Insert here your thesis' task.

Czech Technical University in Prague
Faculty of Information Technology
Department of Software Engineering

Bachelor's thesis

Phishing Email Detection based on Entity Recognition

Šimon Let

Supervisor: Ing. Vít Listík

15th May 2017

Acknowledgements

I would like to thank my supervisor Ing. Vít Listík, without his assistance and guidance this thesis would have not be accomplished.

Then I would like to thank Ing. Michal Bukovský along with his whole Email.cz team in Seznam.cz for letting me work with them and use their data.

My thanks also goes to RNDr. Jana Straková at. el for developing Nametag, Named entity recognition solution, I used in this thesis, and answering my questions.

Finally I want to thank my family and friends for their support trough my whole studies.

Declaration

I hereby declare that the presented thesis is my own work and that I have cited all sources of information in accordance with the Guideline for adhering to ethical principles when elaborating an academic final thesis.

I acknowledge that my thesis is subject to the rights and obligations stipulated by the Act No. 121/2000 Coll., the Copyright Act, as amended, in particular that the Czech Technical University in Prague has the right to conclude a license agreement on the utilization of this thesis as school work under the provisions of Article 60(1) of the Act.

In Prague on 15th May 2017 .....

Czech Technical University in Prague
Faculty of Information Technology

Citation of this thesis

Let, Šimon. *Phishing Email Detection based on Entity Recognition.* Bachelor's thesis. Czech Technical University in Prague, Faculty of Information Technology, 2017.

Abstrakt

Phishing je vážný problém, který způsobuje nezanedbatelné finanční ztráty mnoha organizacím po celém světě. Současné metody pro detekci phishingu pro český jazyk jsou nedostatečné. Rozpoznávaní entit (NER) bylo úspěšně použito pro detekci phishing emailů v anglickém i jiných jazycích.

Cílem této práce je použít NER pro určení cíle potenciálního phishing emailu a použít tuto informaci k detekci phishingu.

Vyvinuli jsme rešení schopné detekovat cíl potenciálního phishing emailu. Naše řešení funguje dobře pro významné phishing cíle jako třeba finanční a technické společnosti. Dokázali jsme, že detekce phishing cíle je zdrojem cenných informací o phishing emailu a může být použita pro detekci phishingu.

Náše řešení pro rozpoznání phishingu dosáhlo 86.9% úspěšnosti při použití pouze jedné vlastnosti vygenerované pomocí detekce phishing cíle. Při použití bežně používaných vlastností naše řešení dosáhlo 97.7% úspěšnosti. Přidání vlastností založených na detekci phishing cíle způsobilo malé zlepšení. Rešení dosáhlo 98% úspěšnosti.

Pro rozpoznávání entit jsme použili Nametag, současně nepřekonaný NER pro český jazyk. Natrénovali jsme vlastní modely pro Nametag, použili jsme původní a vlastní přidaná data. NER jsme použili pro rozpoznání entit v emailu. Použili jsme vlastní řešení pro mapování jmen organizací na domény pomocí dat z ARES a Firmy.cz. Namapované domény jsme použili pro určení cíle potenciálního phishing útoku.

Model pro Nametag byl otestován proti datům pro úkol z CoNLL2003 zabývající se rozpoznáváním entit. Detekce phishing cíle byla otestována pomocí 2628 emailů z dat od Email.cz. Nakonec jsme naše řešení pro detekci phishingu otestovali pomocí datasetu obsahujícího 3744 emailů z veřejných i neveřejných zdrojů.

Klíčová slova Phishing, strojové učení, email, rozpoznavání entit, Český jazyk

Abstract

Phishing is a serious problem that causes a significant financial loss to many companies wolrdwide. Current phishing detection methods for Czech language are not satisfactory. Named entity recognition (NER) was successfully used for phishing email detection in English and other languages.

The goal of this thesis is to use NER to determine target of the potential phishing email and use this information to detect phishing emails. We have developed a solution that can detect target of the potential phishing email. Our solution works well for major phishing targets such as financial and technology organizations. We have proved that target detection gives a significant amount of information about the email and can be used to detect phishing.

Our recognizer achieved 86.9% accuracy when using only one feature generated using target detection. When using only commonly used features the recognizer achieved 97.7% accuracy. Adding target based features gave us minor improvement and achieved 98% accuracy.

For Named entity recognition we use Nametag, the state of the art NER for Czech language. We have trained our own models for Nametag using both original and custom data. NER is used to recognize organizations in email. Custom solution is used to map organizations to domains using data from ARES and Firmy.cz. Mapped domains are used to determine target of the potential phishing email. Target and collected historical data about domains is used to generate features for random forest classifier. Random forest classifier is used to detect whether or not the email is phishing attack.

Nametag model was tested against CoNLL2003 shared task on Named entity recognition. Target detection solution was tested using 2628 emails from Email.cz data. Finally the phishing detection recognizer was tested using dataset containing 3744 emails from both public and private sources.

Keywords Phishing, machine learning, email, Named entity recognition, Czech language

Contents

List of Figures

Introduction

Phishing is a fraudulent technique with the intention of obtaining confidential user data. Attackers usually create a similar looking website to a legitimate one. Therefore uneducated users may have trouble spotting the difference and can type their credentials into form on malicious website thinking they are dealing with a legitimate one. In such case attacker gains access to their sensitive data such as login information or credit card credentials. Attackers send hypertext links leading to their malicious website using email.

Emails containing such links are called phishing emails. Phishing emails are hard to detect because they have varying form and content [49]. Attackers are using various techniques such as obfuscation and tokenization to fight phishing detection. [49]. Phishing emails are not usually sent in such volumes as spam. Therefore traditional anti-spam solutions perform poorly at phishing email detection. Various groups of people has successfully managed to use machine learning for phishing email detection.

The goal of this thesis is to research possibilities of using Named entity recognition combined with machine learning and develop filter capable of detecting phishing emails using machine learning approach.

Motivation

There has been over 90 thousand phishing attacks reported just in the December 2016. [29] Phishing emails have caused huge financial loss to various companies worldwide. [13]

Detection of phishing email is an uneasy task that cannot be entirely solved using conventional programming methods [29]. Many groups of people tried to solve the problem using machine learning with varying results [30] [28] [26] [41].

Goal

The goal of this thesis is to develop a phishing detection solution that is based on Named entity recognition and proves that Named entity recognition can be used to gain a significant amount of information about the potential phishing email.

First we need to research state of the art Named entity recognition and phishing detection solutions. Then we have to choose Named entity recognition solution that is most fit for our purpose. We will use the chosen solution to extract information from the emails and generate features for phishing email detection.

Next step is to develop baseline phishing email detection solution based on previous research and evaluate the performance of this solution. Then use features generated using Named entity recognition as features for developed phishing email detection solution.

Compare the performance of baseline solution and solution with added features. Describe the benefits of using features generated using Named entity recognition.

Evaluate and summarize both the process of research an development and the achieved results. Describe the most important accomplishments of this thesis and suggest possible improvements for future work.

Chapter 1

State-of-the-art

1.1 Phishing email types

Phishing has varying form because attackers are trying to disguise phishing emails so it does not get detected by phishing and spam detectors. This section will give a quick overview of different types of phishing emails.

1.1.1 Link-based

Link-based phishing emails are one of the most common forms of phishing. Attackers put hypertext link leading to malicious website into the email. This type of phishing email will be main focus of this thesis.

1.1.2 Text-based

Text-based phishing emails are such emails where attackers put forms requesting sensitive user data directly into email itself, effectively getting rid of hypertext link which may help the phishing email go undetected. They usually create the form using either HTML form tags [8] or just plain text. We will detect text-based phishing emails in this thesis.

1.1.3 Image-based

Image-based phishing emails are a result of attackers effort to obfuscate the content of their phishing emails with the aim of making it more difficult to detect their attacks. Such emails use images instead of HTML or text. Attackers create an image containing text making it harder for spam filters to discover the content of the email because image processing and on screen character recognition are complicated tasks. Which makes it hard to detect whether email is phishing or not. Such phishing emails are relatively rare and their detection will not be focus of this work.

1.1.4 Attachment-based

Attachment-based phishing mail takes advantage of email attachments and their diverse formats. Email attachment represents another layer of protection from phishing detection. Many possible formats attachment and complexity of used attachment format makes detection more difficult. We will not focus on detection of this type of phishing email.

1.2 Phishing detection approaches

Phishing emails evolve and change with time. Because of attackers changing the phishing emails they are sending, and colossal financial loss caused by phishing, need for more advanced phishing detection techniques is raising. Scientists have been working on different phishing detection methods. Variety of ways how to detect phishing emails has been developed. This section will list and explain some of approaches used for phishing email detection. Most common phishing detection approaches can be divided into two groups.

1.2.1 Blacklist-based

Generally blacklist is a list of entities, people or other subjects that are considered untrustworthy. Blacklist-based methods use database of known phishing domains, links or email addresses and check potential phishing emails against the database. The main weakness of blacklist-based phishing detection is the blacklist itself. You need to keep blacklist up to date or it becomes worthless and even if we keep the blacklist updated we will never be able to detect zero-day phishing emails. Unfortunately domains used for phishing are often relatively new or hacked legitimate ones [25] which lowers recall 1.6.2 of blacklist based methods. Blacklisting has high precision 1.6.1 because legitimate websites are unlikely to get into the blacklist. In conclusion blacklisting is simple phishing detection method with high precision but low recall.

1.2.2 Machine learning approach

Machine learning methods use phishing and non-phishing emails to create a classifier for phishing email detection. As well as blacklist-based approach machine learning approach has certain drawbacks. The process of training the classifier is time-intensive and requires the dataset containing both phishing and non-phishing emails. The dataset greatly influences the performance of classifier. Also the implementation of machine learning algorithms is far more complicated than implementation of blacklist-based methods.

1.3 Phishtank

One example of black-list based solution is Phishtank. Phishtank is a community website that gathers information about confirmed phishing attacks. The information about attacks is publicly available and can be downloaded in many formats. Anyone can sign up and either submit new phishing attacks or verify existing ones. Furthermore Phishtank offers a free API for both non-commercial and commercial use. [14]

1.4 Machine learning

Machine learning is a subfield of computer science that gives computers ability to learn without being explicitly programmed. [33] Rather than describing the whole subject of machine learning in detail, this section will briefly describe essential parts of machine learning related of thesis to phishing email detection. Machine learning tasks are usually classified into three general categories.

1.4.1 Supervised learning

Supervised learning maps input data to desired output. It is trained using annotated data. Supervised learning the main focus of this thesis. Common tasks include classification tasks such as Computer vision, Speech recognition, Optical character recognition, Information retrieval, Document classification and Phishing email detection.

1.4.2 Unsupervised learning

Unsupervised learning annotates or finds structure in data. No training and no annotated training data is required. Some minor parts of this thesis will focus on this category. Unsupervised learning includes certain NLP tasks such as word embeddings [23] and Brown clustering.

1.4.3 Reinforcement learning

Reinforcement learning is very different from the two previous categories. It involves a computer interacting with dynamic environment. There is no annotated training data. Instead of training data the system is learning trough reward and punishment. This category includes task such as vehicle driving.

1.5 Datasets

Datasets are collections of data. Datasets are crucial part of all machine learning methods used for phishing email detection. Training dataset has huge impact on performance of the resulting phishing email detector. Ideally

the dataset should match the real life data otherwise the resulting model may perform poorly when used with real life data despite the fact that it performs well on held-out data from the dataset.

1.5.1 SpamAssassin

SpamAssassin [20] is a high performing [30] open source spam detection platform. Spam assassin project has gathered a huge amount of emails.

SpamAssassin email corpus is one of most commonly used publicly available datasets for English language [21]. SpamAssassin email corpus consists of both spam and ham messages. Messages were either collected from publicly available sources or sent to author of corpus with permission for public use. SpamAssassin email corpus has been used in many works focused on phishing email detection [48] [30] [44]

1.5.2 Email.cz dataset

Thanks to Email.cz we have access to moderate dataset of emails. Dataset includes both phishing and non-phishing emails from various sources. There is 5472 phishing and about 10 thousand non-phishing emails in the dataset.

Phishing emails come from three different sources. First part is a public dataset that was downloaded from [10] [1] Second part consists of phishing emails reported by Email.cz users. Last little part was obtained from hoax.cz.

Non-phishing emails is a sample of emails collected from traffic.

To our knowledge there are no publicly available phishing datasets for Czech language. All Czech emails used in this thesis were taken from Email.cz dataset.

1.6 Evaluation methods

Evaluation methods are used for establishing how accurately does given machine learning system perform. When looking at two independent systems we need a well-defined measure that will allow us to compare them.

1.6.1 Precision

Precision represents how many of tagged samples are relevant. Concerning phishing email detection precision is really important because low precision means that a lot of legitimate emails are being tagged as phishing. Precision only measures the quality of the system not the quantity therefore it can't be used for comparison of two systems by itself.

[1] Unfortunatelly the link is not working anymore. We do not know if or where it can be downloaded now.

1.6.2 Recall

Recall represents how many of relevant samples were tagged. Concerning phishing email detection low recall means that a lot of phishing emails are not detected. Recall only measures quantity of the system not the quality that's why it can't be used for serious comparison of two systems by itself.

1.6.3 F-measure

F-measure is a combined measure that incorporates precision and recall. It can be used for measuring performance and comparing machine learning systems. F-measure is defined as a harmonic mean of precision and recall.

$$F\_measure = 2 \cdot \frac{precision \cdot recall}{precision + recall}$$

1.7 Machine learning in phishing detection

This section will give a brief overview of chosen machine learning methods that has been already successfully used for phishing email detection in previous works. Tolan at el. [48] has compared a performance of variety of ML methods when used for phishing email detection.

1.7.1 Linear regression

Linear regression is one of the simplest machine learning methods. It uses simple mathematical functions with precisely tuned parameters to model the phenomenon. Input of the function are decimal numbers representing chosen features of tested email. Output of the function is one decimal number that is used to determine whether the tested email is phishing or not.

According to [24] Linear regression has the best results out of all compared methods.

1.7.2 Support vector machine

Support vector machine (SVM) is one of the most commonly used machine learning method for phishing detection. The principle behind this method is following. Each email is converted to vector in multi dimensional space. The goal is to find a hyperplane that separates phishing and non-phishing emails.

Support vector machines has been successfully used to detect phishing emails by many researchers including [28], [26], [38] and [48].

1.7.3 Decision trees and Random forest

Decision trees learning are a simple machine learning method. Feature vector extracted from the email travels from the root to the leaves of the tree. The

path trough the tree is determined by conditions in the nodes of the tree. Result is assigned to each leaf.

Random forest is a collection of decision trees. Each tree classifies the email independently and the final result is determined by the voting system.

Decision trees are prone to overfitting the dataset. Random forest should prevent overfitting by training each of the trees with randomly selected of features. Random forests were successfully used in [30] and [37].

1.7.4 Neural networks

Neural networks are popular machine learning method commonly used for various tasks. This method is inspired by actual neural networks. Neural networks consist of neurons that send and accept signals from one another. Bonds between the neurons have different weight. Neurons are usually organized into layers.

This method is not widely used for phishing email detection. It was successfully used alongside with other methods in [24] and [26].

1.8 Features

Generally speaking features are pieces of information extracted from the raw data presented to recognizer. When considering phishing detection only data that is guaranteed to be present is the email itself. Websites linked from inside of the email may not be working or links might not be present at all. Therefore many features used for phishing detection in state of the art systems rely only on data available directly from the email itself.

Choosing features carefully is on of the most important tasks when designing any machine learning system [48]. Poor choice of features may drastically reduce performance of phishing detection recognizer. There are numerous works that use dozens of features and compare their effect on performance of phishing email detection system such as [48], [26], [30], [38] and [36].

1.9 Features for phishing email detection

In 2010 Tolan has extracted 40 features from emails from three different datasets to determine the features that bring the most valuable information for spam and phishing email detection. [48] Some features were carefully selected form previous papers other features were added by author himself. Features that had brought most significant information gain in all three datasets are following.

1.9.1 Body richness

Body richness is defined as the total number of words divided by the number of characters in the email body. Body richness formula is shown in formula. Body richness was originally proposed by Chandrasekaran et al. [28] and it was in the top three most significant features in all three tested datasets in [48].

$$bodyRichness = \frac{numberOfWords}{numberOfCharacters}$$

1.9.2 Subject richness

Subject richness is defined as number of words divided by the number of characters in the email subject. Subject richness formula is shown in formula. This feature was one of the top three most significant features in all three tested datasets in [48].

$$subjectRichness = \frac{numberOfWords}{numberOfCharacters}$$

1.9.3 Number of links

This feature represents number of links in the body of the email. It was previously used in following papers [30] [27]. This feature has been also used to detect Czech phishing emails in [37].

1.9.4 Number of external links

Number of external links represents number of links in the body of the email with target outside of the email body. This feature was previously used in [27].

1.9.5 Number of domains

Number of domains measures total number of domains in all URLs in the email. This feature was used in [30] and [37].

1.9.6 Number of characters

This feature represents number of characters in the body of the email. It was used in [28] and [37].

1.9.7 Body HTML

Body HTML is a binary feature that is true when HTML is present in the email. Previously used in [30] and [37].

1.9.8 Maximum number of periods

This feature measures maximum number of periods present in a link in the email. This feature was previously used in [30] [27].

1.9.9 Number of words

Number of words represents total number of words in the body of the email. Used in [28] and also used in [37] to detect Czech phishing emails.

1.9.10 Advanced features

Some researchers took experimenting with features even further by extracting advanced features from the email. [44]. Shyni et al. [44] has developed a successful phishing email detector using features such as Named entity recognition, Topic modeling and Image processing. It is clear that using advanced features can improve the performance of the phishing email detector.

1.10 Named entity recognition

Named entity recognition (NER) is a data extraction task that belongs into group of tasks called Natural language processing. The input is unannotated stream of text and the goal of Named entity recognition is to recognize words or chunks of words that are proper names (entities), such as people, organizations and locations. In addition to entities Named entity recognition implementations sometimes extract other information, such as time, date, hypertext links and email addresses.

State of the art implementations reach close to human performance with 93.39 F-measure in MUC-7 [9] task and 90.80 F-measure on dataset from CoNLL-2003 shared task [42] [3].

1.10.1 CoNLL-2003 shared task

Shared task of Conference on Computational Natural Language Learning in 2003 [2] was focused on language-independent Named entity recognition. CoNLL-2003 shared task provided contestants with annotated training and testing data as well as unannotated data and lists of names. One of the challenges for contestants was to incorporate additional provided data into their solutions.

CoNLL-2003 shared task dataset contains four entity types: person, organization, location and misc for entities that do not belong to previous three categories. Entities are non-embedded, non-overlapping and annotated with exactly one label. [46] There are data available for English an German languages. [3]

Despite being almost 13 years old CoNLL-2003 shared task is still being used as standard measurement for comparison of Named entity recognition systems performance. [46] [45] [42] Initial baseline performance for CoNNL-2003 shared task is 59.61 F-measure. Over dozen of teams have participated in the shared task. The best result achieved was 88.76 F-measure [31]. Current state of the art Named entity recognition performance on CoNLL-2003 shared task is 90.80 F-measure. [42]

1.10.2 Czech Named Entity Corpus

For Czech language the Czech Named Entity Corpus is available. CNEC 2.0 is a large corpus of manually annotated Czech sentences. Corpus contains 46 types of named entities. These fine-grained classes are grouped into 7 supertypes: numbers in addresses, geographic items, institutions, media names, artifact names, personal names and time expressions. Entities can be labeled with one or more classes and can be embedded. [4]

Czech Named Entity Corpus was originally released together with a Named entity recognition system that achieved 62 F-measure on fine-grained classes and 68 F-measure on supertypes. In 2011 it was outperformed by recognizer that achieved score of 72.94 F-measure on supertypes. [34] Currently the best performing named entity recognizer on Czech Named Entity Corpus is Nametag with 82.82 F-measure. [46]

1.11 Nametag

Nametag is state of the art implementation of Named entity recognition for Czech language. Nametag also achieved score of 89.16 F-measure on CoNLL-2003 shared task dataset which is close to performance of the current state of the art Named entity recognition system for English language.

The recognizer is based on Maximum Entropy Markov Model and a Viterbi [32] algorithm decodes an optimal sequence labeling using probabilities estimated by a maximum entropy classifier. The classification features utilize morphological analysis, two-stage prediction, word clustering and gazetteers. [46]

1.11.1 Overview

Nametag classifier is based on Maximum Entropy Markov Models which are models derived from Hidden Markov Models and use Maximum entropy principle. In couple of following sections we are going to take a closer look at both Hidden Markov Models and Maximum Entropy Markov Models as well as the Maximum entropy principle. For decoding Nametag uses custom implementation of Viterbi algorithm. [32]

Nametag uses standard set of classification features: form, lemma, tag, chunk (only English) of current word and surrounding words in window of two words on each side, orthographic features (capitalization, punctuation, lowercase and uppercase form of the word), suffixes and prefixes of length 4 and regular expressions identifying possible year, date and time (in Czech). [46]

Nametag also uses global features by repeating the prediction process and using the prediction from the first run in the second one.

1.11.2 Markov Models

Markov models are statistical models used to predict series of random events. It is assumed that each event depends only on the current state and not on the previous events, this assumption is called Markov property.

Markov model can be represented by finite state machine where each transition between states has a fixed probability. Each state has a set of outputs with their emission probabilities assigned. Each state produces an output chosen from the assigned set.

Markov chain is a simplest form of Markov Model. It is a model where all states and outputs are observable.

1.11.3 Hidden Markov Models

Hidden Markov Model is a Markov Model where state of the system are not fully observable. It is often used for various natural language processing problems. Hidden Markov Model can represented by finite state machine very much like Markov chain except the system state is not fully observable. Outputs emitted by unobservable states are visible. Therefore knowing the outputs can be used to estimate the probable sequence of states.

Hidden Markov Models have been successfully used for various natural language processing tasks.

1.11.4 Maximum entropy principle

Entropy is a measure of the randomness of a system. Therefore a system with maximum entropy is the one with the greatest randomness, under the given constraints.

Model with maximum entropy under given constraints is the one that makes the least assumptions about unknown information. Models with higher entropy have more uniform distribution of probabilities. Because of least amount of assumed information, model with the highest entropy is more generalizing and less biased.

Entropy of the model is measured using previously chosen boolean features. Instead of just emitting the correct sequence of outputs the chosen model has

to produce the previously selected features in the same proportions they are present in the training data.

1.11.5 Maximum Entropy Markov Models

Maximum Entropy Markov Models are a variation on Hidden Markov Models. There are many advantages of using Maximum Entropy Markov Models over Hidden Markov Models. These models can use more descriptive features such as capitalization, punctuation, part-of-speech tags, word endings and formating. Richer representation of words can be used for prediction. Long distance features can be used.

Hidden Markov Models are generative models. That means that they generate the observable output sequence based on hidden states. In contrast Maximum Entropy Markov Models are discriminative models. The output sequence is given and the states are conditioned on the given output sequence.

1.11.6 Viterbi algorithm

Viterbi algorithm is a dynamic programming algorithm that is used to find most probable path trough Markov Model given the sequence of outputs. Viterbi algorithm became a standard way to find the most likely sequence of states in many machine learning systems such as Hidden Markov Models, Maximum Entropy Markov Models and Conditional random fields.

1.11.7 Nametag classifier

Classifier predicts all possible sequences of tags and positions inside of entity. Number of classes is depends on the dataset. CoNLL2003 dataset has 4 classes. CNEC has 42 classes and 7 supertypes.

Position of a word in an entity is described using BILOU scheme. B means beginning of an multi-word entity, I is inside of multi-word entity, L is last word of multi-word entity, O means outside of entity and U is unit entity - a single word entity.

Using classes from dataset and BILOU schema classifier assigns tags such as B-LOC, I-ORG and O to words.

1.11.8 Decoding

Probabilities estimated by Maximum Entropy classifier are decoded using dynamic programming. Nametag custom implementation of Viterbi algorithm has been optimized by removing impossible BILOU scheme transitions. BILOU scheme itself has been modified by reducing number of possible tags without losing any information.

1.11.9 Nametag features

Nametag uses standard set of classification features used for Named entity recognition. The features used in Nametag are following: form, lemma, part-of-speech tag and chunk (only in English) of current word and surrounding words (two preceding and two following words).

One big group of features are orthographic features such as capitalization, punctuation, lowercase and uppercase form of the word.

Nametag also uses simpler features including suffixes and prefixes of length 4 and regular expressions identifying possible year, date and time (in Czech).

The whole process of classifying and decoding runs twice (two stages) and the output from the first stage is used as one of input features for the second stage. The second run uses predictions for five words before and after the current word. It also uses predictions for the same words as the current word from previous 500 words.

Because Named entity recognition relies on external knowledge Nametag uses gazetteers which were collected from various sources and retrieved from Wikipedia [22]. Gazetteers from Wikipedia were created by processing the appropriate categories such as people, births and cities.

Nametag utilizes information from Brown clusters for both Czech and English language.

1.11.10 Gazetteers

Gazetteers are lists of words from the same category such as locations, organizations, etc. These lists of words are often used as additional data for various Natural language processing tasks. CoNLL-2003 shared task data includes gazetteers to improve Named entity recognition results and to research possibilities of using external data such as gazetteers for the task.

1.11.11 Brown clusters

Brown clustering algorithms group words into predefined number of clusters based on their context. Each cluster represents one ”class” of words that has similar contexts (eg. cities, countries, days of week, etc.). Brown clusters can be generated using unsupervised learning algorithms therefore any unannotated dataset of sentences can be used for training.

1.11.12 Word embedding

Word embedding algorithms map words or phrases to vector of real numbers. Vector distance of two given words represents similarity of their common context. Word embeddings are usually generated using neural networks. Vector representations in Word embeddings have better ability to model relationships between words compared to classes in Brown clusters.

1.11.13 Word2vec

Word2vec is the state of the art implementation of algorithm capable of generating word embeddings. It was developed by Thomas Mikolov et al. and released under Apache license [1]. Word2vec uses very simple model architectures compared to previous solutions [39]. Because of that it can be used on much larger datasets [2]. Nametag does not use word2vec but it is commonly used for natural language processing tasks. [47] [35]

[2]The training speed is in the order of billions of words [39].

Chapter 2

Analysis and design

2.1 Brief introduction

Phishing emails are usually targeted at a specific domain. Attackers try to impersonate a certain company. We believe that target of the phishing email is a valuable information that can be used for phishing email detection.

Named entity recognition recognizes entities in the text. Organization recognized in the email should give us enough information to determine the target of the phishing email. Named entity recognition has been already successfully used to detect phishing emails. [44]

2.1.1 Nametag

We have decided to use Nametag, the state of the art Czech Named entity recognition system implemented by Straková et al. [46] We suspect that we will need to train our own models for Nametag because Named entity recognition is a domain specific task. Nametag has models for English language that perform similarly to other state of the art Named entity recognition solutions. Therefore we will use Nametag for both Czech and English Named entity recognition.

2.1.2 Phishing target detection

We will develop phishing target detection solution based on Nametag. We plan to parse the email and send the parsed text to Nametag. Then we want to detect the phishing target based mainly on detected organizations.

2.1.3 Phishing email detection

Finally we will develop phishing email detection solution. We plan to use phishing target detection to generate features for the described phishing de-

tector. The rest of the features will be picked from the most successful features used in [48] and from the features used in [37].

We will use random forest classifier because it is a well performing simple method. If the performance of the random forest will not be sufficient we will use neural networks and compare those two methods.

We are going to use python [15] for development of our phishing detection solution and majority of related tools.

2.2 Nametag

In following sections we will describe how we plan to use Nametag, challenges we expect to face and how we want to solve those challenges.

2.3 Models for Nametag

The most important part of using Nametag are the models. Named entity recognition is domain specific task and we can not assume that original models are usable for phishing email detection.

We will evaluate the performance of original models available on Nametag website. [11] We suspect that the performance of the original models will be insufficient.

If so we will need to train our own models using custom training dataset. It is important to make sure that the models we train ourselves have similar performance to the original models. We will need to train the models using original datasets if we want to be able to properly test the custom trained models.

2.4 Parsing original dataset

In order to train Nametag models we will have to parse the original training data. We are worried that parsing will cause performance decrease because the CNEC dataset format is non-trivial.

There is a repository created by Straková at el. [19] which can be used to train Named entity recognition model. Using this repository may be beneficial because we will not have to parse the CNEC dataset and configure all the learning options and features for Named entity recognition. Therefore we believe models trained using this repository will have similar results compared to original models for Nametag.

On the other hand Nametag has other advantages. It is an optimized Named entity recognition solution that is polished and ready to use. [11] It has quite simple training format and feature selection. Also there is a brief documentation that describes running the recognizer, training new models and using Nametag as a REST server.

We will try to use the repository [19] before Nametag because it should be easier to use as is. After that we will decide which solution is better for our purpose.

2.5 Custom training data

Training data for Named entity recognition needs to be annotated. Data is usually annotated by human annotators. [4] This is a challenge we need to solve.

For testing purposes we will annotate some data by hand. But annotating huge amounts of data by hand is out of consideration and we will need to find other ways to create training data.

2.6 Creating training data

We have access to Email.cz data which includes thousands of legitimate emails grouped by domain. Emails are not annotated so we have to find a way to annotate this data.

2.6.1 Regular expressions

We could use regular expressions to match name of the organization in the email. We would need to create regular expression to match the organization name inside of emails with same domain. It would still require lot of human involvement because domain usually does not match the organization name and regular expressions might be hard to generate automatically.

2.6.2 Supervised learning

We might use a method based on supervised learning to annotate the data. This method takes advantage of the fact that Named entity recognition can recognize entities that are not present in the training data if the context of said entities is similar to data from the training dataset.

This method consists of iterations. In each iteration the emails grouped by domain are annotated using existing model. The most accurately tagged domains are determined. Emails belonging to the chosen domain are annotated using tagged data. Annotated data is added to the training dataset and model is retrained.

Some parts of the process would still require human action such as selected domains checking and generated training data checking. But that is expected when generating annotated data.

Essential part of this method is choosing the most accurately tagged domains. To achieve this we need to develop metrics that correlate with accuracy of the annotation. Metrics should be based on traditional evaluation metrics

such as precision and recall. Expected organization name for group of emails will be based on domain of said group.

2.6.3 Domain and organization name

Performance of both methods will be affected by the fact that organization name usually does not match the domain name. This is a serious problem that needs to be solved.

2.7 Matching organization and domain

In order to determine domain for each organization and vice versa we will use Firmy.cz. To get legal names for organizations we plan to use data from ARES. ARES is the information system which can be used to search economical subjects in Czech Republic. We will get data from Firmy.cz [6] for subjects obtained from ARES.

Firmy.cz is a catalogue website that gathers information about Czech organizations and offers interface to search in the gathered database. A website owned and maintained by Seznam.cz [18], leading technology company in Czech Republic.

Data available trough Firmy.cz contains domain of the subject if it has any. It also often contains some additional data such as common name for the subject.

2.8 Performance testing

We want test and evaluate the models we train in order to confirm that the performance of the models is sufficient for our purpose.

2.8.1 CoNLL2003 shared task

There is a evaluation script available for CoNLL2003 shared task. We plan to use it and evaluate the performance of the models we train.

2.8.2 Czech Named Entity Corpus

We did not find any evaluation script for CNEC dataset. We will try to find a way to evaluate our models using CNEC dataset as a reference.

2.9 Target detection

The goal of target detection is to determine the most likely target of the potential phishing email based on Named entity recognition output. We will

need to detect the most target in order to use Named entity recognition for phishing email detection.

We will find the organizations from the email that were recognized the most times. Recognized organizations will be used to determine the most likely target.

We are going to use data from Firmy.cz to map detected entities to domains. We will use domains as unique identifiers for targets of the phishing email.

2.10 Target detection performance testing

We are going to test the target detection performance of the trained models. We have found two resources we will use to evaluate the trained models. One of the resources is Phishtank and the second one are the emails grouped by domain from Email.cz data.

2.10.1 Phishtank

Phishtank has a database of phishing reports. These reports include target of the phishing attack. This can be used to create a dataset of phishing emails with targets. We will use this dataset to estimate the performance of our models.

2.10.2 Domain based testing

Emails grouped by domain can be also used to estimate the performance of the trained models. We are going to use domain name as a reference. We will use data from Firmy.cz to determine domains based on entities recognized by NER. We will use evaluate the performance of our trained models by comparing determined domains with reference.

2.11 Using NER for phishing detection

In this section we will briefly describe how we are going to use Named entity recognition for phishing email detection. First we plan to use NER to detect organizations present in the phishing email. As next step we want to determine most likely targets of the phishing email using detected organizations. The most likely targets will be used to generate features for the phishing detector itself.

We have a dataset of legitimate emails sent by many organizations gathered by Email.cz at our disposal. The legitimate emails from organizations and websites include many types of emails such as newsletters, special offers, emails with activation hypertext links, service updates, security notices and notifications. The emails are grouped by organization. We plan to train models on

these legitimate emails. We want to use the trained models to recognize any emails that look like they are sent by given organization but are not similar to legitimate emails regularly sent by given organization.

We plan to extract more data from the emails using Named entity recognition in order to improve phishing detector performance.

2.12 Phishing email detector

Before using Named entity recognition for phishing email detection we have to develop a baseline solution without features based on target detection. Once we develop a phishing email detection solution we will evaluate the performance of the detector without features generated by Named entity recognition. Then we are going to add the features we generated using Named entity recognition.

We will use random forest classifier implementation from scikit-learn [40] [17] python library for development of our phishing email detection solution.

We are going to use both phishing and non-phishing emails from Email.cz data 1.5.2 to evaluate the performance of our solution.

Features we are planning to use are described in following sections.

2.13 Common features

We have chosen a number of features for our baseline solution. We plan to use the most successful features according to [48] listed earlier 1.9 along with chosen features used in [37]. Features should detect some kind of suspicious characteristics that could mean that email is phishing.

2.13.1 Link-based features

Phishing emails almost always contain link leading to the pishing website. Therefore many features we are going to use represent some kind of information about links in the email.

Some features will be based on simple metrics such as max number of periods in the URL or total number of links in the email. Other features are going to be more sophisticated. One example of the later is a number of links that are not similar to majority of the links in the email.

2.13.2 Other suspicious characteristics

Links are not only thing that is characteristic for phishing emails. Anything that is common for phishing emails can be used as a feature. We plan to use features based on such suspicious characteristics. Some examples include the presence of HTML form tag or the presence of the HTML itself.

2.13.3 General features

Features that represent general characteristics of the email can provide a lot of information for phishing email detection. Because of that we will use features such as word count, body richness and subject richness.

2.14 Target detection features

We will generate features using the data from target detection. This section is describing basic types of features we plan to generate and use for the phishing email detection.

2.14.1 Simple target based features

Simple features that are easy to implement can still provide valuable information about the email. That is the reason why we are going to create a boolean feature that is true when any target was detected and false when there was no target detected.

2.14.2 Specific target features

Target detection gives us a lot of information. We want to use all the information obtained using target detection to detect phishing emails.

Therefore we will generate a features including information about specific target or targets that were detected from the email.

2.14.3 Suspicious links features

Links are important part of phishing emails. Because of that we are going to create a features that will detect suspicious links in the email. This feature will be based on the detected target and historical data about links used by the detected target.

Chapter 3

Realization

3.1 Named entity recognition

In following sections we are going to describe the process of using Named entity recognition for phishing target detection. We will focus on challenges and problems we had to solve in order to develop target detection that improves the performance of our baseline phishing email detection.

3.2 Nametag models

First challenge we faced was to obtain models that would be fit for our task. In this section we are going to describe how we determined the fitness of the existing models. We are also going to look at the process of training our own models and related challenges.

3.2.1 Original models

Nametag was released with models for Czech and English languages trained using CNEC and CoNLL2003 datasets respectively. We have suspected that it will be necessary to train custom models for Nametag because Named entity recognition is a domain-specific task and solutions trained on certain data won't perform adequately on data that is very different such as emails. [43]

We have used various emails from Email.cz as a testing data. While observing the output of the Nametag we have noticed that many organizations were not being recognized. After that we examined the training data and found out that majority of the organization we needed to detect was not present in the training data. We have confirmed that performance of the original models is insufficient for our purpose because of domain specificity.

We assumed that the biggest problem with original models was that many organizations were not present in the training data. Probably the second problem with the original data was that structure and language are a lot different

compared to typical email. The perfect solution for both problems would be creating a large custom dataset mainly using data extracted from emails. Problem with this approach is that creating the dataset is time consuming because the training data has to be annotated.

We have decided to use the original datasets as a core of our dataset and extend it by adding custom training data extracted from emails.

3.3 Training models using original data

We wanted to make sure that we are able to train models with similar performance as original models trained by Straková et al.

In order to compare the models properly we have used repository [19] created by Straková et al. because it has all the tools necessary for parsing the data and training Named entity recognition models. Which makes it easier to replicate the training process and match the results stated in [46].

First we downloaded all the necessary data such as CoNLL2003 dataset, unannotated data, brown clusters and gazetteers. Then we generated word2vec [23] word embeddings [3]. Finally we trained the Named entity recognition models [4]. Performance of the models was evaluated using tools available in the repository and the results were comparable to results from [46].

We have achieved to train models that match the performance claimed in the Nametag paper [46]. Table 3.1 shows results for . However we have faced several problems when using [19] repository. One of the problems was slow training speed. Next problem was the huge amount of RAM required for training. Another problem was complex training data format that would complicate adding custom data to the dataset. Also the repository is a quite complicated collection of scripts with a few hardcoded paths which makes it difficult to use.

Figure 3.1: Performance of our model trained using [19] repository [5]

	All entities (p/r/f)	Oneword (p/r/f)	Twoword (p/r/f)
Type:	74.71 / 80.95 / 77.71	79.36 / 82.68 / 80.99	79.63 / 81.13 / 80.37
Suptype:	78.37 / 84.89 / 81.50	83.97 / 87.45 / 85.68	81.35 / 82.88 / 82.11

[3] Generating word2vec took over 24 hours

[4] Training models took over 72 hours and required about 300GB of RAM.

[5] Table shows performance for one word, two word and all entities. "p/r/f" stands for precision/recall/f-measure. Type and Suptype stands for entity types and supertypes as described in section 1.10.2.

3.4 Training models for Nametag

Because of various problems we encountered while using [19] repository we have focused on Nametag. It can be used as an executable, a library or a REST server. The format of the training data is simple. And recognizer has two different output formats. [11]

The reason we used [19] repository before Nametag is that Nametag has a lot of configurable options for training models and we have not found parser to parse CNEC and CoNLL2003 training data format usable with Nametag. We originally wanted to reproduce the training process as closely as possible. We have suspected that using custom parser and custom configuration for training Nametag models would significantly reduce the performance of the trained models.

3.4.1 Parsing the training data

We have created a custom parser for CNEC plain text training data format. CNEC is available in plain text, XML, HTML and treex formats. We have chose the plain text format because we wanted to make it easier to add training data in the future. We also parsed the CoNLL2003 training dataset.

3.4.1.1 Training configuration

We used recommended training configuration [11] for Nametag training. We have trained Nametag models for Czech and English using parsed CNEC and CoNLL2003 training data respectively.

Figure 3.2: Performance of our model trained using Nametag [6]

	Precision	Recall	F-measure
All:	86.08 %	77.25 %	81.43 %
LOC:	91.05 %	74.20 %	81.76 %
MISC:	89.34 %	69.96 %	78.47 %
ORG:	75.40 %	77.48 %	76.43 %
PER:	89.09 %	83.77 %	86.35 %

3.4.2 Testing

We have used an official testing script [12] for CoNLL2003 shared task to evaluate the performance of the trained models. The results of model trained using CoNLL2003 dataset are in the 3.2. Models we trained ourselves achieved 82 F-measure on CoNLL2003 shared task. According to [46] original models

[6] First line shows total score of the model. Other lines show score for individual entity types.

achieved 89 F-measure. We consider performance of the models we trained suitable for our purpose.

3.4.3 Nametag training data format

Nametag training format is quite simple. It consists of two columns where the first column is the word and the second column is tag for the word with simplified BILOU scheme prefix 1.11.7. That is why we had no trouble adding more data to parsed CNEC and CoNLL2003 datasets. Adding custom data has proven to be extremely important as we found out later when performing tests.

3.5 Phishing target detection

We have found out that we are able to train models that achieve sufficient performance in general Named entity recognition task. Next task was to detect the most likely target from the emails using Named entity recognition. Input file is in EML format [16]. Desired output is the list of the most likely targets of the potential phishing email. We have developed a python script which is described in the next section.

3.5.1 Implementation

At first we parse the EML file using closed source szn-lib-mime library [7]. We use all HTML and plain text parts of the email. From headers we extract subject and the from field. We process the from field and use following information: email address, name, domain and local-part of the email address. All parsed text is passed to next part of the script.

We use guess_language python module [7] to detect the language of the email. Language detection can be overridden be passing language option to the script.

Based on guessed language or the language specified by user we choose the Nametag model that will be used for Named entity recognition. For English language we use model based on CoNLL2003 training data and for all other languages we use model based on CNEC training data. Our primary focus are emails in Czech language and Nametag was developed as a Czech Named entity recognition. That is why we decided to default to Czech model in cases when guess_language module fails to detect the language or when there is insufficient data for accurate language detection.

We use Nametag REST server to recognize the entities in the text. Originally we used Nametag executable instead of REST server. Before actual Named entity recognition Nametag loads the models into the memory which

[7] A library used to parse EML files in Email.cz

takes a long time compared to actual recognition [8]. Nametag REST server can stay running in the background and the models don't have to be loaded for every email which speeds up the recognition significantly.

After Nametag REST server returns recognized named entities output processing takes place. First we count the occurrences of each detected entity in the text. All words are lowercased before counting. After that we filter out entities that are not organizations. Then we get number of occurrences for every detected organization and sort all detected organization by the occurrence count. We have tested multiple different ways to process the Nametag output and this one gives the best results.

In order to simplify the future integration of the phishing target detection into the phishing detection solution we have created python module that can be easily used in any python script.

3.5.2 Testing using Phishtank

After developing phishing target detection solution we needed to estimate the accuracy of the prediction. We have decided to use data from Phishtank [14].

Phishing reports from Phishtank include malicious link used for phishing. Reports also usually include the phishing target of the attack.

We have used dataset of emails from Email.cz and picked the emails with existing Phishtank report. Then we have filtered out all the reports that did not have valid phishing target. Targets we removed included "Other", strings full of numbers and empty strings.

Because many reports were tagged by invalid phishing target the resulting dataset consisted of 58 emails with valid phishing targets.

We do not think that the results of the test should be taken very seriously because of the small amount of testing samples. But despite the small dataset we can still get general idea about performance and accuracy of the phishing target detection of our solution.

Next reason why we performed the tests was to compare see the difference between the models trained using original data and the models trained using custom and original datasets.

The models trained using original CoNLL2003 training data have predicted the phishing target correctly in 29 %of the cases.

We have taken some phishing emails from dataset available at Email.cz. The chosen emails have been annotated by hand and added to the training data. We have trained new models using CoNLL2003 dataset extended by new data from emails.

The new models trained using the extended dataset have predicted the phishing target correctly in the 58 % of the cases.

[8] Loading models takes a few seconds. The recognition itself takes under 0.01 seconds.

We have observed the testing data and the results of all performed tests. We have found out that before adding more more training data the main problem was that the Named entity recognition did not recognize the entities.

This changed after adding more training data to the dataset. Majority of the emails where target detection failed were either heavily obfuscated or the phishing target was not mentioned at all. This is something that concerns us because real world data can also have no target or can be obfuscated.

The performance is not ideal but it is sufficient for our usage. Because our solution will not be based entirely on the phishing target detection. We will use phishing target detection as one of the features for our phishing detection solution.

During the testing we have found out that Nametag is capable of recognizing entities that are not present in the training data.

3.5.3 Czech language

Unfortunately to our knowledge there is no database similar to Phistank for Czech language. Therefore we could not test Czech model using similar approach.

We wanted to find a way to test the target detection performance of the Czech models.

3.5.4 Email.cz data

We have access to the Email.cz dataset that includes legitimate emails sent by many organizations. These emails include many types of emails such as newsletters, special offers, emails with activation hypertext links, service updates, security notices and notifications. The emails are grouped by domain.

We have decided to use this data to test performance of the Czech models. In order to be able to use emails grouped by domains as reference we had to develop a solution that maps detected organizations to domains.

3.6 Organization to domain mapping

We gathered data about large number of organizations and obtained domains for those organizations. Then we have developed a solution that maps organization names to domains.

References to organizations in the emails are not usually done using their legal name. To solve this problems we used alternative names for the organizations. And we have developed a set of rules to generate mappings for organizations to match the organizations names effectively.

3.6.1 Data gathering

First we have downloaded a list of about 183 thousand organizations from [5]. We have extracted the legal names of the organizations. Organizations that were marked as defunct were deleted. Quotation marks were removed from organization names.

Next we have used Firmy.cz [6] data to get domains and common names for each organization. Sample of the data we have created is in the figure 3.3.

Figure 3.3: Sample of data from Firmy.cz[9]


```
{"title_use_alternative": false, "title_addition": ", s.r.o.",
"title": "ALWAID", "subject_id": 2166604,
"_ORGANIZATION": "ALWAID s.r.o.", "_HAS_URL": true,
"title_alternative": "", "_FOUND": true, "id": 647044,
"url_visible": "http://www.zachoval.cz"}

{"_ORGANIZATION": "ALZABRADLI s.r.o.", "_FOUND": false,
"_HAS_URL": false}

{"title_use_alternative": true, "title_addition": ", a.s.",
"title": "Alza.cz", "subject_id": 2121578,
"_ORGANIZATION": "Alza.cz a.s.", "_HAS_URL": true,
"title_alternative": "Alza.cz", "_FOUND": true,
"id": 12847874, "url_visible": "http://www.alza.cz"}

{"_ORGANIZATION": "Al-Zain Group s.r.o.", "_FOUND": false,
"_HAS_URL": false}
```


3.6.2 Implementation

We have developed a python script that uses this data 3.3 to map organizations to domains.

First the data about organizations is loaded. Organizations without specified domain are ignored. For each domain following process is used to create mappings.

All names and titles are extracted from the loaded JSON record. Extracted names and the domain are used to generate a set of possible alternative names using replacement and removal rules. Predefined groups of interchangeable strings, removable strings and removable characters are used to generate a large number of organization name alternatives. Names are added to the set and used as mappings for given domain.

[9]Data format is JSON record on separate line for each organization. Empty lines in this figure were added for readability.

Sample from the resulting set is in the figure 3.4. Some domains have multiple organizations associated with them which increases the size of the generated set.

3.6.3 English mappings

Mappings for 18 major international technology and financial organizations were added manually. The main focus of this thesis is Czech phishing email detection. Therefore this is sufficient number of english organizations.

Figure 3.4: Sample of domain mappings[10]

alpine pro , a.s.	alpine pro
alpine pro, a.s.	alpine pro morava s . r . o .
alpine pro stores,s.r.o.	alpine pro,a.s.
alpine pro stores,s.r.o	alpine pro stores, s.r.o
alpine pro stores , s.r.o.	alpine pro stores, s.r.o
alpinepro.cz	alpine pro stores, s r . o

3.7 Extending Nametag training data

It was necessary to extend the dataset because many organizations were not being recognized when using models trained with original data. We have already added some international organizations manually. But we could not do this for Czech organizations because of the huge amount of various organizations we want to detect.

3.7.1 Adding unstructured data

We have have added a few organizations to the dataset in the form of unstructured data without sentences. We have observed a very little performance decrease in general Named entity recognition task. However the performance of the recognition significantly improved for the entities we added to the dataset. We saw that we can add entities to the data without having to annotate sentences.

3.7.2 Large amounts of added data

Because we wanted to recognize a huge number of organizations we have added all 183 thousand entities from [5] to the dataset. Created training dataset was over 10 times the size of the original one. Performance of the recognition decreased drastically because of low quality of the added data.

[10]This is a selected sample from the set generated for ”alpinepro.cz” domain. Original size was 43 entries.

Many organization have long names that contain addresses, names or repetitive generic words. Such words, addresses and names were often misclassified as organizations during testing. Low quality of the data also caused a lot of errors in chunking which drastically affected overall performance of the system.

3.7.3 Choosing data for the dataset

We needed to minimize the errors caused by adding large amounts of low quality data. We have used a set of most frequent targets of the phishing attacks obtained from Email.cz. Names for the targeted domains were determined using data from Firmy.cz 3.3. Resulting 876 organization names were added to the dataset. Amount of recognized entities was significantly increased for the targets that were added to the dataset. We have noticed a small amount of chunking errors. Misclassifications caused by added data were almost nonexistent.

3.7.4 English dataset

Unfortunately data from Firmy.cz does not include English entities and their domains. Because of that we could not have used the same approach. We have manually annotated and added a few dozens of hand picked emails to the English dataset.

3.8 Testing domain detection

We have extended original Nametag datasets and trained new models. We have also developed solution for mapping organizations to domains.

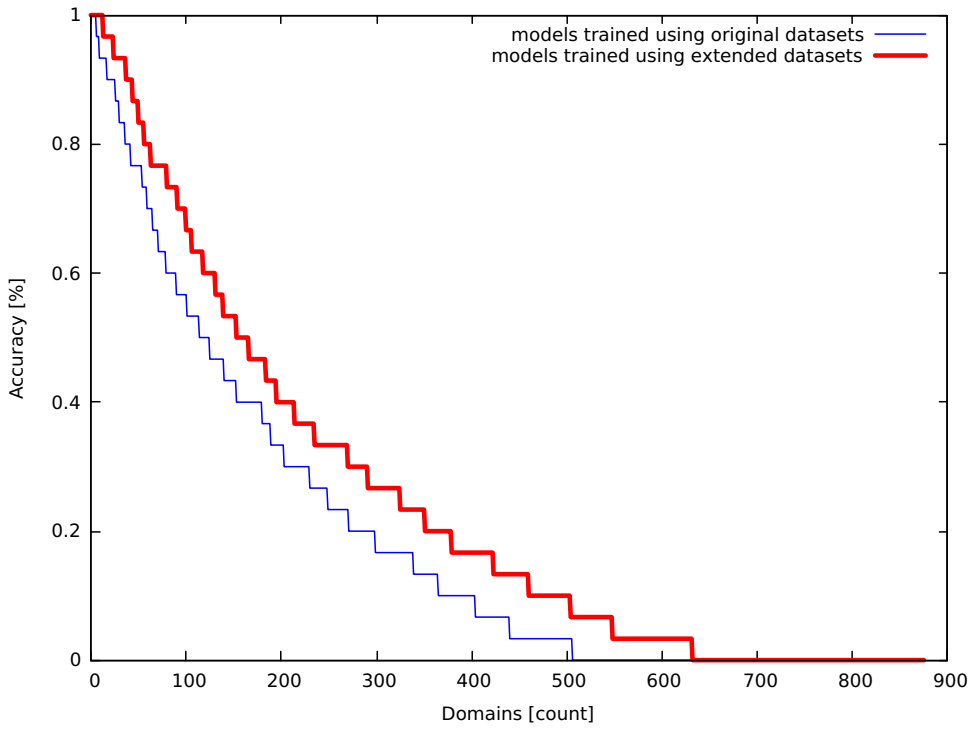
Next thing we did was testing and evaluation of the performance of our target domain detection solution. In the following sections we will describe the process and the results of the testing.

The data used for testing comes from Email.cz dataset. The emails we used were grouped by domain. We have taken 30 emails for each of 876 domains and used our domain detection solution. Detected domains were compared with reference domains. We have calculated the detection accuracy for each domain.

We have evaluated the performance of the models we trained. Performance of the models trained using extended dataset was compared to the performance of the model trained only with original datasets.

The graph 3.5 shows the accuracy of the detection for each of used domains. The area under the graph represents total accuracy of the domain detection solution. Domains on the left were detected correctly 100 % or the times. Domains on the right were not detected at all.

Figure 3.5: Target detection accuracy[11]



3.8.1 Results

Total accuracy of the original model is 18 %. Total accuracy of the extended model is 24 %. We can see that the extended model caused a significant performance increase in domain detection. However 24% accuracy cannot be generally considered a good result.

In order to make conclusions about performance of our domain detection solution we have to look at the distribution of the domains within the graph. This is important because we need to detect the domains that are common targets of the phishing attacks. Also it is not a problem if we cannot detect domains that are never attacked.

Figure 3.6 shows us a three samples from the distribution of the domains. We can see that sample of the most detected domains contains a lot of financial institutions. In contrast the sample of domains that were not detected at all contains mostly unknown domains or domains that are unlikely targets of the phishing attacks.

We examined the emails from the dataset and we have found out that some emails do not contain any mention of the targeted organization. This naturally lowers the performance achievable in this test.

[11] Graph shows accuracy of detection for each domain.

Another thing that makes the domain detection hard is the number of domains we are trying to detect.

Because of observation we made we consider our domain target solution sufficient for our purposes. We have proved that it is possible to improve domain detection performance by training better named entity recognized models.

Figure 3.6: Target detection samples[12]

80 % and more	around 55 %	0 %
airbank.cz	praha9.cz	alza.cz
erasvet.cz	totalbrokers.cz	arcelormittal.com
itesco.cz	berpujcku.cz	babynabytek.cz
realmoney.cz	ceskapojistovna.cz	boschrexroth.cz
aaaauto.cz	cez.cz	canariatravel.cz
ebay.com	dracik.cz	c-budejovice.cz
ifortuna.cz	eufrat.cz	cement.cz
zasilkovna.cz	homecredit.cz	ceskyserver.cz
tchibo.cz	paypal.com	coleman.cz
victoriatip.cz	pns.cz	e-rezervace.cz
gigaserver.cz	jtbank.cz	fler.cz
royalvision.cz	sberbankcz.cz	mendelu.cz
slavia-pojistovna.cz	agrofert.cz	mora.cz
equabank.cz	hyundai.cz	o2.cz
vodafone.cz	lr-czech.com	pokrok.cz
bohemiacargo.cz	okgroup.cz	prvni-lekarna.cz
armexenergy.cz	uvt.cz	sam73.cz
europ-assistance.cz	ascari.cz	sipkova.cz
kbps.cz	axa.cz	spcr.cz
reprofit.cz	mall.cz	vzp.cz

3.9 Phishing email detection

We have successfully developed a targeted domain detection solution. Next thing we had to do was developing a phishing detection solution that would allow us to evaluate the performance of the features based on the detected target.

[12]We have randomly selected the same amount of domains for each column. Then we have picked the most interesting domains from each selected sample.

3.9.1 Baseline solution

We have implemented a baseline phishing email detection solution using python [15]. We have used random forest classifier implementation from the scikit-learn library. [17] The classifier was configured to use 10 decision trees in the random forest.

For the baseline solution we chose 26 features. Features we used include all features listed in the features section 1.9. Some of the used features were taken from [37].

3.9.2 Testing baseline solution

We have split the Email.cz dataset and used 20 % of the data for testing. Accuracy achieved by the baseline solution using all available features was 97.6.

We have performed more tests using only one feature at the time. The three features with highest accuracy were number of phishing keywords, maximum number of periods in the URL and body richness.

Number of phishing keywords achieved 82.2 accuracy. Phishing keywords are predefined words that are characteristic for phishing emails.

Maximum number of periods in URL achieved 81.2 accuracy. It is interesting that such a simple feature can be so helpful for phishing email detection.

Body richness achieved 80.5 accuracy. This feature is defined in state of the art of this thesis 1.9.1 section.

3.10 Final solution

Final solution was created by incorporating target based features into the baseline solution. Target detection solution outputs an array of detected domains for each email. Each domain in the array has an assigned probability that represents the likelihood of the domain being a target of the email.

Probability is determined by counting all entities in the email that can be mapped to the domain. Then the counts are normalized for all domains returned in the array so that the sum of all probabilities equals to 100 %.

3.10.1 Target based features

Output from target detection was used to generate following features.

Domain detected feature is a boolean feature that is true if any domain was detected.

Top 1 detected domain feature which includes two values. First value is an id of the most probable detected domain. The second value is percentage that the domain is the target.

Top 5 detected domains feature that is much like the top 1 feature. Except it includes percentages and id's for top 5 detected domains.

Last feature we have generated was boolean feature that detects mismatched links. Link is considered mismatched if it leads to domains that are not usually present in the emails sent by this specific target. Commonly linked domains are determined using historically collected data by Email.cz. We have implemented aggressive and benevolent versions for this feature. Both of these versions achieved similar performance and we will not differentiate between them from now on.

3.10.2 Testing final solution

We have used the same testing data for both final solution and baseline solution testing. Accuracy achieved by the final solution using all available features was 98. Which is a minor improvement over baseline solution with 97.7 accuracy.

We have performed more tests using only one feature at the time. All three features with the highest accuracy were based on target detection.

The three top features were: boolean domain detected feature, percentage for top 1 detected domain and link mismatch feature.

All three features achieved similar accuracy of 86.9. Compared to 82.2 accuracy for best feature from baseline solution.

3.10.3 Results

Testing showed us that target based features can improve the performance of the phishing emails detection. We have found out that target based features provide a significant amount of information about an email. Target based features outperform the commonly used features for phishing email detection.

3.10.4 Possible improvements

This thesis proves that Named entity recognition can be used for Czech phishing email detection. However there is a lot of ways to improve the current solution.

More data for Named entity recognition may be created. New better Nametag models could be trained using created data. This should improve to domain specific performance of the model.

Mapping is currently done using keyword and dictionary based approach. This is not ideal because it depends on predefined data. More methods for creating mappings should be researched.

Bigger phishing email dataset should be gathered to reduce the bias of the dataset. Observations made in this thesis should be confirmed by testing using live traffic.

Conclusion

We have researched the state of the art phishing email detection methods and the common methods of machine learning performance evaluation. We have researched the state of the art Czech Named entity recognition system and the possibilities of using it to detect phishing attacks.

We have analyzed Nametag, the state of the art Czech Named entity recognition system. We managed to train models that match the performance of the original models [46]. Models for Nametag were trained using original datasets. The performance of the models was insufficient for our purpose because emails differ from structure of the original training data. Domain specific performance have been improved by adding custom data to the original datasets.

We have developed a solution that can predict the targeted organization of the phishing attack based on the Nametag output. We have used data from Phishtank [14] to test the accuracy of our solution. Models used for testing were trained using both original dataset and improved extended dataset.

We have downloaded the list of the Czech subjects from ARES [5]. Firmy.cz [6], the catalogue of Czech organizations, was used to get matching domains for downloaded subjects. We have developed a solution that maps recognized entities to domains using data from ARES and Firmy.cz. Mappings for several important international banking and technology companies were added manually. Czech dataset was extended using generated mappings and list of most attacked domains from Email.cz. We have used legitimate emails from Email.cz dataset grouped by domain to evaluate performance of the solution. Results of the testing showed the performance increase achieved by using the extended models.

We have developed a baseline phishing email detector using random forest classifier. Features used for baseline solution were based on research of both English and Czech phishing detection solutions. The performance of the baseline solution was measured.

Phishing target detection solution and domain mappings were used to generate features for phishing email detection. Generated features were added

to the baseline solution and the performance was evaluated. Target based features caused a minor improvement in the total accuracy of the phishing detection solution. Accuracy achieved by the final solution with target based features was 98 compared to 97.7 accuracy achieved by baseline solution.

We have performed tests using only one feature at the time to discover the most significant features for phishing email detection. The target based features achieved the best accuracy out of all used features. The top three most significant features were all based on target detection. These features achieve 86.9 accuracy when used as one feature at the time. For comparison the best feature from the baseline solution scored 82.2 accuracy.

These results suggest that target based features are powerful features for phishing email detection. Target based features improve the overall performance of the phishing email detection solution. We are convinced that target based features are effective way to detect phishing emails.

More fine-tuning of the solution is a subject for future work. Further improvements may be achieved by improving various parts of the process such as Named entity recognition, target detection and domain mapping. Phishing email detection solution itself could be improved by adding more target-based features or trying different machine learning methods. Observations made in this thesis should be confirmed by testing using bigger dataset or live traffic.

Bibliography

[1] *Apache License, version 2.0.* Accessed: 15th May 2017.
URL http://www.apache.org/licenses/LICENSE-2.0

[2] *Conference on Computational Natural Language Learning.* Accessed: 15th May 2017.
URL http://www.cnts.ua.ac.be/conll2003/

[3] *CoNLL-2003 shared task.* Accessed: 15th May 2017.
URL http://www.cnts.ua.ac.be/conll2003/ner/

[4] *Czech Named Entity Corpus.* Accessed: 15th May 2017.
URL http://ufal.mff.cuni.cz/cnec

[5] *Data about subjects from ARES system - opendata.cz website.* Accessed: 15th May 2017.
URL https://linked.opendata.cz/cs_CZ/dataset/mfcr-ares-or/resource/1f7bb861-51c1-458e-9da6-830787f67de3

[6] *Firmy.cz - Catalogue of Czech organizations.* Accessed: 15th May 2017.
URL https://www.firmy.cz/

[7] *guess-language 0.2 package page on Python package index.* Accessed: 15th May 2017.
URL https://pypi.python.org/pypi/guess-language

[8] *HTML Forms - w3schools.* Accessed: 15th May 2017.
URL https://www.w3schools.com/html/html_forms.asp

[9] *HTML Forms - w3schools.* Accessed: 15th May 2017.
URL http://www-nlpir.nist.gov/related_projects/muc/proceedings/ne_task.html

[10] *J. Nazario. Phishing corpus.* Accessed: 1st February 2015.
URL http://monkey.org/~jose/wiki/doku.php?id=phishingcorpus.

[11] *NameTag website*. Accessed: 15th May 2017.
URL http://ufal.mff.cuni.cz/nametag

[12] *Official CoNLL2003 shared task evaluation script*. Accessed: 15th May 2017.
URL http://www.cnts.ua.ac.be/conll2002/ner/bin/conlleval.txt

[13] *Phishing activity trends report - APWG*. Accessed: 15th May 2017.
URL http://docs.apwg.org/reports/apwg_trends_report_q4_2016.pdf

[14] *Phishtank website*. Accessed: 15th May 2017.
URL https://www.phishtank.com/

[15] *Python - programming language*. Accessed: 15th May 2017.
URL https://www.python.org/

[16] *RFC 5322 - Internet Message Format*. Accessed: 15th May 2017.
URL https://tools.ietf.org/html/rfc5322

[17] *scikit-learn - Machine Learning in Python*. Accessed: 15th May 2017.
URL http://scikit-learn.org/stable/

[18] *Seznam.cz - Web portal and search engine in Czech Republic*. Accessed: 15th May 2017.
URL https://www.seznam.cz/

[19] *Software and data accompanying paper Neural Networks for Featureless Named Entity Recognition in Czech*. Accessed: 15th May 2017.
URL https://github.com/strakova/ner_tsd2016

[20] *Spam Assassin project*. Accessed: 15th May 2017.
URL http://spamassassin.apache.org/

[21] *Spam Assassin project phishing corpus*. Accessed: 15th May 2017.
URL http://spamassassin.apache.org/old/publiccorpus/

[22] *Wikipedia - The Free Encyclopedia*. Accessed: 15th May 2017.
URL https://www.wikipedia.org/

[23] *word2vec - Tool for computing continuous distributed representations of words*. Accessed: 15th May 2017.
URL https://code.google.com/archive/p/word2vec/

[24] Abu-Nimeh, Saeed, Nappa, Dario, Wang, Xinlei, and Nair, Suku. *A comparison of machine learning techniques for phishing detection*. In Proceedings of the anti-phishing working groups 2nd annual eCrime researchers summit, pp. 60–69. ACM, 2007.

[25] Almomani, Ammar, Gupta, BB, Wan, Tat-chee, Altaher, Altyeb, and Manickam, Selvakumar. *Phishing dynamic evolving neural fuzzy framework for online detection zero-day phishing email.* arXiv preprint arXiv:1302.0629, 2013.

[26] Basnet, Ram, Mukkamala, Srinivas, and Sung, Andrew H. *Detection of phishing attacks: A machine learning approach.* In Soft Computing Applications in Industry, pp. 373–383. Springer, 2008.

[27] Bergholz, Andre, Chang, Jeong Ho, Paass, Gerhard, Reichartz, Frank, and Strobel, Siehyun. *Improved Phishing Detection using Model-Based Features.* In CEAS. 2008.

[28] Chandrasekaran, Madhusudhanan, Narayanan, Krishnan, and Upadhyaya, Shambhu. *Phishing email detection based on structural properties.* In NYS Cyber Security Conference, pp. 1–7. 2006.

[29] Falk, Courtney. *CERIAS Tech Report 2016-3 Knowledge Modeling of Phishing Emails.* 2016.

[30] Fette, Ian, Sadeh, Norman, and Tomasic, Anthony. *Learning to detect phishing emails.* In Proceedings of the 16th international conference on World Wide Web, pp. 649–656. ACM, 2007.

[31] Florian, Radu, Ittycheriah, Abe, Jing, Hongyan, and Zhang, Tong. *Named Entity Recognition through Classifier Combination.* In Walter Daelemans and Miles Osborne, editors, Proceedings of CoNLL-2003, pp. 168–171. Edmonton, Canada, 2003.

[32] Forney, G David. *The viterbi algorithm.* Proceedings of the IEEE, 61(3):268–278, 1973.

[33] Friedberg, Richard M. *A learning machine: Part I.* IBM Journal of Research and Development, 2(1):2–13, 1958.

[34] Konkol, Michal and Konopík, Miloslav. *Maximum entropy named entity recognition for Czech language.* In International Conference on Text, Speech and Dialogue, pp. 203–210. Springer, 2011.

[35] Lample, Guillaume, Ballesteros, Miguel, Subramanian, Sandeep, Kawakami, Kazuya, and Dyer, Chris. *Neural architectures for named entity recognition.* arXiv preprint arXiv:1603.01360, 2016.

[36] Lee, Jin-Lee, Kim, Dong-Hyun, and Chang-Hoon, Lee. *Heuristic-based Approach for Phishing Site Detection Using URL Features.* 2015.

[37] Listík, Vít. *Detekce podvodných emailú v češtině.* 2015.

[38] Ma, Liping, Ofoghi, Bahadorrezda, Watters, Paul, and Brown, Simon. *Detecting phishing emails using hybrid features.* In Ubiquitous, Autonomic and Trusted Computing, 2009. UIC-ATC'09. Symposia and Workshops on, pp. 493–497. IEEE, 2009.

[39] Mikolov, Tomas, Chen, Kai, Corrado, Greg, and Dean, Jeffrey. *Efficient estimation of word representations in vector space.* arXiv preprint arXiv:1301.3781, 2013.

[40] Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., and Duchesnay, E. *Scikit-learn: Machine Learning in Python.* Journal of Machine Learning Research, 12:2825–2830, 2011.

[41] Ramanathan, Venkatesh and Wechsler, Harry. *Phishing detection and impersonated entity discovery using Conditional Random Field and Latent Dirichlet Allocation.* computers & security, 34:123–139, 2013.

[42] Ratinov, L. and Roth, D. *Design challenges and misconceptions in named entity recognition.* CoNLL 2009: Proceedings of the Thirteenth Conference on Computational Natural Language Learning, pp. 147–155, 2009.

[43] Ritter, Alan, Clark, Sam, Etzioni, Oren, et al. *Named entity recognition in tweets: an experimental study.* In Proceedings of the Conference on Empirical Methods in Natural Language Processing, pp. 1524–1534. Association for Computational Linguistics, 2011.

[44] Shyni, C Emilin, Sarju, S, and Swamynathan, S. *A Multi-Classifier Based Prediction Model for Phishing Emails Detection Using Topic Modelling, Named Entity Recognition and Image Processing.* Circuits and Systems, 7(09):2507, 2016.

[45] Straková, Jana. *Neural Network Based Named Entity Recognition.* 2016.

[46] Straková, Jana, Straka, Milan, and Hajič, Jan. *A new state-of-the-art czech named entity recognizer.* In International Conference on Text, Speech and Dialogue, pp. 68–75. Springer, 2013.

[47] Straková, Jana, Straka, Milan, and Hajič, Jan. *Neural Networks for Featureless Named Entity Recognition in Czech.* In International Conference on Text, Speech, and Dialogue, pp. 173–181. Springer, 2016.

[48] Tоolan, Fergus and Carthy, Joe. *Feature selection for spam and phishing detection.* In eCrime Researchers Summit (eCrime), 2010, pp. 1–12. IEEE, 2010.

[49] Wittel, Gregory L and Wu, Shyhtsun Felix. *On Attacking Statistical Spam Filters.* In CEAS. 2004.

Appendix A

Acronyms

API Application Programming Interface

ARES Administrativní registr ekonomických subjektů (Administrational registry of economical subjects)

CNEC Czech Named Entity Corpus

CoNLL Conference on Computational Natural Language Learning

EML Elecronic Mail

JSON JavaScript Object Notation

MEMM Maximum Entropy Markov Models

ML Machine Learning

MUC Message Understanding Conference

NER Named Entity Recognition

NLP Natural Language Processing

HMM Hidden Markov Models

RAM Random Access Memory

REST Representational State Transfer

RFC Request for comments

URL Uniform Resource Locator

Appendix B

Contents of enclosed SD card

- `data` ..... the directory of data
  - `feature_data` ..... the directory of extracted features for training
  - `organizations_data` ..... the directory of organizations data
- `Makefile` ..... Makefile for running selected parts of the implementation
- `readme.md` ..... the file with SD card contents description
- `results` ..... the directory with results of the training
- `src` ..... the directory of source codes
  - `code` ..... implementation sources
    - `addTargetFeatures.py` .... script for adding target based features
    - `train.py` ..... script for training the phishing email detector
    - `more` ..... more implementation sources
  - `thesis` ..... the directory of LaTeX source codes of the thesis
- `text` ..... the thesis text directory
  - `thesis.pdf` ..... the thesis text in PDF format