Phishing Detection Using Neural Network

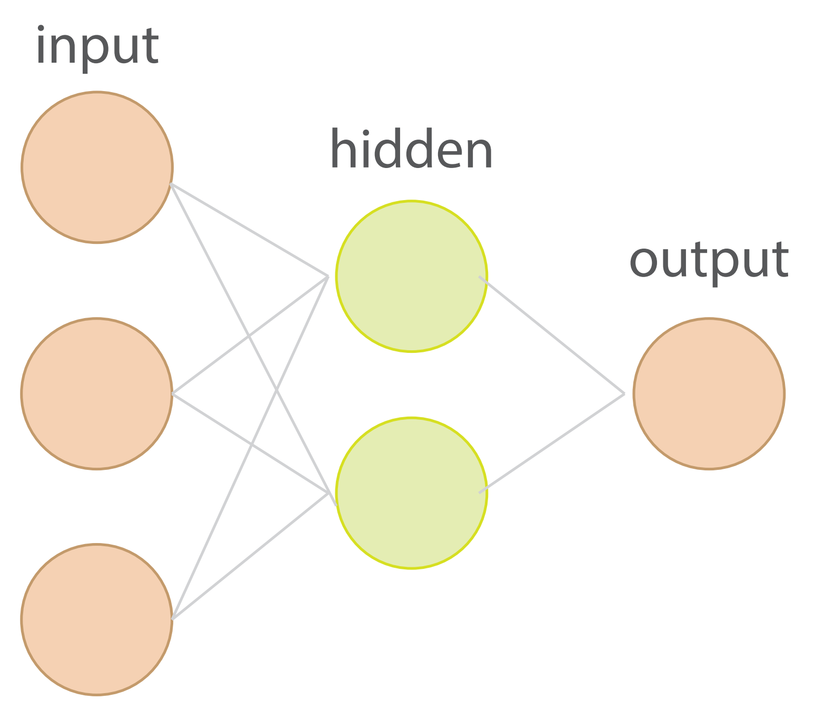
Ningxia Zhang, Yongqing Yuan

Problem and Motivation

Recently there has been a phishing email attempting to capture SUNet ID and password circulating in the Stanford community. As the majority of phishing emails are formatted to appear from a legitimate source, a large percentage of email users are unable to recognize a phishing attack. With the increasing severity of this problem, many efforts have been devoted to tackling it, including applying Machine Learning methods. But little studies have been done with neural networks. In this project, we try to identify phishing emails through feedforward neural networks using a set of selected features pertaining to structural and content characteristics.

Methods

We have designed a set of features we regard as characteristic of phishing emails:



Structural Features: #body parts, #alternative parts

Link Features: #links, # IP-based links, #deceptive links, #links behind an image, maximum #dots in a link, whether there is a link that contains words: click, here, login, update

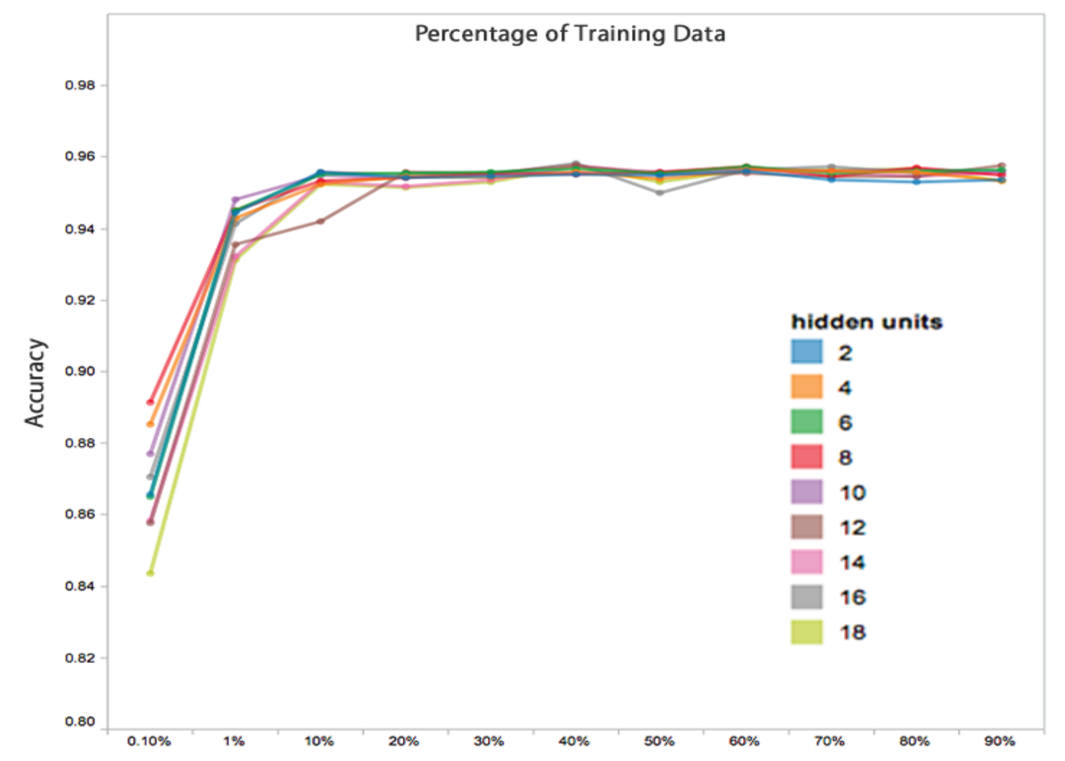
Element Features: whether it is in HTML, whether it contains JavaScript, whether it contains <Form> tag

Word List Features: whether the words or stems listed below appear in the email body: account, update, confirm, verify, secur, notif, log, click, inconvenien

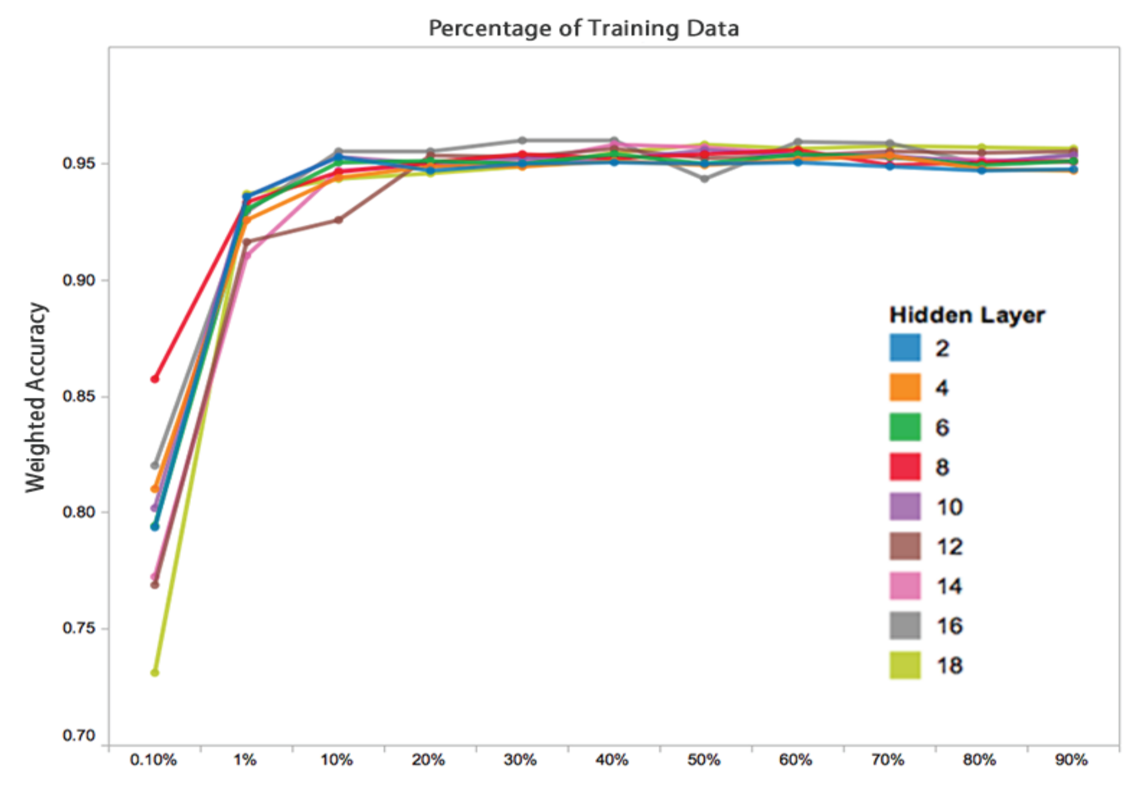
After feature extraction, each email is represented by one vector, which is then normalized so that each feature contributes the same. We use Encog to construct NNs and run hold-out cross validations with different configurations of NN. We also compare the performance of NNs with other major Machine Learnig algorithms.

Results

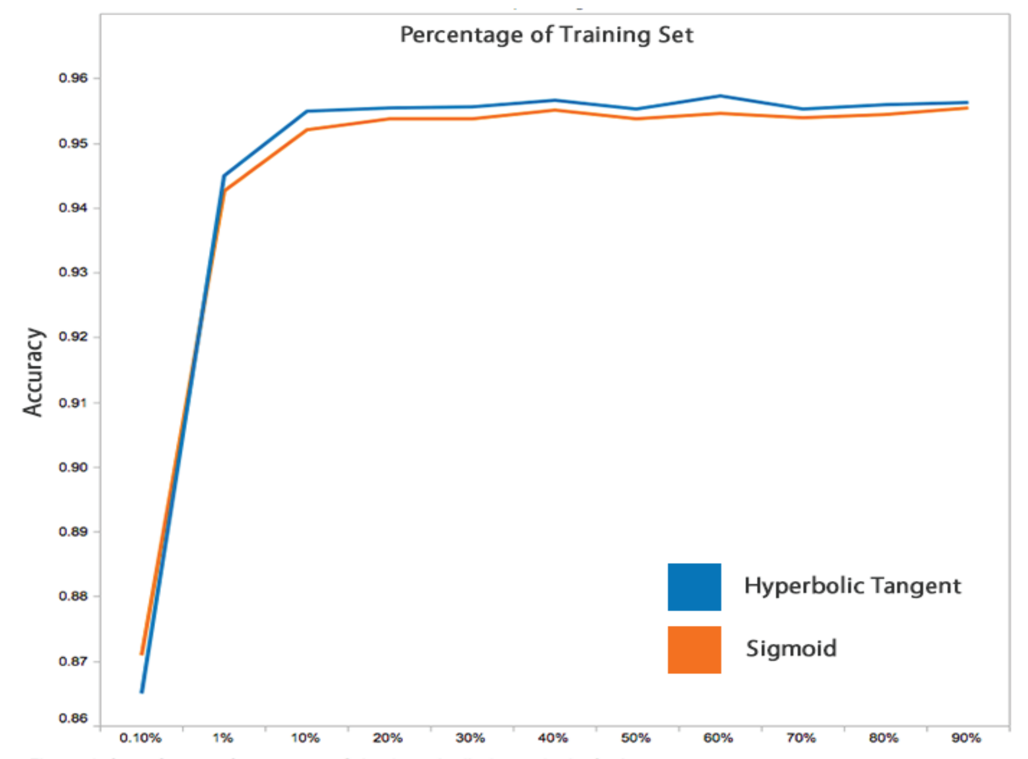
Accuracy

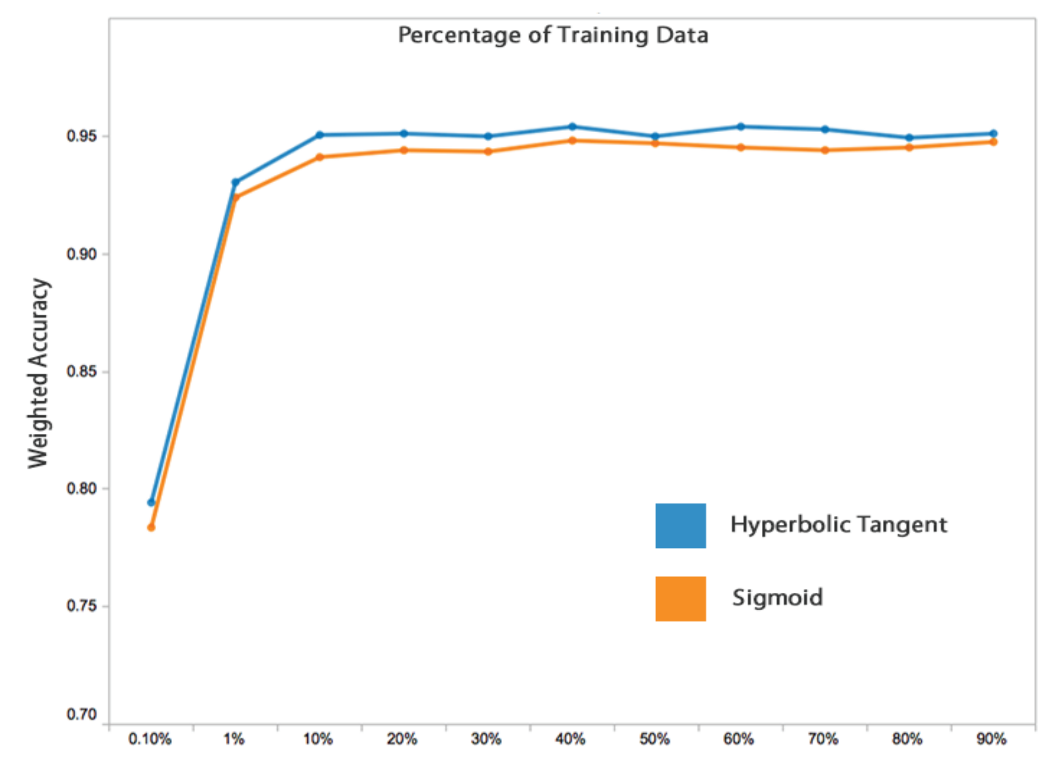


Weighted Accuracy (Penalty for False Postive)

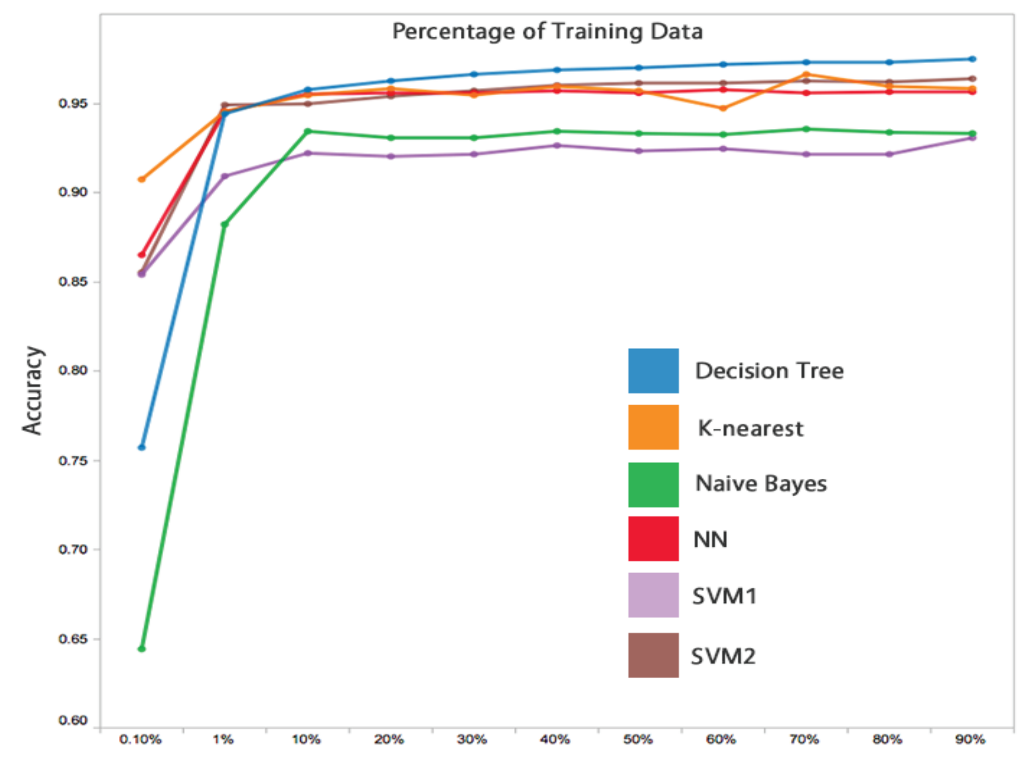


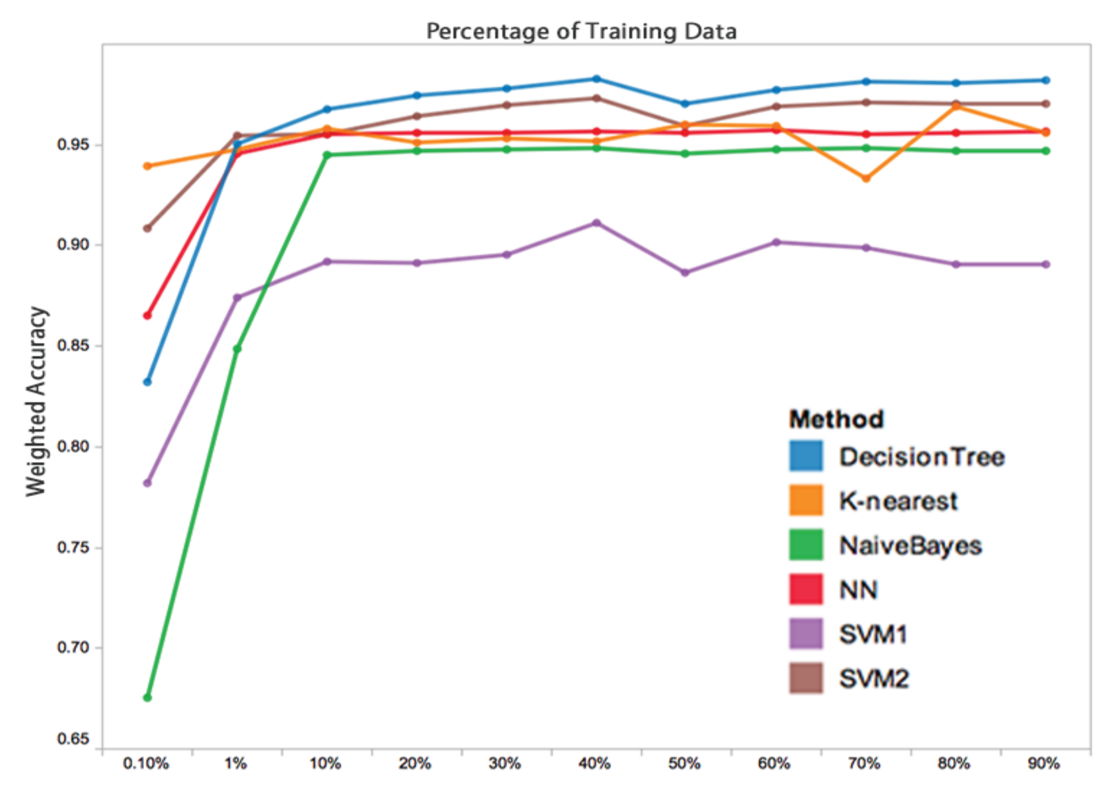
More hidden units tend to overfit the data, while fewer tend to underfit. When the training set is large enough, the number of hidden units does not affect performance.





Hyperbolic tangent performs slightly better than sigmoid as activation function.





Decision tree has the best overall performance, while it falls short on small training sets compared to NN and k-nearest. Generally, most algorithms can reach accuracy of 95%, meaning the feature set has captured the essential characteristics.