Probabilistic password recognition

Tiko Huizinga
thuizinga@os3.nl
Supervisor: Zeno Geradts

February 10, 2019

Abstract

In this paper we create a method to classify strings as either a password or a "normal word". We compare two methods of classification. The first method is a naive probabilistic approach which we designed and implemented ourselves. The second method is using a Support Vector Machine (SVM). An SVM is a machine learning classification tool which can also be used for text classification. The classification happens based on string characteristics like the length and the number of special characters in the string. We use two sets of training data to train our models, a password list and a Wikipedia dump as "normal word" list. We evaluate both methods using the precision, recall and F1 score. Our results show that the naive probabilistic approach achieves better results than the SVM.

1 Introduction

Password authentication is still a very popular way of authenticating users. When a law-enforcement agency seizes the hard drive of a suspect, they have a small window of time to gather evidence from it to extend pre-trial detention. Because of the large amount of data, automated tools can be useful to scan a drive for interesting files. Files containing passwords are especially interesting because they can provide access to extra data. The Netherlands Forensic Institute (NFI) created a search engine called Hansken which uses machine learning to help forensic investigators find evidence and other interesting data[7][3]. Hansken does not yet have the functionality to search for passwords. This research focuses on classifying strings as either a normal word or a password. There has been a lot of research on the strength of passwords [1][2][6] but little to no research has been done on the probability that a string could be a password.

In theory, every string could be a password. Because of this, there is no way to know for sure whether a string is a password or not. This research aims to create a probabilistic classifier and evaluate its performance using common metrics.

2 Research question

The main question for this research is:

How can software be used to classify whether a string is a password or a "normal" word?

These sub-questions will help with answering the main question:

1. What characteristics differentiate a password from 'regular' text?
2. How can these characteristics be used to come to an algorithm that classifies a string as either a password or a "normal" word?

3 Related work

Dell' Amico et al. [2] did an empirical analysis on password strength. Their main research question was "Given a number of guesses, what is the probability that a state-of-the-art attacker will be able to break the password". They found that in the beginning of a dictionary attack, passwords were guessed more frequently than later when the attack continues with less common passwords. This means that a major part of users use a relatively easy to guess password and as the password strength increases, the chance of guessing the password decreases rapidly.

Bonneau et al. did a thorough statistical analysis on 70 million passwords [1]. The main focus of their research was on the probability for an attacker to guess a users password in N attempts. One of their findings is that the password distributions vary little under different populations of users (gender, age, language preference). This means that an attacker performing an optimal attack with knowledge of the most used passwords, on average, needs around the same amount of tries to guess a password for each population.

Melicher et al. use artificial neural networks to model passwords' resistance to guessing attacks [6]. This means given a password, the neural network assesses the strength of the password, based on a list of known passwords.

Veras et al. use natural language processing (NLP) to classify and generalize semantic categories from passwords [8]. For the semantic classification, three classes for substrings of passwords are used: verb, noun or gap-segment. They also use a WordNet based classification to classify substrings of passwords to different categories like car, dog or love. Where their research uses classification of substrings within passwords, our research classifies strings to be either a password or not.

4 Scope

The goal of this research is to help forensic investigators with quickly finding passwords saved in plain text on a suspects hard drive. The idea is that all

the words in these text files go through the classifier to determine if it is a password. This classification is not aimed to guarantee whether a string is a password or not because theoretically, any string could be a password. It aims to give investigators of the hard drive an indication of where to look.

In this context, we define a word in a text file as a string separated by a starting and ending space or newline character. This means that we will not be able to classify passwords with a space character in them.

The classification will happen based on two training sets: a password set and a "normal word" set. The results of this research are dependent on the content of these sets. Using a different training set will lead to different results.

5 Characteristics, data sets and statistics

This section first explains which characteristics of strings to train our model with. We then choose what data sets we will use to calculate these characteristics of for both the password and word list. We support this choice with the statistics of these characteristics for both the password and word list.

5.1 Characteristics

In related work, there have been many ways to analyze the structure of passwords. One method we see in numerous studies is the use of basic string characteristics like *password length, number of special characters, number of digits, number of capital letters* and *number of lowercase characters* [2][1][5]. Other characteristics in related work are semantics and word categories [8] and language [2][4].

For our research, we will use the basic characteristics for password classification.

- Length
- # Special characters
- # Digits
- # Uppercase characters
- # Lowercase characters

We choose for these characteristics because we have seen them the most in related work and because this research serves as an introduction to password classification. Using more and other types of characteristics for this type of password classification can be useful future work.

Listing 1: Function to create word characteristics

```
def createWordCharacteristics(word):
        length = special = digits = capital = small = 0
        for c in word:
                length += 1
                ascii = ord(c)
                if ascii < 48:
                        special += 1
                elif ascii < 58:
                        digits += 1
                elif ascii < 64:
                        special += 1
                elif ascii < 91:
                        capital += 1
                elif ascii < 97:
                        special += 1
                elif ascii < 123:
                        small += 1
                else:
                        special += 1
        return length, special, digits, capital, small
```

5.2 Training data

To train our classifiers what a password and what a normal word is, we need training data. The training sets we choose influence the model and the result. A classifier performs better when the characteristics of the password list and the word list differ more. It is very important to have a training set with a very close relation to the real world application the classifier is used in. The scope of this research defines that the classifier is used on words in text files on hard drives of suspects.

5.2.1 Passwords

There are different options to obtain a password list. We need to consider a few aspects while choosing between these options. The first aspect is the ethical considerations. Users creating a password do not expect anyone to ever read or know their password. Passwords are per definition very private because they are supposed to be secret. The second aspect is how close the password list relates to the real world application of the classifier. The last aspect is the size of the data set. When a data set is too small, the resulting classifier can not train enough and will perform worse. Keeping these aspects in mind, we consider the following choices for obtaining the password set: a user survey, a common password list and a password breach compilation list.

User survey: Doing a user survey asking users to enter a password to use for this research is not ideal. Ethically speaking, there are no problems when the users are informed what the survey is used for. The problem lies in the fact that in the four weeks of this research, we will not get enough results to optimally train the classifiers. It will also be hard to determine if the characteristics of the resulting list are similar enough to the real world application because users might change their behavior when filling in a survey asking for a password compared to entering it on a real website.

Common password list: A common password list does not contain any Personally Identifyable Information (PII) per definition because all passwords in there are used by many users. The problem with a common password list, is that it loses a lot of information compared to a full password breach list. Not every users password is in a common password list or shares characteristics with those passwords and such more complex and unique passwords will then be harder to classify.

Password breach compilation: A full password breach compilation would result in a data set that relates very close to the real world. It would also contain more than enough data to train a realistic model. A disadvantage of such a list is that it contains PII and contains passwords users could still be using.

For this research we choose to use a password breach compilation so we can optimally train our models. The password breach we use is published on GitHub[1]. We will not use the list for anything else than for the purposes of this research. We believe that because this list is publicly available, the advantages of using this list outweigh the relatively small ethical disadvantage.

This password list contains 1.1 billion passwords from various sources. Unfortunately we can not verify the sources and thus the authenticity of this list. For this research we will assume that this list contains valid passwords. If it turns out this is not a real password breach compilation, the resulting classifiers can be easily retrained with another password list.

5.3 Words

We want to create a word list that represents the reality. The reality is a computer with text files. To represent the content of these text files, we will use a partial English Wikipedia dump [2]. Before we can use this data, we first need to parse it to remove the xml and Wikitext syntax. When this is done, we are left with a text file with just the plain text from the articles and the URL's from the references. We assume that this represents text files on computers. The total size of the resulting file is 53 million words.

5.4 Statistics

Using the characteristics we can create statistics on the password and word list. These statistics are used to train the naive probabilistic classifier. For each

[1] https://gist.github.com/scottlinux/9a3b11257ac575e4f71de811322ce6b3
[2] https://dumps.wikimedia.org/enwiki/20190120/

characteristic for both data sets a table is crated with the absolute and relative occurrence of that characteristic. An example of a part of such a table can be seen in Table 1. The values in this table are rounded for readability purposes.

Word length	Count (10^6)	Percentage (%)
1	1.6	3
2	8.0	15
3	9.3	17
4	6.4	12
5	5.9	11
6	5.0	9
7	4.9	9
8	4.1	8

Table 1: Table with length of words in Wikipedia list

5.4.1 Length

In the Wikipedia word list at the top of Figure 1, we see a clear peak in words with three characters. The form of the chart is a right skewed normal distribution. The chart is cut off at length larger than 20 characters.

The length of passwords in the breach compilation list is plotted at the bottom of Figure 1. This is a very clear multimodal normal distribution because of two peaks at both 8 and 10 characters. This could have been caused by different requirements for password in the breach compilation. One website could have required an 8 character password and another website 10 characters. Further, we see a smaller peak 15 character passwords. This could indicate another website requiring 15 character passwords.

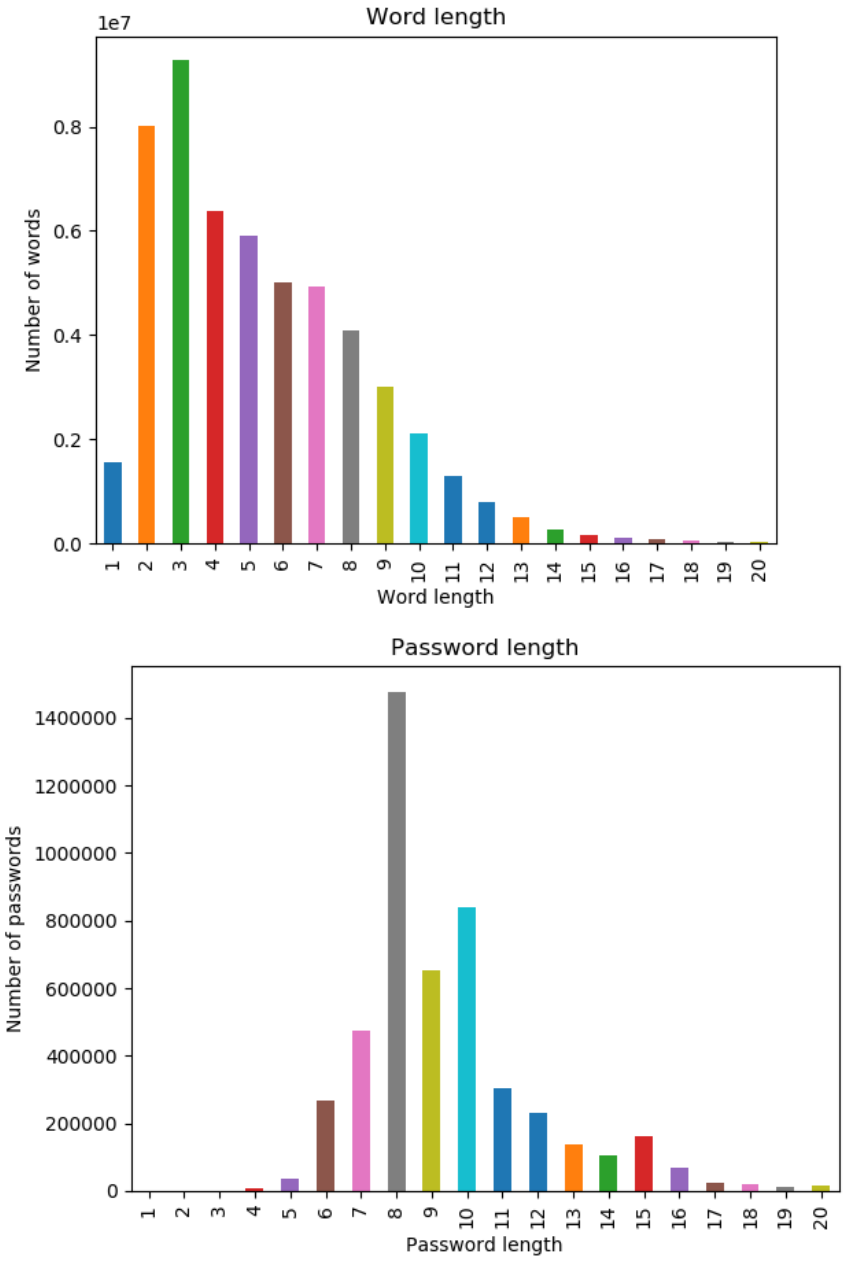

Figure 1: The length of words and passwords

5.4.2 Number of digits

In the Wikipedia word list in Figure 2, most words do not contain digits. There is a small peak at words containing 4 digits. These are probably year numbers.

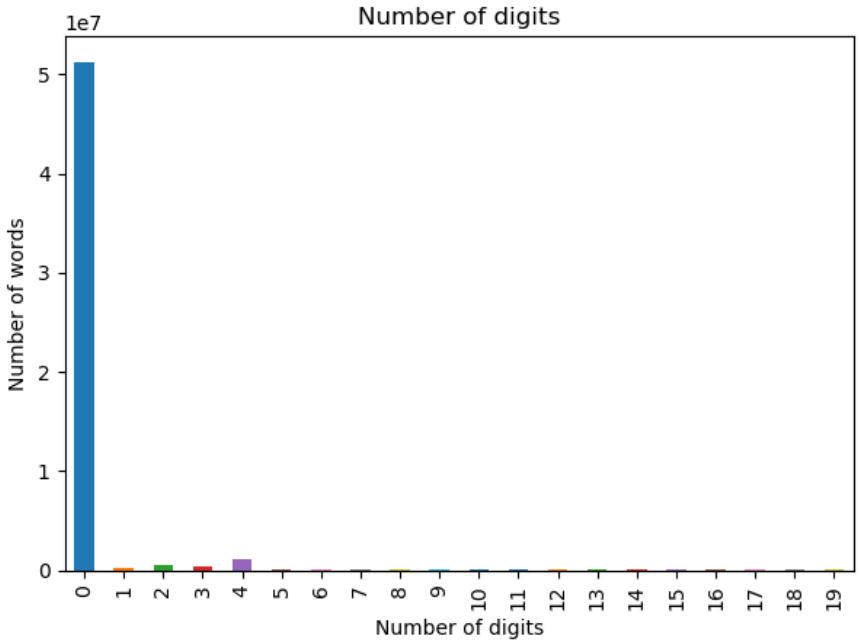

Figure 2: Number of digits in words in the Wikipedia list

In the password list, the largest peak is at passwords without any digits but we do see that digits in passwords are way more common than in the Wikipedia list. We can clearly see peaks at the even numbers, a possible explanation for this is that dates and times are often written in an even amount of digits. We also see a clear peak at passwords with 15 digits. This increase is about the same as the increase we saw for passwords with length 15 so that means that a subset of the breach compilation contains 15 digit long passwords.

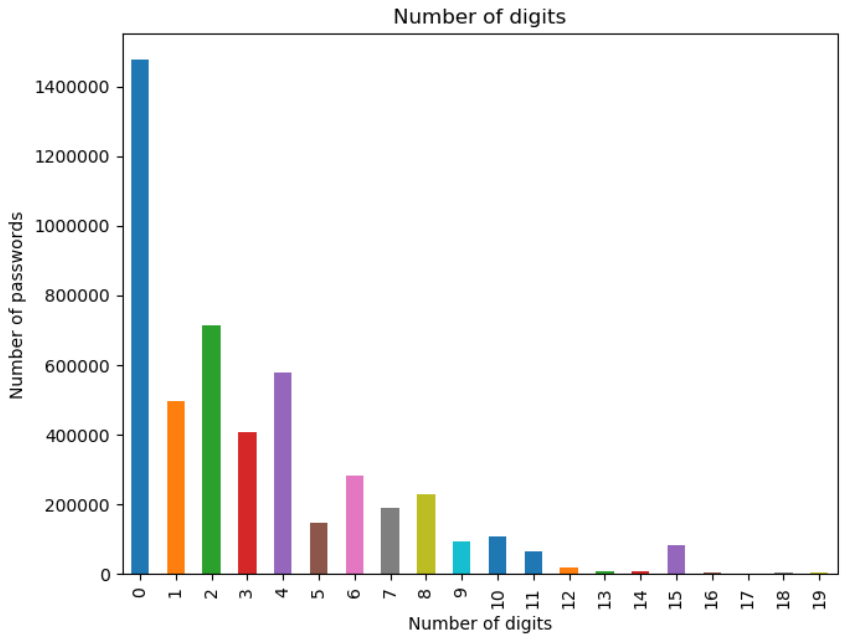

Figure 3: The number of digits of passwords in the breach compilation list

6 Classifiers

6.1 Naive probabilistic classification

The first classifier we create is a naive probabilistic classifer. This classifier is based on the statistics over the characteristics defined in Section 5.1. We define the following sets class C and characteristics X.

$C = \{Password, Word\}$

$X =$ {Length, #Special characters, #Digits, #Capital characters, #Lower case characters}

We then define the following functions to retrieve the statistical values from the earlier created statistics for both the word and password list.

$w(x, y) =$ Percentage of words with value y for characteristic x

$pw(x, y) =$ Percentage of passwords with value y for characteristic x

Probability of the class being Password, given a single characteristic is the percentage of passwords with that characteristic divided by the percentage of passwords with that characteristic plus the percentage of words with that characteristic.

$$P(C = Password|x\epsilon X, y\epsilon\mathbb{N}) = \frac{pw(x,y)}{pw(x,y) + w(x,y)}$$

The probability of the class being Password, using multiple characteristics, is the average of the probability for all the separate characteristic probabilities.

$$P(C = Password|x_1, \ldots, x_n\epsilon X, y_1, \ldots, y_n\epsilon\mathbb{N}) = \frac{P(C=password|x_1,y_1)+\ldots+P(C=password|x_n,y_n)}{n}$$

If the result of this probability calculation is larger than or equal to 0.5, the input word is classified as Password. Otherwise, the input word is classified as Word.

6.2 Support Vector Machine (SVM) classification

A Support Vector Machine is a machine learning binary classification algorithm. In a field with n dimensions, the algorithm tries to find a $n-1$ dimensional hyperplane as a binary separator. This hyperplane is called a linear classifier. For this research we use the SVM classifier from the scikit-learn library for python[3].

7 Evaluation of classifiers

In this section, we will explain what metrics we use to evaluate the classifiers and then apply these metrics to the trained classifiers. To evaluate the classifiers, for each classifier we split the password and the word list up in a training set and a test set. These sets do not have overlapping data. This prevents overfitting.

7.1 Metrics

We ultimately use the F1 score to evaluate our classifiers. The F1 score is the harmonic mean of precision and recall. The precision and recall are both calculated using the values in the confusion matrix.

7.1.1 Confusion matrix

A confusion matrix is a table layout that visualizes the performance of a classification algorithm, showing True Positives (TP), True Negatives (TN), False Positives (FP) and False Negatives (FN). Table 2 shows the confusion matrix for our classification problem seen from the password class.

		Predicted class	
		Word	Password
Actual class	Word	True Negative	False Positive
	Password	False Negative	True Positive

Table 2: Confusion matrix for password classification

[3]https://scikit-learn.org/stable/modules/svm.html

7.1.2 Precision

The precision, also called the Positive Predicted Value (PPV), is the number of True Positives divided by the number of True Positives plus the number of False Positives.

$$PPV = \frac{TP}{TP + FP}$$

This means, from everything that was classified as positive, how many classifications are correct?

7.1.3 Recall

The recall, also called the True Positive Rate (TPR), is the number of True Positives divided by the number of True Positives plus the number of False Negatives.

$$TPR = \frac{TP}{TP + FN}$$

This means, from everything that should have been classified as positive, how many items were actually classified as positive?

7.1.4 F1 score

The F1 score is the harmonic mean of the precision and recall. It is often used to score how well a classifier performs. The best score is a 1 with perfect precision and recall. The worst score is a 0.

$$F1 = 2 \cdot \frac{precision \cdot recall}{precision + recall}$$

7.2 Evaluation of probabilistic classifier

Table 3 shows the confusion matrix for the probabilistic classifier. Table 4 shows the resulting values for the precision, recall and F1 score. The highest possible F1 score is a 1 so a score of 0.91 is a very good result for a classifier using only statistics combined with basic probability theory.

		Predicted class	
		Word	Password
Actual class	Word	886	115
	Password	69	925

Table 3: Confusion matrix for probabilistic classifier

7.3 Evaluation of SVM classification

Table 5 shows the confusion matrix for the SVM classifier. Table 6 shows the resulting values for the precision, recall and F1 score.

	Precision	Recall	F1 Score
Word	0.93	0.89	0.91
Password	0.89	0.93	0.91

Table 4: Precision, recall and F1 score for the SVM classifier

		Predicted class	
		Word	Password
Actual class	Word	4917	458
	Password	1312	3641

Table 5: Confusion matrix for SVM classifier

	Precision	Recall	F1 Score
Word	0.79	0.91	0.85
Password	0.89	0.74	0.80

Table 6: Precision, recall and F1 score for the SVM classifier

What we can see here is that the classifier classifies a password too often as a word. This results in a bad recall score for classifying passwords. The SVM classifier does not make up for this error with a significantly higher recall score for words.

A possible explanation for SVM performing worse than the probabilistic classifier

8 Conclusion

Our research question was: *How can software be used to classify whether a string is a password or a "normal" word?*

During this research we defined characteristics which can be used to classify passwords. Using these characteristics, we created statistics for a password list and a word list. We then created a probabilistic password classifier and compared it with with an existing SVM implementation.

We found that the simple probabilistic approach to password classification using basic probability theory and statistics performs surprisingly good at classifying passwords with an F1 score of 0.91.

The machine learning tool commonly used for text classification performs significantly worse with a F1 score of 0.80 to 0.85.

9 Discussion

We realize that the resulting scores are very dependent on the chosen training set and test set.

The password list we used did not contain any duplicates. If the list would have duplicates for more common passwords, the list would be more effective.

We were unable to find the source of the passwords within the breach compilation. Thus, we can not guarantee its authenticity. However, our classification tool can be easily trained with new input data which can be chosen for specific situations.

10 Future work

To increase the performance of the classification algorithms, it would be interesting to use more and different characteristics. A relatively easy addition would be to add the location of certain types of characters within the string as a characteristic. We would expect numbers in passwords to occur more often at the end of a word. More advanced characteristics could also be added like word categories (love, animal, car, ...).

Another idea for future research is to find a more optimal classification method. Example machine learning classifiers that might be interesting to look at are: Decision trees, Neural networks, Random Forests or SVM with a non-linear kernel.

11 Ethical notions

Passwords can contain personal information but the password list used in this research is a public list of most used passwords and thus they do not contain personal identifiable information. We did not use the password data for anything else than the research conducted. The resulting tool is developed to get an answer to my research question and to support the "Nederlands Forensisch Instituut". It could however, also be used by adversaries to find passwords in large datasets. Publication of the tool will happen in contact with the ethics committee of OS3 and my supervisor Zeno Geradts.

References

[1] Joseph Bonneau. The science of guessing: analyzing an anonymized corpus of 70 million passwords. In *Security and Privacy (SP), 2012 IEEE Symposium on*, pages 538–552. IEEE, 2012.

[2] Matteo Dell'Amico, Pietro Michiardi, and Yves Roudier. Password strength: An empirical analysis. In *INFOCOM, 2010 Proceedings IEEE*, pages 1–9. IEEE, 2010.

[3] Ate Kloosterman, Anna Mapes, Zeno Geradts, Erwin van Eijk, Carola Koper, Jorrit van den Berg, Saskia Verheij, Marcel van der Steen, and Arian van Asten. The interface between forensic science and technology: How technology could cause a paradigm shift in the role of forensic institutes in the criminal justice system. *Philosophical Transactions of the Royal Society B: Biological Sciences*, 370(1674):20140264, 2015.

[4] Zhigong Li, Weili Han, and Wenyuan Xu. A large-scale empirical analysis of chinese web passwords. In *USENIX Security Symposium*, pages 559–574, 2014.

[5] Jerry Ma, Weining Yang, Min Luo, and Ninghui Li. A study of probabilistic password models. In *Security and Privacy (SP), 2014 IEEE Symposium on*, pages 689–704. IEEE, 2014.

[6] William Melicher, Blase Ur, Sean M Segreti, Saranga Komanduri, Lujo Bauer, Nicolas Christin, and Lorrie Faith Cranor. Fast, lean, and accurate: Modeling password guessability using neural networks. In *USENIX Security Symposium*, pages 175–191, 2016.

[7] Darren Quick and Kim-Kwang Raymond Choo. Impacts of increasing volume of digital forensic data: A survey and future research challenges. *Digital Investigation*, 11(4):273 – 294, 2014.

[8] Rafael Veras, Christopher Collins, and Julie Thorpe. On semantic patterns of passwords and their security impact. In *NDSS*, 2014.