URL文字列の深層学習による詐称サイト識別

便利なサービスがインターネットで提供されるようになった一方で、悪用される事例も増えています。複雑化、巨大化したシステムをすべて人手で見守ることは困難になっており、すでに様々な自動化の仕組みが運用されています。ここ数年、セキュリティ分野では深層学習の活用に注目が集まっています。経験と知識が重要となる分野において、深層学習による補助が実現できれば、より多くの人材が高度な運用に関わることが可能となり、結果的に安全なサービスが実現できるでしょう。本稿では、深層学習をサイバー攻撃の防御に活用する試みを紹介します。

2.1 Webの登場と詐称サイトの戦い

欧州原子核研究機構(CERN)のフェローとして活動していたTim Bernars-Leeが最初のワールドワイドウェブ(WWW)サーバ、CERN HTTPdを公開してからおよそ30年が経ちます。ハイパーテキストをインターネットで実現したこの仕組みは、同氏らによって同時期に提案されたHTTP (Hypertext Transfer Protocol)とURL (Uniform Resource Locator)との抜群の組み合わせにより、瞬く間に世界の情報を繋ぐ技術になりました。1980年代から90年代は、TCP/IPを実装したBSD UNIXが学術機関を中心に爆発的に普及した時期でもあり、世界中のコンピュータが相互に接続される土壌が育ちつつあったことも無関係ではないでしょう。更に、米国立スーパーコンピュータ応用研究所(NCSA)がGUIを備えたWebブラウザMosaicを公開したことにより、コンピュータ技術者でなくても簡単に世界の情報にアクセスできるようになります。Web技術の進化は今なお続いており、毎日どこかで新しいサービスが立ち上がっています。

あらゆる技術に共通して当てはまることですが、世界を良くすることができる技術は同時に世界を悪くすることもできます。様々なサービスがWebで提供されるようになるにつれ、それを利用した詐称行為が登場します。よく見られるのは、有名なオンラインサービスサイトや銀行サイトに似せたWebページを準備し、偽メールなどを通じて利用者を誘導、個人情報やパスワードなどを盗み取るというものです。もちろん、詐称行為自体はWeb以前から存在するものですが、迷惑メールと同様、電子化によって低コストでより多くの人を狙うことができるようになりました。情報化は、表の世界と裏の世界、分け隔てなく恩恵を与えてくれたということです。

詐称サイトへのアクセスから利用者を守ることは、近年のネットワークサービス運用者にとって重要な課題の1つです。ISPであれば、接続を提供している顧客に詐称サイトブロックなどのサービスを提供している場合も多いでしょう。もしあなたが組織の情報システム担当者なら、自組織内の利用者に対して何らかのセキュリティ対策ソフトウェアの導入を推進したりしているかもしれません。現在広く利用されている詐称サイト防御技術は、基本的にはブラックリストを用いた手法です。ただし、容易に想像できるように、単なるブラックリストでは広大なWeb空間を網羅することは困難です。研究者たちは、より効率的に悪性サイトを判別する方法を模索してきました。例えば、既存の詐称サイトのドメイン名に類似した文字列を機械的に推測して、少ないブラックリストからより多くの悪性ドメイン名の候補を作り出す試みがありました[*1]。また、単なるリストの範疇を超え、ドメイン名が登録された時期、Googleでの検索順位などを参考にし、登録されて間もないドメイン名や、順位の低いドメイン名の信頼度を低く見積もる手法などもありました[*2]。実際にアクセスされたページの内容を透過プロキシなどで解析し、サイトが悪性かどうかを判断する技術なども提案されています[*3]。更に、近年の深層学習の発展に伴い、セキュリティ分野への深層学習応用も進んできています。

*1 P. Prakash, M. Kumar, R. R. Kompella, and M. Gupta, "PhishNet: Predictive blacklisting to detect phishing attacks," in 2010 Proceedings IEEE INFOCOM, ser. INFOCOM, 2010, pp. 1–5.
*2 S. Garera, N. Provos, M. Chew, and A. D. Rubin, "A framework for detection and measurement of phishing attacks," in Proceedings of the 2007 ACM Workshop on Recurring Malcode, ser. WORM '07. New York, NY, USA: ACM, November 2007, pp. 1–8.
*3 Y. Zhang, J. I. Hong, and L. F. Cranor, "CANTINA: A content-based approach to detecting phishing web sites," in Proceedings of the 16th international conference on World Wide Web, ser. WWW '07. ACM, May 2007, pp. 639–648.

© Internet Initiative Japan Inc.

2.2 URL文字列に隠された意味

詐称サイトとの戦いに終わりはありません。何らかの防御技術が考え出されれば、それを回避する仕組みが生み出されていきます。それでも、より安全なインターネットの利用のため、新しい防御技術を検討していくことが重要です。

詐称サイトの判定率でいえば、実際にアクセス先のコンテンツを確認して判断するプロキシ型のものが有利です。ただし、実サイトへアクセスするという行為は危険な場合もあります。処理負荷やプライバシーなどの課題もあり、実際のコンテンツへアクセスしない手法も多く提案されています。最も単純な手法が、URLそのものだけを見て判断する方法です。この場合、URLを構成する文字列に詐称サイトか否かを判断するための何らかの意味が含まれているのかどうかが問題となります。

この疑問に対する正確な答えを持っている人はいません。ただ、過去の研究を調べてみると、意味があると考えた人もいたことが分かります。例えば、よく知られた考え方の1つとして、ドメイン名は発音可能な文字列である、というものがあります。ドメイン名は何か現実の物やサービスに関連したものであることが多いため、自然言語や名称を元にした文字列が多く、必然的に人間が発音できるものになりやすいというものです。マルウェアの中には、機械的に生成されたドメイン名(Domain Generation Algorithm、DGA)を利用しているものもあり、多くの場合これらの名前は発音不可能な文字列で構成されることになります。この区別ができれば、通常のアクセスと疑わしいアクセスを区別できるのではないかという考え方です。

また、異常に多いサブドメイン(たくさんのドットが含まれているホスト名)や、異常に深いパス(たくさんのスラッシュが含まれるURL)が使われる場合は悪性の場合が多いという考え方もあります。経験則に基づいた様々な条件を検討し、それらの組み合わせを用いて悪性かどうかを判断するのが、この分野では一般的な手法です。

そしてその最前線に位置する技術として、深層学習を活用したURL判定技術が検討されています。

2.3 深層学習の復活

深層学習が一般にも注目されるようになったのは5〜6年程前でしょうか。深層学習で使われるニューラルネットワーク自体は古典的といっても良いくらい昔からある考え方です。ただ、多層のニューラルネットワークを用いる深層学習の仕組みは、その計算量の多さや、適切に学習させることの技術的困難さから、長らく実用化が困難であると考えられてきました。ところが、2010年代に入ると画像認識の分野で目覚ましい成果を挙げる手法が考案され、一躍注目を浴びるようになります。諸説あると思いますが、2012年に開催された大規模画像認識競技会(ILSVRC、ImageNet Large Scale Visual Recognition Challenge)でAlex Krizhevskyらによって実証された深層学習を用いた画像認識システム[*4]が、今に通じる深層学習の流れの始まりとみなされることが多いようです。それまで25%程度だった誤り率を一気に10%も向上させ、深層学習が現実的な場面に応用できる可能性を示したのです。以降、主に画像や音声認識の分野で広く深層学習が使われるようになり、言語の翻訳、文書の分類、果ては囲碁に至るまで様々な分野に応用されていきました。

ネットワークについても、主にセキュリティ分野で深層学習を用いた技術が次々と提案されている状態です。この記事では、私たちが提案するURL文字列から詐称サイトを判定する手法[*5]を紹介しますが、当然この提案が世界初の試みというわけではあ

*4 A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," Advances in neural information processing systems, 2012.
*5 K. Shima, D. Miyamoto, H. Abe, T. Ishihara, K. Okada, and Y. Sekiya, "Classification of URL bitstreams using Bag of Bytes," in Proceedings of First International Workshop on Network Intelligence(NI2018), 2018.

© Internet Initiative Japan Inc.

りませんし、またこれからもたくさんの研究者・技術者がより良い方法を提案していくことになると思います。ネットワーク、特にインターネットのような自律分散環境では、永久に動作する何かを作り出すことは困難です。時代と共にシステムもデータも変遷していき、そのすべての情報を把握することは不可能です。私たちは常に世界の一部しか見えておらず、見えている情報ですら瞬く間に古くなっていくからです。

深層学習は万能の仕組みというわけではありません。巷でいわれているような人間を超える知能がそこから生まれるのかどうか、まだ私たちには分かりませんが、今できることは分かっています。

深層学習は機械学習の一手法であり、ある入力ベクトルに対して、一定の演算を行い、別のベクトルを出力します。これが分類問題や識別問題に使われることになります。画像認識の例でいえば、猫の画像（をベクトルの形に変換したもの）を入力して、それが猫かどうかを0/1で出力するなどです。「一定の演算」の内容を決めるため、深層学習では大量のデータを必要とします。猫の例でいえば、大量の猫の画像と、猫以外の画像などです。これらは学習データと呼ばれ、あらかじめ答えが分かっているものを使う場合を教師あり学習、そうでない場合を教師なし学習と呼びます（その中間もあります）。深層学習は、こういった分類問題で大きな成果を収めています。

2.4 URLのベクトル化

さて、いよいよURLの分類の話に進んでいきたいと思います。目的は、与えられたURLが問題のない通常のサイトなのか、詐称サイトなのかを判定することです。深層学習の手法を利用するため、まずはURLを深層学習の入力に利用できるベクトルの形に変換しなければなりません。

深層学習以前の機械学習では、このベクトル定義（特徴量の定義）が重要な工程でした。対象データを区別するために必要十分な情報を如何に事前に定義できるかが性能に大きく影響するためです。先にも述べたとおり、URL分類では、発音可能かどうか、ドットやスラッシュの数、アルファベット、記号、数字の比率、文字の出現場所、n-gram文字列の出現頻度など、様々な要素がURLを区別する特徴として検討され、検証されてきました。やみくもに特徴量を増やしてしまうと計算時間に影響してくるため、従来の機械学習の手法では対象データの深い知識を持った専門家が、既存のデータをあの手この手で解析しながら役に立ちそうな特徴を厳選していました。

これに対し、深層学習では、大量のデータを用いて学習させることによって、自分で特徴量を見つけてくれるといわれています。実際にはそう単純ではなく、注意深いデータの前処理が最終的な結果に影響してくることも多いと思いますが、特徴量定義の負担をある程度物量でカバーしてくれることも事実でしょう。

今回、URLを分類するにあたり、既存の特徴量は使わないこととします。その代わり、単純な変換処理を定義してURL文字列を固定長のベクトルに変換します。確かに、URLの分類に関しては先人の知恵を活用することもできたと思いますが、今後他のデータセットを対象としていく場合、常に有用な特徴量が定義できるとは限りません。もし、単純な前処理と大量の学習データでURLの識別が可能であると分かれば、他のデータセットでも同様の戦略を取ることができるかもしれないという淡い目論見もあります。

今回私たちが用いたベクトル化の手順は次のようになります。

1. URLを文字単位に分割する
2. 各文字をASCIIコードの16進表記に変換する
3. ホスト部、パス部それぞれの先頭から1バイトずつ、4ビットずつシフトしながら、出現する値を列挙する
4. ホスト部、パス部それぞれに出現した値（0x00から0xFF）の個数を数え、256次元のベクトルにする
5. ホスト部、パス部から計算された256次元のベクトルを連結し、512次元のベクトルにする
6. ベクトルを正規化する

© Internet Initiative Japan Inc.

手順1から3を図-1に示します。続く手順4で値の個数を数えます。図-1に示した例では、ホスト部の値は次の通りです。

0x16(1個)、0x2E(3個)、0x42(1個)、0x61(1個)、0x64(1個)、0x69(2個)、0x6A(2個)、0x70(1個)、0x72(1個)、0x77(5個)、0x96(2個)、0xA2(1個)、0xA7(1個)、0xE6(3個)

ホスト部のベクトルをVとし、次元iの値をv_i(i:出現した値)とすると、$v_{0x16} = 1$、$v_{0x2E} = 3$、$v_{0x42} = 1$、...、$v_{0xE6} = 3$となります。出現しなかった次元の値は0です。元のURLが何であれ、URLの長さがいくつであれ、必ず512次元のベクトルに変換することができます。ただし、このままではURL文字列が長くなるとベクトルのサイズが増大してしまうため、手順6で正規化しておきます。こうして変換した512次元のベクトルを、「URL特徴ベクトル」と定義します。

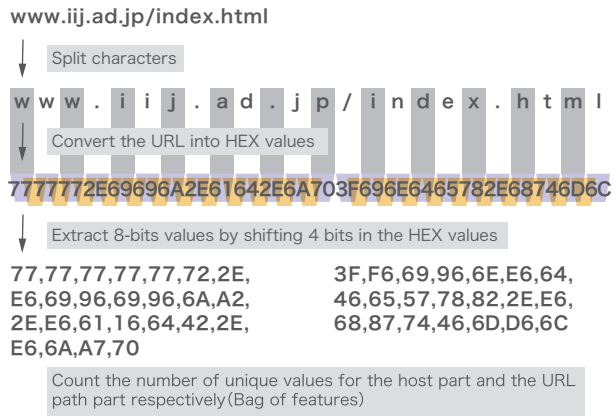

図-1 URLのベクトル化

2.5 ニューラルネットワーク設計

URLを固定長のベクトルに変換する準備は整いました。次は、そのURL特徴ベクトルをどう学習させるのかを決めなければなりません。今回の試みでは、シンプルな3層のニューラルネットワークを用いました。深層学習というにはいささか浅いネットワークですが、この種の手法に効果があるのかどうかを確認するためには十分だと考えます。

図-2に今回利用したニューラルネットワークのトポロジを示します。このトポロジをどこかで見たことがある方もいるかもしれません。この3層全結合トポロジは、株式会社Preferred Networksが開発しているオープンソース深層学習ライブラリChainer(https://chainer.org/)のサンプルとして登場し、MNISTデータセット(http://yann.lecun.com/exdb/mnist/)

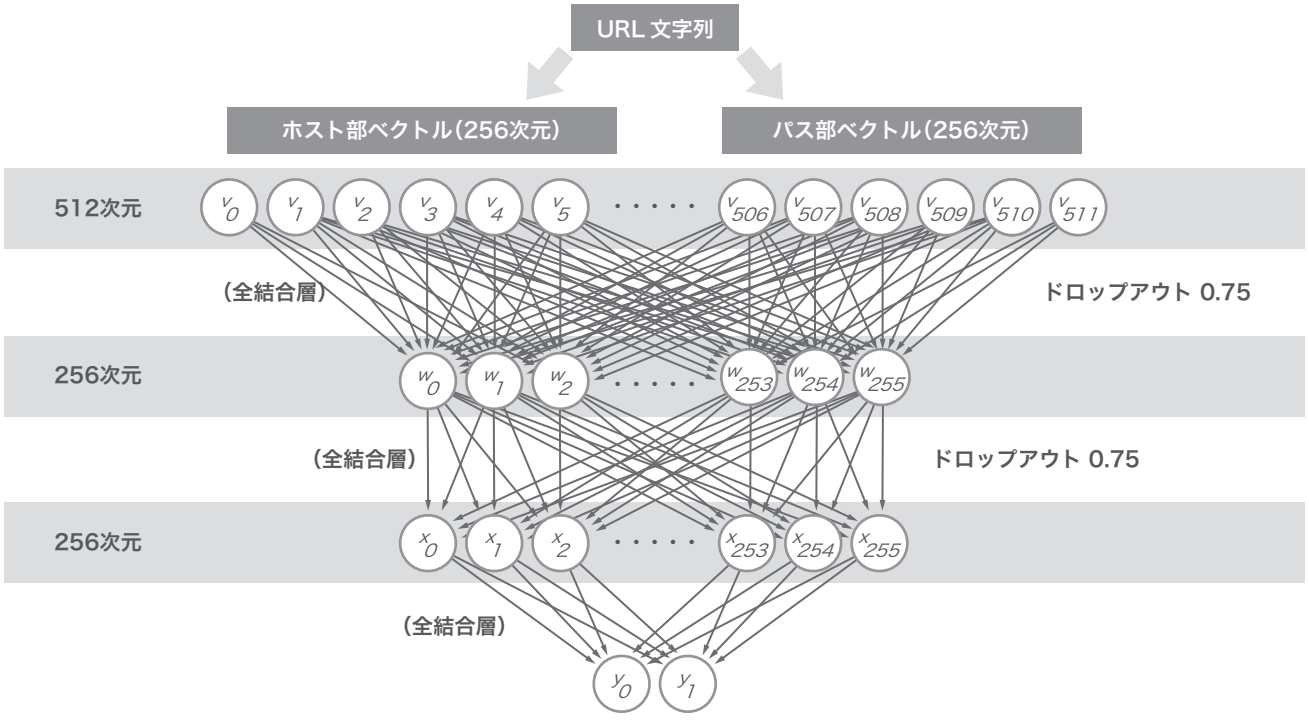

図-2 ニューラルネットワークトポロジ

© Internet Initiative Japan Inc.

を利用した手書き数字の認識モデルとして使われています。これを元に、次の2つの変更を加えました。

1. 入出力次元数：MNISTサンプルの場合、入力画像の縦横サイズが28x28なので、784次元の入力、出力が0から9までの数字なので10次元の出力となっていました。今回は、入力が512次元のURL特徴ベクトル、出力はURLが詐称サイトかどうかを示す0か1の2次元に設定しています。
2. ドロップアウト率：MNISTサンプルでは過学習を抑えるための仕組みであるドロップアウトを利用していませんが、今回のデータに関しては激しい過学習が観測されたため、高めのドロップアウトを設定しています。

今回は、提案手法の検証にもChainerを用いています。Chainerで定義したニューラルネットワークモデルを表-1に示します。

2.6 データソースの選定

データ構造とニューラルネットワークモデルが揃ったので、実際のデータを使って検証してみます。インターネットのようなダイナミックな環境では次の2つが大きな課題になります。

1. データの正確性：MNISTのようなデータは、すべてのデータが事前に検証され、あらかじめ正しいラベル（MNISTの場合だと、手書きの数字画像と、その画像が表す数字の値）が準備されています。ところが、インターネット上で観測・収集されるデータに正確にラベルを付けることは困難です。もし、間違った情報で学習を進めてしまえば、当然間違った答えを推測するモデルが育ってしまいます。
2. 網羅性：学習に使うデータが真に一般的なデータなのかどうかを示すことはできません。偏ったデータで学習してしまうと、それ以外のパターンが出現したときに対応できなくなります。これは手書き数字認識などでも同じことがいえるのですが、0から9までの数字という、ある程度問題が限定された場合と、インターネット上のURL文字列のような無限に広がる空間では、自ずと対応できる範囲に差が出てしまいます。

こういった問題があることを前提としつつ、なるべく正確な、網羅性の高いデータを準備することが大切です。今回、詐称サイトのデータとしてPhishTank.com（https://www.phishtank.com/）に登録されているアクティブな詐称サイトのデータを利用します。全世界すべての詐称サイトが登録されているわけ

```
from chainer import Chain
import chainer.functions as F
import chainer.links as L
class Model(Chain):
    def __init__(self):
        super(Model, self).__init__()
        with self.init_scope():
            self.l1 = L.Linear(None, 256)
            self.l2 = L.Linear(None, 256)
            self.l3 = L.Linear(None, 2)
    def __call__(self, x):
        h1 = F.dropout(F.relu(self.l1(x)),
                       ratio=0.75)
        h2 = F.dropout(F.relu(self.l2(h1)),
                       ratio=0.75)
        y = self.l3(h2)
        return y
```

表-1 Chainerによるニューラルネットワークモデル

© Internet Initiative Japan Inc.

ではないため、網羅性は保証できませんが、詐称サイトかどうかの判断は人の目による投票ベースの処理が入るためある程度信頼できます。より難しいのは「詐称サイトではない」通常のサイトのデータです。検証では、ある研究組織のアクセスログから、前述のPhishTankに登録されたものを取り除いたものを詐称サイトではないURLと定義して利用していますが、より網羅性を高めるためには異なる種類のアクセスログでの追試が必要になるでしょう。

2.7 深層学習の適用可能性

前節で準備した2種類のデータソースから、それぞれおよそ26,000個ずつ、ランダムにURLを抜き出します。このうち80%を学習に使いました。残った20%で分類精度(Accuracy)を確かめたところ、94%のデータに対して正しく詐称URLとそうでない通常のURLを区別することができました。この結果が良いのか悪いのか、意見の分かれるところだと思います。過去に提案されている分類手法では、この値よりも良い値を実現するものもあります。それらの手法は、単なる文字列だけではなく、他の情報(例えばWhoisの情報やGoogleの検索順位の情報など)を用いている場合もあるので単純な比較はできません。また、URL文字列のみを用い、私たちと同様に深層学習を活用した試みもあり[*6]、こちらも、今回の私たちの結果よりも高い分類精度を実現しています。ただ、彼らの提案したニューラルネットワークモデルを独自に実装し、私たちのデータセットを用いて追試してみたところ、論文で示されているような高い値を出すことはできませんでした。学習に用いたデータセットによって、同じニューラルネットワークモデルでも精度に大きく差が出るということです。

全世界すべてのデータを入手できない以上、ある程度の偏りのある学習になってしまうことは避けられません。最近ではビッグデータという言葉を聞かなくなりましたが、広範囲のデータを大量に持っている組織が有利になる状況は深層学習の世界でも継続、むしろ以前よりもその差が拡大するのではないかと思います。

2.8 まとめ

本稿では、詐称URLの判断に深層学習を応用する試みを紹介しました。簡単なニューラルネットワークを用いた検証だったにもかかわらず、94%の精度でURLの分類が可能となりました。同時に、データ収集の困難さ、データを持つことの強みについても再確認できました。

ネットワークデータへの深層学習応用はこれからどんどん進歩していくと思います。コンピュータの能力も向上し、比較的簡単に深層学習を利用することもできるようになりました。今後も新しい技術を取り込みつつ、より安全なインターネットの実現を目指していきたいと思います。

《 謝辞 》
本研究は、JST、CREST、JPMJCR1783の支援を受けたものです。

執筆者:
島 慶一(しま けいいち)
IIJ 技術研究所 主幹研究員。

*6 J. Saxe and K. Berlin, "eXpose: A character-level convolutional neural network with embeddings for detecting malicious URLs, file paths and registry keys," CoRR, vol. abs/1702.08568, February 2017.

© Internet Initiative Japan Inc.