Texception: A Character/Word-Level Deep Learning Model For Phishing URL Detection

Farid Tajaddodianfar*, Jack W. Stokes**, Arun Gururajan**

* Currently at Amazon Inc., Seattle WA, USA. Farid performed this research while being at Microsoft
** Microsoft Corp., Redmond WA, USA
URL string to reveal phishing content

- URL string can be indicative of maliciousness of its host
  - Specific patterns in URL string can be used over and over by bad actors
- There are signals other than URL (e.g. HTML content, page DOM, etc)
  - Need significant level of transformation before being used
- URL can be used as a relatively strong and available feature to detect phishing attacks
Convolution on Character Embedding
CNN on Character embeddings

- Assign a vector to represent each character in the alphabet of observed characters
- CNN with fixed filter size $N$ is analogous to extracting $N$-grams

1D Convolutional output channel size

Character embedding dimension $M$

Filter = 3
Dilation = 1
Stride = 1
CNN on Character embeddings

- CNN with dilation resembles extracting skip-grams
- Previous methods use “fixed” filter size
- That effectively limits model’s flexibility to “fixed” n-grams. E.g. only 3-grams
Convolution on Word Embedding
Word Embedding Concepts

- First proposed to capture semantic similarity in NLP
- Given a corpus of natural language text, randomly assign an N-dimensional vector to each word
- During training, move vectors closer if they appear in the same context
- Trained in unsupervised settings (Word2Vec, FastText, GloVe)

\[ \cos \alpha = 0.50 \]

"More than a \textit{good} movie; it's really \textit{exceptional}"
FastText Word embedding on URL

- Tokenize URL based on special characters [./;$:] and treat it as a sentence!
- Treat each special character as a word too
- Train FastText word embedding on the whole training data
- Use the trained embeddings in the downstream classification task
CNN on Word embeddings

- Each word is represented by the FastText embedding vector
- CNN with fixed filter size N is analogous to extracting word N-grams
- Using fixed filter size limits the model flexibility
- Why not more?
Texception Block
&
Texception Model
Texception Block

- Multiple parallel Convolutional layers inside
- Each with Batch Normalization, Max Pooling, ReLu
- Features extracted from each path are concatenated and outputted

Sequence of i-dimensional vectors

Conv1D, F = 2, D = 1
i = 32, o = 64

BN, Max pool

ReLu

Conv1D, F = 3, D = 1
i = 32, o = 64

BN, Max pool

ReLu

Conv1D, F = 6, D = 1
i = 32, o = 64

BN, Max pool

ReLu

Sequence of o-dimensional vectors

F = filter size
D = Dilation
i = input channel
o = output channel
w = Max pool window
Texception Model

- Multiple branches to extract information from text
- Can grow wider or deeper at each branch
- Data scientist controls bottleneck of information flow
- Provides sufficient flexibility to learn from text
<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Characters Branch number of blocks</td>
<td>1</td>
</tr>
<tr>
<td>block filters</td>
<td>[2,3,4,5]</td>
</tr>
<tr>
<td>Adaptive MaxPool output</td>
<td>32,32</td>
</tr>
<tr>
<td>maximum characters</td>
<td>1000</td>
</tr>
<tr>
<td>Words Branch embedding dimension</td>
<td>32</td>
</tr>
<tr>
<td>number of blocks</td>
<td>1</td>
</tr>
<tr>
<td>block filters</td>
<td>[1,3,5]</td>
</tr>
<tr>
<td>Adaptive MaxPool output</td>
<td>32,16</td>
</tr>
<tr>
<td>maximum words</td>
<td>50</td>
</tr>
<tr>
<td>FastText Model minimum words to include</td>
<td>50</td>
</tr>
<tr>
<td>vocabulary size</td>
<td>120000</td>
</tr>
<tr>
<td>window size</td>
<td>7</td>
</tr>
<tr>
<td>n-grams</td>
<td>2-6</td>
</tr>
<tr>
<td>embedding dimension</td>
<td>32</td>
</tr>
<tr>
<td>epochs trained</td>
<td>30</td>
</tr>
</tbody>
</table>

Table 1. Hyperparameters used for experimentation.
Experimentation

**Train set:**
- 2 weeks of traffic labeled by proprietary labeling systems
- Clean class down sampled to keep positive class ratio at 5%
- 1.7M samples in training set
- 20% used as validation set

**Test set:**
- No down sampling to keep the distribution like the real production data
- Positive class ratio at 0.01%
- 20M samples
Results

Superior performance at very high precision
• Are word embeddings useful?

• Is it better to use pre-trained embeddings or just start with random weights and let the model learn the embeddings itself?

• If using pre-trained word weights, is it better to freeze the word weights or let the model adjust them during training?

Good to use word embeddings
Even better to randomly initialize them!
Results

Good to use word embeddings
Even better to randomly initialize them!
Moving beyond URL

Texception is applicable where data is in the form of text string

- File name
- CTPH
- Command lines

Not limited to security data

- Texception is a generic model for text classification
Thank You!