelligence and machine learning.

TABLE I: Comparing PyTorch with Keras

Category	PyTorch	Keras
API Level	Low	High
Datasets	Large datasets, high-performance	Smaller datasets
Debugging	Good debugging capabilities	Challenging
Pretrained models	Yes	Yes
Speed	Fast, high-performance	Slow, low-performance
Written in	Lua,	Python
Visualization	Limited	Depends on backend

Table 1 provides a detailed comparison between the PyTorch and Keras frameworks. [1] The key factor influencing the choice of PyTorch is its impressive performance and the ability to handle large datasets seamlessly. This pivotal decision is grounded in the framework's robust capabilities, making it a reliable choice for our study.

IV. Overview of spam statistics

As discussed previously about what are the common patterns in email spams, it is crucial to know about the statistics too. These statistical values shows how the companies and end users are not having the good amount of knowledge how to handle potential harmful messages in their inbox if that is not handled by the spam filter nor how their personal data is valuable for the hackers.

V. Methodology

A. Data collection and preprocessing

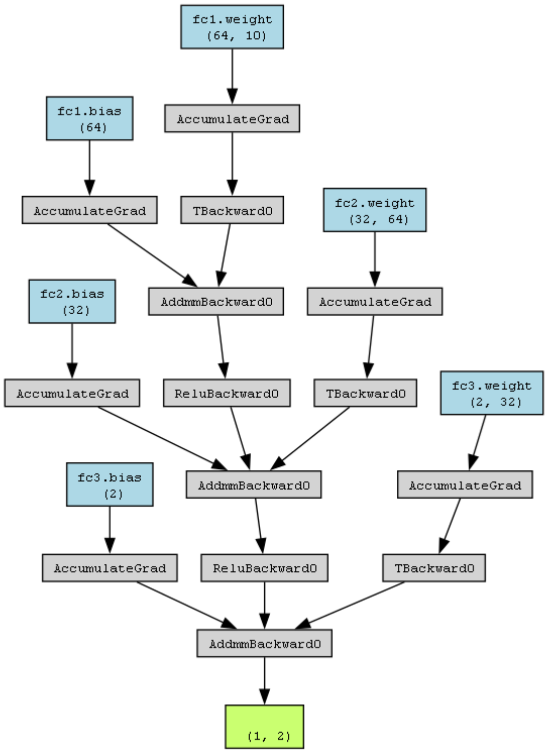
B. Model architecture

A modellnek meg kell adni, hogy milyen dimenziójú bemenettel kell számolnia. Belátható, hogy a modellnek három rétege van, melyek teljesen összekapcsolt rétegek, amik alkotják a neurális hálózatot és a `forward` függvényen keresztül halad át a bemeneti adatokon. Tegyük fel, hogy az `input_dim` értéke 100, így annak 100 elemű vektorral kell rendelkeznie, ami az első réteget illeti.

```
    class TextClassifier(nn.Module):
        def __init__(self, input_dim):
            super(TextClassifier, self).
                __init__()
            self.fc1 = nn.Linear(input_dim,
                64)
            self.fc2 = nn.Linear(64, 32)
            self.fc3 = nn.Linear(32, 2)

        def forward(self, x):
            x = torch.relu(self.fc1(x))
            x = torch.relu(self.fc2(x))
            x = self.fc3(x)
            return x
```

Listing 1: Modell Python kód tartalma



References

[1] *PyTorch vs Tensorflow vs Keras*, https://www.datacamp.com/tutorial/pytorch-vs-tensorflow-vs-keras, Last viewed; 20:59, 6th of December, 2023

[2] Basemah Alshemali and Jugal Kalita, *Improving the Reliability of Deep Neural Networks in NLP: A Review*, Knowledge-Based Systems, 2020.

[3] Emmanuel Gbenga Dada, Joseph Stephen Bassi, Haruna Chiroma, Shafi'i Muhammad Abdulhamid, Adebayo Olusola Adetunmbi, Opeyemi Emmanuel Ajibuwa, *Machine learning for email spam filtering: review, approaches and open research problems*, Heliyon, 2019.

[4] *Spam statistics: a deep dive into unwanted emails*, https://eftsure.com/statistics/spam-statistics/, Last viewed; 18:34, 6th of December, 2023