Cipher Type Detection

Malte Nuhn* and Kevin Knight†

*Human Language Technology and Pattern Recognition, RWTH Aachen University
†Information Sciences Institute, University of Southern California

RWTH AACHEN UNIVERSITY

Overview

Manual analysis and decryption of enciphered documents is a tedious and error prone work. Often—even after spending large amounts of time on a particular cipher—no decipherment is found.

Automating the **decryption** of various types **of ciphers** makes it possible to sift through the large number of encrypted messages found in libraries and archives, and to focus human effort only on a small but potentially interesting subset of them.

We **train a classifier** that is able to **predict which encipherment method has been used** to generate a given ciphertext.

Problem Definition

Each encipherment method transforms a given plaintext into a ciphertext using a given key. For unknown ciphers, we are interested in the original plaintext.

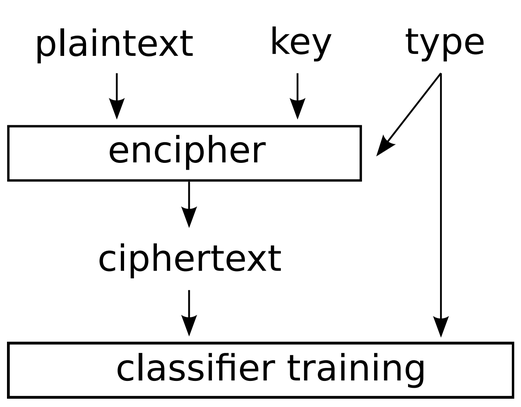
Obtaining the plaintext from an enciphered message is difficult. We assume that the decipherment of a message can be separated into three steps: **finding the encipherment method**, **finding the key**, and the actual **decoding**.

In this paper, we focus on identifying the encipherment method:

Given an unknown ciphertext, predict what kind of encryption method was used to generate it.

Machine Learning Approach

We take a machine learning approach which is based on the observation that **we can generate an infinite amount of training data**. Based upon a large English corpus, we first **choose possible plaintext messages**. Then, for each encipherment method, we choose a **random key** and **encipher each of the plaintext messages** using the encipherment method and key. We then train a classifier on this data. We do this for 50 cipher types defined by the American Cryptogram Association (ACA).



Classifiers

We train an **SVM using the libSVM** toolkit. This is feasible for up to 100k training examples. In order to use more training data, we use **Vowpal Wabbit to train a linear classifier** that is trained with up to 1M training examples.

Features

On top of 55 previously published features for cipher type detection, we add three newly developed sets of features.

Repetition Feature (`REP`): How often does the ciphertext contain symbols that are repeated exactly n times in a row? Add features for repetitions of length $2 \leq n \leq 5$.

Amsco Feature (`AMSC`): Apply all possible permutation patterns used in the AMSCO cipher, and measure perplexity of the resulting text in a bigram LM.

Variant Feature (`VAR`): Calculate individual unigram count statistics for specific parts of the cipher text. Compare these frequency profiles against the frequency profile of english.

Example Cipher

In general the key of an encipherment method can consist of more than just a codeword:

Type:

```
CMBIFID
```

Plaintext:

```
WOMEN NSFOO TBALL ISGAI NINGI
NPOPU LARIT YANDT HETOU RNAME
```

Key:

```
LEFTKEY='IACERATIONS', RIGHTKEY='KNORKOPPING'
PERIOD=3, LROUTE=1, RROUTE=1, USE6X6=0
```

Ciphertext:

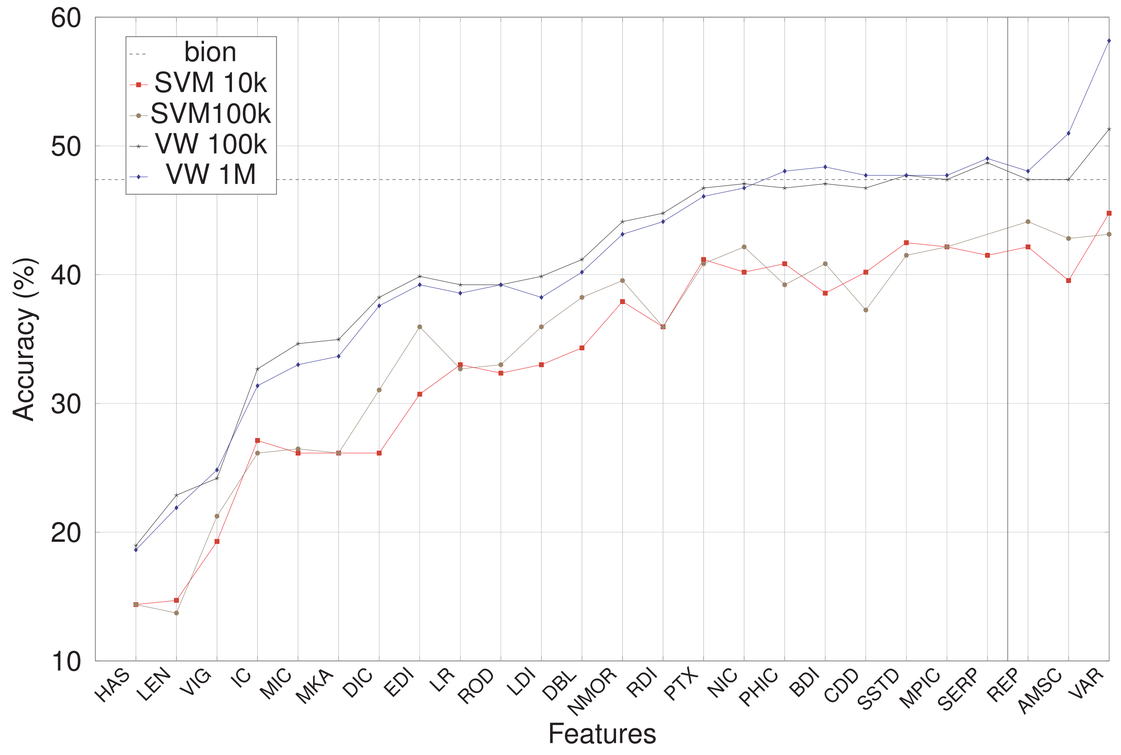
```
WTQNG GEEBQ BPNQP VANEN KDAOD
GAHQS PKNVI PTAAP DGMGR PCSGN
```

Supported Cipher Types

- 6x6bifid
- 6x6playfair
- amsco
- bazeries
- beaufort
- bifid6
- bifid7
- cmbifid
- columnar
- digrafid
- dbl chckrbrd
- four square
- fracmorse
- grandpre
- gromark
- gronsfeld
- homophonic
- mnmedinome
- morbit
- myszkowski
- nicodemus
- nihilistsub
- patristocrat
- period 7 vig.
- periodic gromark
- phillips
- plaintext
- playfair
- pollux
- porta
- portax
- progkey beaufort
- progressivekey
- quagmire2
- quagmire3
- quagmire4
- ragbaby
- randomdigit
- randomtext
- redefence
- runningkey
- seriatedpfair
- swagman
- tridigital
- trifid
- trisquare
- trisquare hr
- two square
- two sq. spiral
- vigautokey

Results

We identified the best classifier on a held-out set of 1000 ciphers, i.e. 20 ciphers for each cipher type. Here our three **new features** improve the VW-1M classifier from 50.9% accuracy to 56.0% accuracy, and the VW-100k classifier from 48.9% to 54.6%. The figure below, shows the results of our method on the **completely independently created ACA test set**:



e-mail: *<surname>@cs.rwth-aachen.de, †<surname>@isi.edu