Crypto Wallet Artifact Detection on Android Devices Using Advanced Machine Learning Techniques

Abhishek Bhattarai(✉), Maryna Veksler, Hadi Sahin, Ahmet Kurt, and Kemal Akkaya

Electrical and Computer Engineering Department, Florida International University, Miami, FL 33174, USA
{abhat031,mveks001,asahi004,akurt005,kakkaya}@fiu.edu

Abstract. As cryptocurrencies started to be used frequently as an alternative to regular cash and credit card payments, the wallet solutions/apps that facilitate their use also became increasingly popular.

This also intensified the involvement of these cryptowallet apps in criminal activities such as ransom requests, money laundering, and transactions on dark markets. From a digital forensics point of view, it is crucial to have tools and reliable approaches to detect these wallets on the machines/devices and extract their artifacts. However, in many cases forensic investigators need to reach these file artifacts quickly with minimal manual intervention due to time and resource constraints. Therefore, in this paper, we present a comprehensive framework that incorporates various machine learning approaches to enable fast and automated extraction/triage of crypto related artifacts on Android devices. Specifically, our method can detect which cryptowallets exist on the device, their artifacts (i.e., database/log files), the crypto related pictures and web browsing data. For each type of data, we offer a specific machine learning technique such as Support Vector Machine, Logistic Regression and Neural Networks to detect and classify these files. Our evaluation results show very high accuracy detecting the file artifacts with respect to alternative tools.

Keywords: Crypto wallet · Cryptocurrency artifacts · Triage · Forensics · Machine learning · Android devices

1 Introduction

The adoption of the cryptocurrency has been expanding in the last decade. Bitcoin [27], originally introduced in 2009, has proposed a cash like payment system that relied on digital elements and cryptography but also have the characteristics of physical cash. The convenience of performing financial transactions in

M. Veksler, H. Sahin and A. Kurt—These authors contributed equally.

S. Goel et al. (Eds.): ICDF2C 2022, LNICST 508, pp. 111–132, 2023.
https://doi.org/10.1007/978-3-031-36574-4_7

a decentralized, secure, and peer-to-peer manner made cryptocurrencies popular and a preferable way over traditional payment methods. With this, multiple cryptocurrencies have also been introduced with various features and market capitalization such as Ethereum [33] and Monero [30].

The cryptocurrencies are held in *cryptocurrency wallet* applications and can be used in financial transactions with the associated *private key* of the wallet. Wallets can communicate with the underlying blockchain software allowing users to send and receive cryptocurrencies with ease. Crypto wallets generate different types of the supporting files within the device of the deployment, called artifacts, that include log files, databases, mnemonic files. In this work, we focus on mobile Android devices, as it leads the current market share with 71.85% worldwide [23].

The convenience for using the cryptocurrencies and lack of oversight made them attractive for criminal activities such as money laundering, drug trade and tax evasions. Therefore, the artifacts acquired from wallet applications are crucial in digital forensics investigations. Specifically, recovered data is useful to trace illegal money flows and recovery of stolen money. Wallet applications generally record transaction history on the device storage which may include cryptocurrency wallet addresses, timestamps, sent and received transaction amounts. The most critical artifacts include *private key*, *mnemonic*, and *seed files*. The law enforcement uses the obtained data in the attempts to unlock the wallet and access the funds.

Thus, the accurate retrieval of wallet's artifacts is crucial. However, the current approaches lack generalization, while tools such as Cellebrite require thorough manual analysis of the extracted data. Moreover, the various types of data within the phone may significantly contribute to the investigation. For example, the screenshots and notes may contain seed phrases, while browsing activity reveals the suspect's intentions.

In this paper, we propose an automated tool for the detection of crypto wallets and crypto artifacts. Moreover, we design a tool for the investigators to analyze extracted data efficiently. Proposed framework consists of 3 components, (1) dynamic detection of crypto wallet applications using machine learning (ML) classifiers, (2) image analysis to identify the presence of crypto related information using deep learning (DL) techniques, and (3) image analysis to identify the presence of crypto related information using deep learning (DL) techniques.

The results indicate that our crypto wallet detection method outperforms Cellebrite Crypto Tracer solution with the accuracy of 100%. We also achieve 98% accuracy for crypto related images classification. We demonstrate that our browsing activity analysis is robust across various browsers, unlike Cellebrite, and allows to accurately identify all crypto related artifacts, including history, bookmarks, and cache.

The rest of this paper is organized as follows. First, we discuss the related work in Sect. 2. In Sect. 3, we present our automated framework for crypto wallet mobile forensics and describe its components. We provide the implementation details and evaluation results in Sect. 4. Finally, we conclude the paper in Sect. 5.

2 Related Work

The majority of the works targeting crypto wallet forensics heavily rely on Cellebrite [1] software. Being the most popular digital forensics tool, it allows to extract all artifacts present within the device using Cellebrite UFED and Physical Analyzer suits. Moreover, Cellebrite "CryptoCurrency Analyzer" interface specifically focuses on identifying crypto wallet application. However, we determined that this software is not dynamic and cannot detect recently released applications. Specifically, Cellebrite successfully detected "Trust Wallet", but not "CryptoWallet PRO: Earn Crypto" applications. Despite Cellebrite's ability to extract all of the artifacts present within the phone, it fails to identify and filter out newly added crypto-related data, thus requiring a significant manual effort.

In [26], the authors used Cellebrite to analyze Bitcoin, Litecoin, and Darkcoin artifacts on both Android and iOS mobile devices. They focuses on the application installation and deletion dates, and preformed crypto transactions. As a results, five folders containing the artifacts were extracted and analyzed. Chang et al. in [6] conducted a more detailed analysis of cryptocurrency related information (e.i., transactions, timestamps, emails, and browser cookies). Moreover, the authors used Ciphertrace software [2] to examine the extracted data. They demonstrated the structural differences in the crypto wallet artifacts. However, their analysis was limited to two coins, Bitcoin and Dogecoin, and three wallets, Coinomi, Atomic, and Coinbase.

The authors in [21] focused on detecting wallet artifacts for Exodus and Electrum applications within Linux operating system. This work point out the weaknesses of data protection and tools for exploring wallet structure and its artifacts. As a result, the authors proposed a command line interface (CLI) tool for the evidence analysis.

The categorization of the cryptocurrency application in a phone accelerates the forensic investigation. This can be achieved by analyzing each application's description to understand its functionality.

Qiang Xu et al. [34] presented a framework that automate categorization of health and fitness applications gathered from Apple's apps and Google's play stores. The applications were first classified into two broad categories, paid and free, and then further sub-classified into 11 categories using supervised learning.

In [15] authors proposed a game apps classifier using the Latent Semantic Indexing. They classify various genres of games applications in the Apple App Store using support vector machine (SVM) classifier with a mean accuracy of 77%. [10] focuses on categorization of mobile apps into more focused categories based on their functionalities. The author categorized 600 Apple App store application from Education, Health and Fitness, and Medical categories. This paper demonstrates that metadata such as installs and ratings does not necessarily improve classification, but information from the title and the description of the application can increase accuracy.

Text categorization method allows to extract information from the text and assign it to the specific predefined categories based on the content (e.i. spam and genuine emails; rain, humid, and sunny). In digital forensics, natural language

processing (NLP) text categorization is widely used to understand the context of the collected evidence [31]. It allows to reduce the manual workload required to manually analyze thousands of text artifacts present on the device.

One of the most popular and effective NLP approaches is Term Frequency-Inverse Document Frequency (TF-IDF) [16]. Dzisevic et al. [9] employs three information retrieval methods on both large and small datasets: plain TF-IDF, modified TD-IDF with LSA (Latent Semantic Analysis) and Linear Discriminant Analysis (LDA). The paper concludes that on both datasets, plain TF-IDF achieve similar accuracy with the other two methods.

3 Framework for Cryptocurrency Wallet Application Analysis

In this section we provide an overview of the proposed automated triage framework for crypto wallet forensics.

3.1 Overview

Figure 1 illustrates the main components of our system - android file system, image, and browser activities. First, we examine the android file system to classify installed crypto wallet applications and extract relevant data such as transaction history, mnemonic codes, and seed phrases. Next, we analyze the images on the device to identify those containing important crypto data (e.i, screenshots of QR codes and photos of seed phrases). Finally, we extract relevant history, bookmarks, cookies, and cache files from the browser activity. We use Cellebrite UFED software to extract the data from Android phones. Specifically, we select Qualcom Live extraction as recommended for the devices using Qualcomm mobile platform.

3.2 Android File System Analysis

The android file system analysis is the first component of our framework. Figure 2 illustrates the steps of the algorithm for crypto wallet applications detection.

First, we identify the Google Play Store ID for each application installed within the device. The names of the folders located in /data/data folder corresponds the application IDs.

We use FuzzyWuzzy string matching Python library package to calculate the differences of the given sequence of strings [7]. We compare discovered applications against those in the local database, which contains the list of the application IDs previously labeled as crypto wallet apps (①). The maintenance of the local database allows to optimize classification and prevent the algorithm from analyzing the same application twice. Subsequently, we identify the application not present in the database for the further categorization analysis (②).

For each unidentified application, we search for its information in the Google Play Store. We use the application ID to obtain the app description using Google-Play-Scraper library [17] (③).

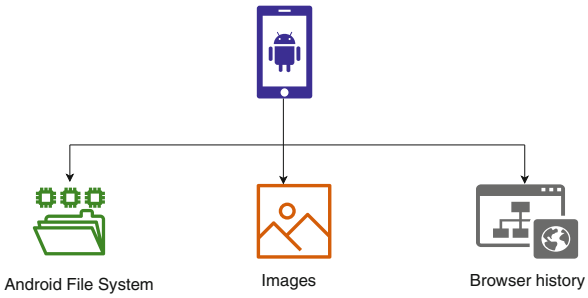

Fig. 1. Components of cryptowallet artifacts that will be analyzed under our proposed framework.

Next, we apply *keyword filtering* (④) using FuzzyWuzzy module in the extracted description text. We pre-compiled the list of keywords (i.e., crypto, coin, bitcoin, etc.) based on the existing crypto wallet applications in Google Play; and added the functionality to expand it as required. The keyword search allows to clean the data before passing it to the our classifier. Thus, we filter out the non-crypto applications such as Facebook to reduce the processing time of the model and increase the classifier's accuracy. All of the filtered-out applications are added to the local database and marked as "0" (i.e., Not Crypto wallet).

For the application that pass *keyword filtering* stage, we apply *preprocessing techniques* (⑤) to reduce the various noises present in the applications' descriptions. The unreadable format, special characters, parentheses, white spaces, brackets, punctuation, digits, and emoticons noises are removed, and the text is converted to lowercase character. We then tokenize the text by splitting the corpus of words using the tokenizer class of Natural Language Toolkit (NLTK). To reduce the dimensionality, we remove the stop words that do carry no significance, such as a, an, but, before. Finally, each word is reduced to its root form and plural or tense forms are removed. For example, made is converted to make.

As ML classifier model is not able to accurately process the resulting application description text, we apply *feature extraction* technique (⑥). We use TF-IDF [16] to extract the significant words from the original text. We maintain the vocabulary created based on the local database, which contains Term Frequency (TF) and Inverse Document Frequency (IDF) and values for each term calculated using Eqs. 1 and 2.

$$TF(t, d) = \frac{\text{Number of each } t \text{ in } d}{\text{Total number of } t \text{ in } d} \tag{1}$$

$$IDF(t) = \log\left(\frac{\text{Number of total } d}{\text{Numbers of } d \text{ containing } t}\right) \tag{2}$$

If t is the term and d is the document, then TF-IDF score is calculated as:

$$TF\text{-}IDF \text{ score} = TF * IDF \tag{3}$$

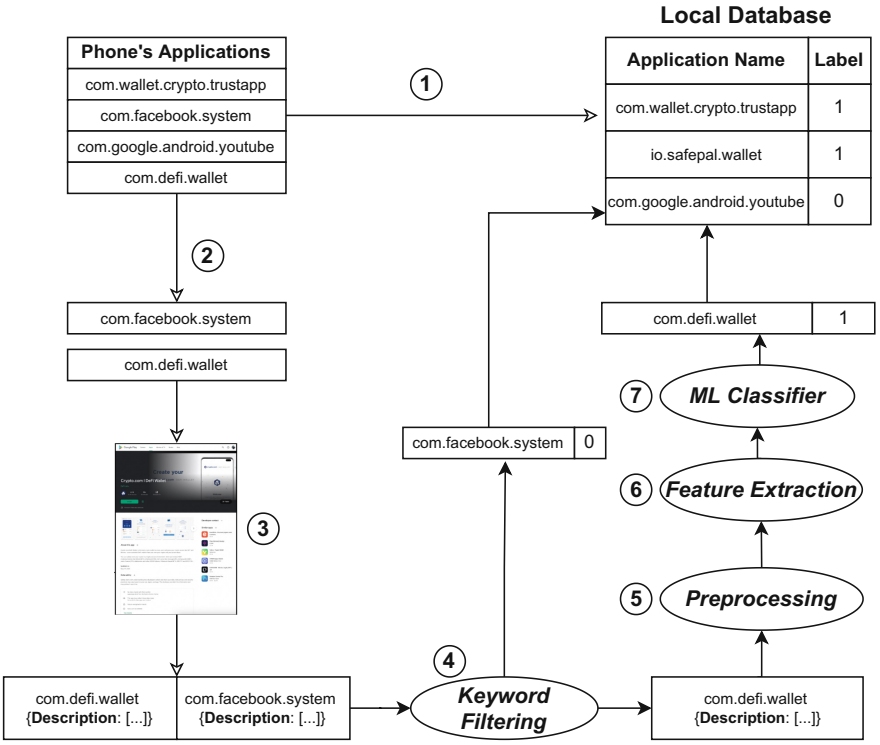

Fig. 2. Classification of the crypto wallet

Finally, we feed the extracted features to *ML classifier* (⑦). As a result, the model predicts the application as belonging to one of three categories - Crypto exchange, Crypto portfolio, and Crypto wallet. Crypto exchange denotes the applications that provides platform for the exchange or trading of crypto coins. Crypto portfolio category is for applications that manages the crypto assets or cryptocurrency profile. Crypto wallet category is for the applications that store the private keys of users and helps sending and receiving digital currency. While our main focus is crypto wallets, we added two specific categories to provide our model with an ability to learn more specific semantic features extracted from apps' descriptions. Based on the classifier output for the description, the application is marked as "1" (Crypto wallet) or "0", before being added to the local database.

3.3 Detecting Crypto Related Images

We use a combination of ML and fuzzy search methods to locate and extract crypto related information from images. Our approach is the first to offer a comprehensive image analysis for cryptocurrencies by integrating image filtering, optical character recognition, handwritten text recognition, text extraction and fuzzy search. Figure 3 provides the overview of our algorithm.

First, we filter out graphics used by mobile applications for the user interface (UI), namely the icons. Mostly located in the application folder, they are of PNG format and characterized by small resolution and size. Next, using Neural Networks (NNs) we classify the images into three categories; images with no text, images with printed-text and images with handwritten-text. We then use optical character recognition (OCR) engine to extract text information from images with printed text. We employ Handwritten Text Recognition (HTR) model using Convolutional Neural Networks (CNNs) and Recurrent Neural Networks (RNNs) to extract text information from images with handwritten-text. Finally, we implement fuzzy search to locate crypto related information and wallet recovery seeds from the extracted texts.

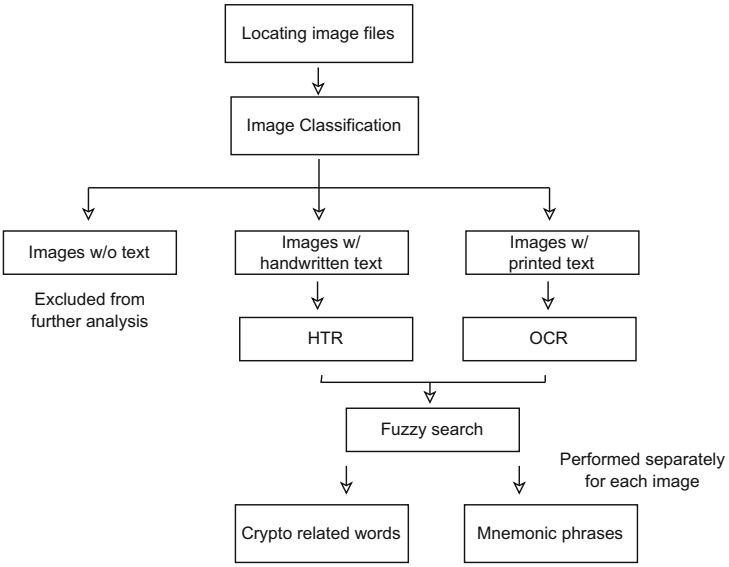

Fig. 3. Steps of image analysis

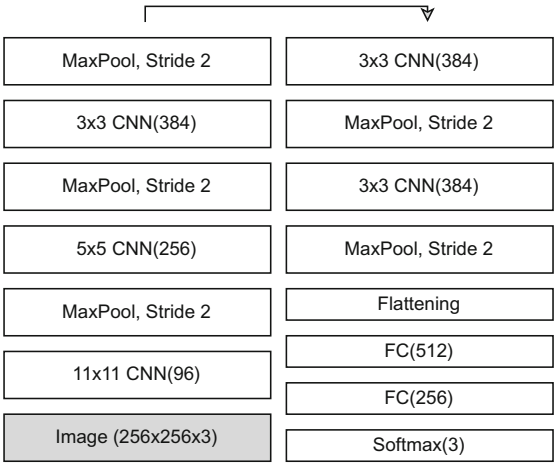

Fig. 4. Model schematics for image classification

Text/Non-Text Image Classification. Our Neural Network architecture, displayed in Fig. 4, implements a modified version of AlexNet [22].[1] Our model includes five CNN modules each followed by a maximum pooling (MaxPool) layer, and three fully connected layers (FC). Using the model we classify images into images with no text, images with printed-text and images with handwritten-text. We remove images with no text from further analyses.

Handwriting Recognition. Before running a handwriting recognition model (HTR), we use text-line and word detection/segmentation techniques to delineate lines and then words in the images. These word segments are then saved as separate images and fed to the trained model for prediction. The segmentation technique focuses on spatial distances between dark/black pixels and puts the connected components into bounding boxes [13]. Although most of the state of the art HTR models use them, it should be noted they are open to error as handwriting style and orientation could change significantly across individuals and images. There are studies which calculate line and word segmentation dynamically and incorporate them into their ML models (See [29,32]). However, they can be computationally expensive as they analyze the whole page rather than word segments [35].

We adapted the model created by [18]. The model architecture is displayed in Fig. 5. It includes two CNN models followed by maximum pooling layer, a fully connected layer, and two bidirectional Long Short-Term Memory (LSTM) layers with 25% dropouts. The model employs Connectionist Temporal Classification (CTC) loss function. CTC is extensively used in sequence labelling for speech

[1] AlexNet has won ImageNet Large Scale Visual Recognition Challenge (ILSVRC) in 2012, and since then has been extensively employed in visual recognition and classification tasks.

and handwriting recognition as it reduces the need for presegmentation of the input and postprocessing [12].

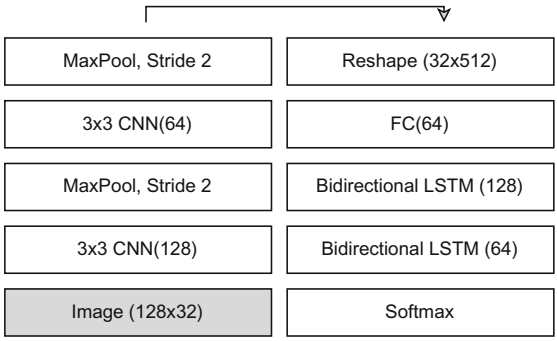

Fig. 5. Model schematics for handwriting recognition

3.4 Detecting Crypto Related Browser History

Web browser activity is an important component of the phones' forensic analysis. Visited websites, search history, and bookmarks can be used to build a detailed profile of user's interests. However, the existing research heavily relies on the manual inspection of the web browser files. Moreover, the investigators are often required to target a specific Web browser, one at a time, resulting in large time effort. Therefore, we propose using a triage approach which aims to recover web browser artifacts that unveil crypto-related activities.

In the analysis, we focused on recovering and identifying the following forensics artifacts: web history, bookmarks, user credentials, cookies, and cache. Web history of the Internet browsers information is generally stored in SQLite format and contains timestamps, urls, search terms, and downloads information. It reveals the interest of the subject and indicates all visited crypto-related websites if any. Similarly, session data can be used to restore users web activity, as it contains a record about open tabs and visited websites.

Bookmarks and user credentials provide more insight into recurring web browsing activity. Moreover, the credentials information allows to pinpoint the suspect's identity and/or aliases if any. The browser cookies are used to store information such as user preferences, session information, and personal information, thus, being an additional source for digital profiling of a suspect. Finally, the cache stores images, strings, and scripting data of the visited websites, which is normally used to accelerate the time required for the website loading. Therefore, cache contains the information to support and extend the browsing history evidence.

In our approach, we focus on the most popular Android web browser applications, Chrome, Opera, Mozilla Firefox, and Samsung Internet. On Android system, the web browsing artifacts are located in the web browsers' applications fold-

ers shown in Table 1. However, the structure of the stored information is not constant across various applications, and varies in the folder hierarchy and file types.

Table 1. The data folders of popular web browsers application and user data.

Web Browser	Folder Name
Mozila Firefox	`/data/data/org.mozilla.firefox`
Samsung Internet	`/data/data/com.sec.android.app.sbrowser`
Opera Browser	`/data/data/com.opera.browser`
Google Chrome	`/data/data/com.android.chrome`

For example, the files containing the browsing history are generally located inside *"Default/History"* folder, while the other artifacts are widely spread out among the browsers data folders. At the same time, the type of the files storing relevant artifacts varies highly from SQLite database to binary formats. For example, the Bookmarks for the Google Chrome browsers are stored as a JSON dictionary, while Samsung Internet stores this information in SQLite database file.

In this work, we designed an automated tool to read and analyze Android browser applications' files using the string-searching algorithm. It operates by traversing all files within the web browser's directory and then applying the search to identify and extract the information relevant for the investigation. We define a starting vector of the cryptocurrency terms consisting of the 3 categories: (1) names of the websites for cryptocurrency exchanges; (2) names and abbreviations of the most popular cryptocurrencies; and (3) frequently used cryptocurrency glossary. The first category contains the list of 50 most used cryptocurrency applications based on the scores derived from the traffic and trading volumes.[2] Similarly, we select 50 cryptocurrencies based on the highest exchanged volume[3]. Finally, we use publicly available cryptocurrency glossaries to complement our string vector by the terms that indirectly indicate the connection to crypto wallets [8,14]. The match cases for each Android browser are then extracted and presented to the investigator as illustrated by Table 2.

Table 2. Results of crypto related web artifacts discovered using the automated triage approach

Data Type	Data	File Location
URLs	http://crypto.com, crypto.com, The Best Place to Buy, Sell, and Pay with Cryptocurrency	`/app_sbrowser/Default/History`
Cookies	crypto.com, /price	`/app_sbrowser/Default/Cookies`
Bookmarks	https://www.coinbase.com, Coinbase - Buy and Sell Bitcoin, Ethereum, and more with trust	`/databases/SBrowser.db`

[2] https://coinmarketcap.com/rankings/exchanges/.
[3] https://coinmarketcap.com/all/views/all/.

4 Evaluation

In this section, we present the details of out implementation and evaluation of our proposed framework. First, we describe the phone data setup and extraction. Then, we analyze each component separately and compare it against Cellebrite.

4.1 Data Extraction

For the experiments, we select three different Android phones: Samsung Galaxy Note 10+, Xiaomi Mi 10T 5G, and Google Pixel 5a. For each phone, we installed six cryptocurrency applications from different categories (i.e., Crypto Wallet, Crypto Portfolio/Tracker, and Crypto Exchange) listed in Table 3. In our evaluation we included the most popular applications according to Google Play Download stats (e.g., Trust (10M+), Crypto Tracker (1M+), Binance (50M+)), and the most recently added (e.g., D-Wallet, Moni, KoinBasket).

To test framework components for crypto related images detection, we took multiple pictures of handwritten seed files. Finally, we browsed cryptocurrency websites using Google Chrome, Android Browser, Opera, and Firefox applications. Table 4 lists a full list of the crypto related artifacts created on the phone of the interest. Once we generated the realistic data on each device, we used Cellebrite UFED Qualcomm Live to extract phone data.

Table 3. Summary of crypto applications installed on the devices of the interest

Wallet	Portfolio/Tracker	Exchange
Trust	Crypto Tracker	Binance
Coinbase Wallet	HODL Real-Time Crypto Tracker	FTX
Exodus	Hodler	KuCoin
Crypto.com — DeFi Wallet	CoinFolio	WazirX
CryptoWallet PRO: Earn Crypto	Coin Portfolio	CoinDCX
D-Wallet	Moni	KoinBasket

Table 4. Summary of the crypto related artifacts present on the phone.

Bookmarks	Visited Websites	Searches
freewallet.org	freewallet.org	cryptowallets online
nerdwallet.com	coinbase.com	set up anonymous cryptowallet
coinbase.com	cryptowallet.com	untraceable online crypto transactions
	nerdwallet.com	
	binance.com	

4.2 Crypto Wallet Application Analysis Results

Creation of Dataset for Crypto Wallet Apps and Training: We trained our application ML classifier using manually created dataset. First, we downloaded a Google Play Store dataset of 2.3 million+ application from Kaggle.com [28]. Then, we reduced the original dataset to finance genre only and then applied Python script to select applications that include the following keywords, *crypto*, *bitcoin*, *blockchain*, *coins*, and *cryptocurrency*. The final dataset included more than 4000 cryptocurrency applications. These applications were then labeled manually into predefined categories based on the information provided in their descriptions. Initially, we identified 12 categories ranging from crypto wallet to banking but later reduced it to three.

As a result, we removed the application that completely irrelevant to the crypto wallets and categorize the rest as crypto wallet, crypto portfolio and crypto exchange. After reviewing the categories, we merged portfolio and tracker apps into one category - crypto portfolio. Our final dataset consisted of 720 applications with the following sample distribution, 281 wallets, 231 portfolios, and 209 exchanges.

Training ML Classifiers: We applied TF-IDF feature selection approach to prepare the dataset for the model training. Thus, for each word in the applications descriptions we obtained TF-IDF score using Eq. 3 such that the higher score indicated more significance of a word in a given text. Next, we established the feature vectors comprising of the TF-IDF values. We split the obtained dataset in train and test sets in 80/20 relation. To determine the best performing ML Classifier, we tested four classifiers Random Forest (RF), Support Vector Machine (SVM), Naive Bayes (NB), and LogisticRegresssion (LogReg). We then tunned maximum and minimum thresholds, unit normalization, and n-grams boundary parameters for TF-IDF as follows **min_df** (0.5), **max_df** (3), **norm** ('l1'), **ngram_range** ((1, 3)). We trained each ML model using 5 cross-validation folds (K-folds).

Performance Results: Table 5 illustrate the results for 2 best performing ML classifiers: SVM and LogReg. We selected, precision, recall, and F1 score metrics to determine the optimal algorithm. Since our main focus is on crypto wallets, we focus on analyzing the performance of ML algorithm for this category exclusively. Table 5 illustrates the results for our models, where **precision** indicates the number of positive class that actually belongs to positive class, **recall** means ratio of true positive class correctly predicted to the total positive class, and **F1 score** is a weighted average of both (e.i., F1 score interprets the overall performance of the classifier).

The results demonstrate that both SVM and LogReg algorithms allows to achieve the recall of 93%. It indicates that both models identified majority of crypto wallet applications. However, SVM slightly outperforms LogReg for precision metrics, indicating that the former algorithm produces less false positive results. We use F-1 score to asses the overall performance of the model as 85% contributed to the relatively high percentage of false positives (22%). In con-

Table 5. Classification results of classifiers models for Crypto Wallets.

	SVM			LogReg		
	Precision	*Recall*	*F1 Score*	*Precision*	*Recall*	*F1 Score*
Crypto Wallet	**0.78**	***0.93***	**0.85**	0.76	***0.93***	0.84

clusion, we identify SVM as an optimal model. We attribute the low precision of ML algorithms to the fact that exchanges functionality is similar to crypto wallets. Particularly, both applications allow to store, receive, and send crypto, which leads to a similar description semantics.

End-to-End Evaluation of Crypto Wallet Categorization Component: We proceeded to test the performance of our algorithm for Android file system analysis. First, we identified the IDs of all applications present on the phone. As a results of description extraction and keyword filtering, we obtained 18 distinct application IDs. Next, we applied ML classifier preceded by text preprocessing and feature extraction.

We were able to classify all applications listed in Table 3 with the F1 score of 100%. Specifically, we identified six wallets, Trust, Coinbase, Exodus, DeFi, CryptoWallet PRO, and D-Wallet (Table 6).

Table 6. Classification results for the applications installed on the test phones.

	Precision	*Recall*	*F1 Score*
Wallet	1.00	1.00	1.00
Portfolio	1.00	1.00	1.00
Exchange	1.00	1.00	1.00
Accuracy			1.00

All applications discovered by our algorithm were added to the local database, such that apps filtered out by keyword search, crypto exchanges, and crypto portfolios were marked as non-crypto, while crypto wallets marked as crypto related. Therefore, the investigators can save a significant amount of time if any of the identified applications will appear on the Android devices in the future.

Comparison with Cellebrite: We compared our approach to the Cellebrite tool. Based on the experiments, we determined that Cellebrite did not recognize *CryptoWallet PRO: Earn Crypto* and *D-Wallet : Crypto Wallet* application installed on the phone. Thus, we concluded that Cellebrite is not robust against apps newly added to the Google Play Store. At the same time, our framework successfully classified both crypto wallets.

4.3 Crypto Related Image Analysis Results

Text/Non-text Image Classification Setup: We implemented our Neural Network Analysis in TensorFlow [3]. We trained the model on the training set for 50 epochs with Adam optimizer [19] and the default learning rate (α) of 0.001. We employed 9080 colored images which were resized to 256×256 pixels. We kept the original aspect ratio and padded the images if needed. The data were split into train and test sets with a ratio of 90% and 10%. We use several data resources to diversify the training set. Below is the list of the data resources:

i. Non-text image data: 1000 images from ImageNet database [9] and 3002 from Flickr30k dataset [36].
ii. Printed-text image data: 322 images from Old Books dataset [5] and 2997 web screenshots data [4].
iii. handwritten-text image data: 1042 images from IAM dataset [25] and 687 from GNHK dataset [24].
iv. 15 screenshot images created on the examined phones.

We run the model on the test set. It was able to classify the images into three categories with 98.57% accuracy.

Handwriting Recognition Setup: We used Adam optimizer [19] with the default learning rate of 0.001. The model was trained on 96,456 grayscale images for 50 epochs. The data were divided into train, validation and test sets with 90%, 5% and 5% ratios. Images were padded to have a uniform size of 128×32. We used Levenstein edit distance to measure the accuracy of our predictions. This metric compares the real and predicted labels and measures the amount of changes needed to transform the predictions into true labels [20]. Our metric takes the average value of the edit distances. The mean edit distance for our model was 17.36.

After extracting the text information, we pre-processed it for fuzzy search analysis. We removed the white spaces and single character strings, and convert the text to lower case characters. We also created a spelling-corrected version of the text. We run fuzzy search analyses with crypto wallet related keywords on both versions of the text. The keywords include the following terms: *crypto*, *currency*, *wallet*, *bitcoin*, *coin*, *chain*, *btc*, *stake*, *nft*, and *token*. The first five words of the list are commonly used in descriptions of wallet related applications as found in earlier sections.

In addition, we run fuzzy search on a list of mnemonic phrases. Specifically we used BIP39 mnemonic phrases used by Bitcoin.[4] This list includes 2048 standard words; each of these phrases correspond to a number and a combination of these, 12-word phrases, includes all information needed to recover bitcoin wallet. Our goal is to detect whether a phone includes these mnemonic phrases, recovery seeds, in the image files.

Performance Results: We created 15 images which include 6 images with handwritten text, 5 images of phone screenshots with printed-text and 4 images with

[4] https://github.com/bitcoin/bips/blob/master/bip-0039/english.txt.

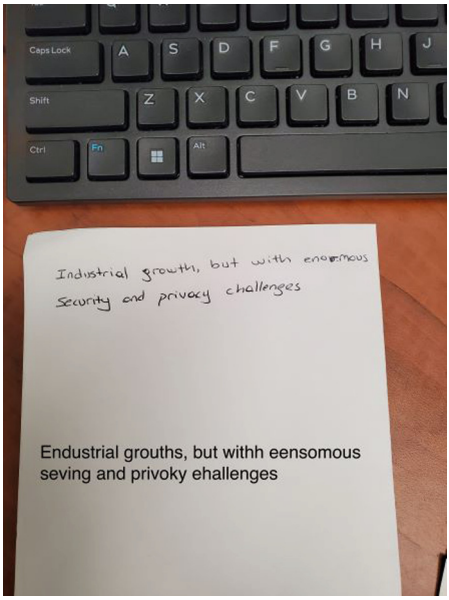

Fig. 6. Image with handwritten text. Extracted text information is included at the bottom of the image.

no text. The handwritten-text images included both the paper the text is written on and the background, as they may appear in the wild (see Fig. 6). The printed text images include various screenshots of the phone menus, crypto-wallet transactions, and QR codes of wallet transactions. These sample images were located in `/data/*` folder.

Text/non-text classification Neural Network model successfully predicted the categories of these 15 images with 100% accuracy. Figure 7 presents the confusion matrix from the results.

Then, we run handwriting recognition model on six handwriting images. Figure 6 displays one of the images with extracted text information at the bottom. As seen from the figure, handwriting recognition may not reliably recognize all words. Therefore, depending on the number of handwritten images, forensic analysts may choose to examine them manually if they found some traces of crypto currency activities in the phone.

Next, we run OCR machine on printed text images. Then, we run fuzzy search on extracted text. We run the analysis with maximum Levenshtein edit distance of one, meaning that we looked at the words that differ from the keywords by one or less alteration. Table 7 presents the crypto-wallet related words extracted from the images. Out of ten keywords our images contained seven crypto currency related words. Our results successfully extracted all these keywords.

We also measured our performance with two commonly used metrics; Recall and Precision rates [11]. Table 8 presents the results. OCR method were able to

Table 7. Results of crypto related word search from images

	No Spelling Check	w/ Spelling Check
bitcoin	'bitcoin'	'bitcoin'
wallet	'wallet'	'wallet'
btc	'btc', 'bitc', 'tc'	'btc', 'bitc', 'tc'
coin	'coin', 'coln', 'coun', 'con'	'coin', 'coln', 'coun', 'con'
chain	'chain'	'chain'
nft	'nft', 'nt', 'net', 'ft', 'nbt'	'nft', 'nt', 'net', 'ft', 'nbt'
token	'token'	'token'

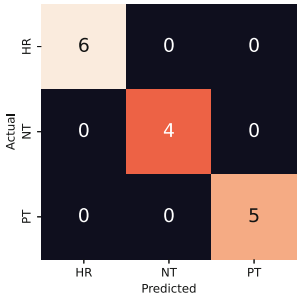

Fig. 7. Results from image classification. PT, NT, HR denote printed text image, non-text image, and handwritten image respectively.

extract printed text with around 90% Recall and Precision rates. HTR method was relatively less successful, Recall and Precision rates were around 65%. But, even at these levels we can comfortably argue whether a phone image includes wallet artifacts or not.

Table 8. Results from search for crypto related words

	# of words	Recall	Precision
Handwritten image 1	12	1.00	0.71
Handwritten image 2	16	0.98	0.64
Handwritten image 3	12	1.00	0.61
Handwritten image 4	9	1.00	0.78
Handwritten image 5	8	0.97	0.75
Handwritten image 6	12	1.00	0.69
Screenshot image 1	12	0.92	0.92
Screenshot image 2	14	0.98	0.98
Screenshot image 3	25	1.00	1.00
Screenshot image 4	10	0.89	0.89
Screenshot image 5	30	0.93	0.93

Table 9 presents results from fuzzy search analysis on mnemonic phrases. We measure success with a modified version of Recall rate. Mnemonic phrases are standard English words. A long text may contain several of them even though the text does not include a recovery seed. Therefore, we penalize the *recall* rate if the text is long. As Eq. 4 shows, we multiply recall rate with a weight which decreases as the number of recognized mnemonic phrases move away from 12.

$$P = \frac{m}{\#\text{ of mnemonic phrases}} \times w \tag{4}$$

$$w = \begin{cases} \exp \frac{\eta - 12}{\eta}, & \eta \leq 12 \\ \exp \frac{13 - \eta}{\eta}, & \eta > 12 \end{cases}$$

where m is the number of extracted mnemonic phrases, w is the weight and η is the total number of extracted words.

Table 9. Results from search for mnemonic phrases

	Includes Mnemonic Phrases	Recall	Requires consideration
Handwritten image 1	True	75.0%	Yes
Handwritten image 2	False	28.4%	No
Handwritten image 3	True	73.6%	Yes
Handwritten image 4	False	26.4%	No
Handwritten image 5	False	39.2%	No
Handwritten image 6	True	70.8%	Yes
Screenshot image 1	False	30.9%	No
Screenshot image 2	False	36.1%	No
Screenshot image 3	False	19.8%	No
Screenshot image 4	False	41.4%	No
Screenshot image 5	False	0.09%	No

In our sample, only three of the images, *Handwritten image 1, 3 and 6* included a recovery seed. We were able to predict them with over 70%. The results show low Recall rates for the images which do not include the phrases, as expected. However, as discussed earlier the list of mnemonic phrases include ordinary words. If we have a longer text, it is possible to get false positives. Therefore, forensic analysts are advised to look closely at cases that have 50% and higher recall rates.

4.4 Browser History Analysis Results

In order to demonstrate the effectiveness of out approach, We evaluated the performance of the tool based on the (1) detection accuracy for crypto artifacts (i.e., the ability to identify all previously created crypto artifacts), (2) robustness

across various browsers (i.e., the ability to extract crypto data from each of the targeted browser), and (3) and time consumption.

First, we analyzed the tool's effectiveness to identify crypto related data. We identified three file types, SQLite database, JSON dictionary, and other. However, due to the large diversity of the file types containing the browsing activity, we identified two stand alone categories as SQLite database and JSON dictionary, while classifying the other types as the plain text. Table 10 contains the summary of all discovered files for each browser filtered by the type.

Table 10. Summary of all discovered files containing the traces of crypto-related activities for the corresponding web browser.

Browser	SQLite Databases	JSON dictionaries	Plain Text
Google Chrome	7	3	745
Samsung Internet	8	2	486
Firefox	12	4	411
Opera	6	0	417

Table 11 lists a summary of the crypto-related forensic evidence extracted for each browser. While not illustrated, the tool also provides the location for each forensics data type, which generally corresponds to a single file per type, except for cache data. The locations for cache data are located in the "/cache/" folder found in the home path of the Android browser applications. However, the type of recovered data varies across the browser applications. For example, Quota Manager data is only applicable for Google Chrome and Samsung Internet applications, while Opera and Firefox apps store the automatically provided browsing suggestions.

Table 11. Summary of the discovered forensics evidence

	Google Chrome	Firefox	Opera	Samsung Internet
History	31	20	20	26
Bookmarks	1	1	1	2
Cookies	42	18	21	32
Favicons	38		N/A	25
Recent Tabs	2	0	0	1
Cache	650	152	310	344
Top Sites	2	1	1	2
Login Data	2	1	1	N/A
Network Persistent State	10	9	7	5
QuotaManager	2	N/A	2	1
Suggestions	N/A	1	9	N/A

To further demonstrate the effectiveness of our approach, we compared it with the Cellebrite analysis. Table 12 illustrates the amount of the artifacts detected by both Cellebrite and our approach based on the initial list of the artifacts in Table 4. Specifically, Cellebrite analysis did not recover the data for freewallet.org and binance.com. Moreover, the Cellebrite analysis did not contain any artifacts from Firefox browser, while only discovering a limited history for the Opera browser. On the contrary, our approach allowed to recover all crypto-related artifacts for each of the four browsers under the investigation. Therefore, our approach outperforms the Cellebrite for the browser history analysis of the crypto related artifacts in the detection accuracy and robustness across all browsers. Finally, when comparing the processing time, our approach finished analyzing the browser artifacts within 10 s compared to the Cellebrite processing the data for approximately 7 min.

Table 12. Amount of recovered crypto-related artifacts discovered by Cellebrite and our approach.

	Cellebrite	Our Approach
Bookmarks	66%	100%
Visited Websites	60%	100%
Searches	100%	100%

Table 13 illustrates the summary of all artifacts detected by our approach. Therefore, our approach was able to successfully identify all visited crypto currency websites and bookmarked pages, recover the Google search information, and partially reveal personal user data for each web browser. Moreover, the cookies and cache data provide a significant evidences to the users activity for each of the crypto-related websites. Also, while revealing the presence of the credentials data for the Firefox, our tool did not recover encrypted login credentials, but only the website it was used for, coinbase.com.

According to our evaluation, our approach for web browser activity provides comprehensive summary of all the artifacts related to the cryptocurrency investigation. Moreover, we were able to robustly extract all manually created artifacts listed in Table 4 for each Android web browser application. The majority of the important crypto activity traces were listed in the format provided in Table 2 allowing us to quickly identify visited websites, bookmarks, and account data. The automated extraction of the data and its further categorization resulted in significant reduction of time and effort required for forensics analysis of the web browsers.

Table 13. Results of crypto related web artifacts discovered using the automated triage approach

	Google Chrome	Samsung Internet	Opera	Firefox
Visited Websites	✓	✓	✓	✓
Google Searches	✓	✓	✓	✓
Bookmarks	✓	✓	✓	✓
Cookies	✓	✓	✓	✓
Cache	✓	✓	✓	✓
Credentials	✓	✓	✓	**X**

5 Conclusion

In this paper, we designed and developed an automated triage framework for Crypto Wallet mobile forensics for Android operating system. We implemented an algorithm to identify crypto wallet applications present on the phone, by leveraging SVM classifier for the contextual analysis of the app's descriptions. We used combination of NNs, CNNs, and LTSMs networks to identify the images containing crypto wallet seed phrases and crypto related information. Our framework also discovered the online traces of crypto activity containing crypto related credentials and browsing history, which allows to support the suspects' intentions and interests.

We evaluated our framework on the data extracted from 3 distinct phones. Our validation result indicate that the framework can successfully classify the crypto wallet applications with the accuracy of 93%. However, in the real-life scenario the proposed approach identified all crypto wallets present on the phones with the precision of 100%. We demonstrated that proposed framework succeeds in identifying images containing handwritten seed phrases with the threshold for the image weight of 70%. Finally, we were able to recover complete crypto related browsing activity. We also established that our tool outperforms Cellebrite for the crypto wallet apps identification and analysis of browsing activity.

For the future work, we will focus on expanding our framework to analyze specific crypto wallet artifacts such as transaction IDs, timestamps, seed files, and keys. We will expand the crypto related image recognition analysis to include the analysis of transaction QR codes and improve the recognition of seed phrases.

Acknowledgement. This work is supported by US National Science Foundation under the grant # 1739805.

References

1. Cellebrite. https://cellebrite.com/en/home/
2. Ciphertrace. https://ciphertrace.com/

3. Abadi, M., et al.: TensorFlow: a system for large-scale machine learning. In: 12th USENIX Symposium on Operating Systems Design and Implementation (OSDI 2016), pp. 265–283 (2016)
4. Aydos, F.: Webscreenshots (2020). https://www.kaggle.com/datasets/aydosphd/webscreenshots
5. Barcha, P.: Old books dataset (2017). https://github.com/PedroBarcha/old-books-dataset
6. Chang, E., Darcy, P., Choo, K.K.R., Le-Khac, N.A.: Forensic artefact discovery and attribution from Android cryptocurrency wallet applications. arXiv preprint arXiv:2205.14611 (2022)
7. Cohen, A.: FuzzyWuzzy (2020). https://pypi.org/project/fuzzywuzzy/
8. CoinMarketCap: Crypto glossary (2022). https://coinmarketcap.com/alexandria/glossary
9. Dzisevič, R., Šešok, D.: Text classification using different feature extraction approaches. In: 2019 Open Conference of Electrical, Electronic and Information Sciences (eStream), pp. 1–4. IEEE (2019)
10. Ebrahimi, F., Tushev, M., Mahmoud, A.: Classifying mobile applications using word embeddings. ACM Trans. Softw. Eng. Methodol. (TOSEM) **31**(2), 1–30 (2021)
11. Faustina Joan, S., Valli, S.: A survey on text information extraction from born-digital and scene text images. Proc. Natl. Acad. Sci. India Sect. A: Phys. Sci. **89**(1), 77–101 (2019)
12. Graves, A.: Connectionist temporal classification. In: Graves, A. (ed.) Supervised Sequence Labelling with Recurrent Neural Networks, pp. 61–93. Springer, Heidelberg (2012). https://doi.org/10.1007/978-3-642-24797-2_7
13. Ha, J., Haralick, R.M., Phillips, I.T.: Document page decomposition by the bounding-box project. In: Proceedings of 3rd International Conference on Document Analysis and Recognition, vol. 2, pp. 1119–1122. IEEE (1995)
14. Hooson, M.: Cryptocurrency glossary of terms & acronyms (2022). https://www.forbes.com/advisor/investing/cryptocurrency/crypto-glossary/
15. Horppu, I., Nikander, A., Buyukcan, E., Mäkiniemi, J., Sorkhei, A., Ayala-Gómez, F.: Automatic classification of games using support vector machine. arXiv preprint arXiv:2105.05674 (2021)
16. Joachims, T.: A probabilistic analysis of the rocchio algorithm with TFIDF for text categorization. Carnegie-mellon univ pittsburgh pa dept of computer science (1996)
17. JoMingyu: Google-Play-Scraper (2022). https://pypi.org/project/google-play-scraper/
18. Keras Team: Handwriting recognition (2022). https://github.com/keras-team/keras-io/blob/master/examples/vision/handwriting_recognition.py
19. Kingma, D.P., Ba, J.: Adam: a method for stochastic optimization. arXiv preprint arXiv:1412.6980 (2014)
20. Konstantinidis, S.: Computing the Levenshtein distance of a regular language. In: IEEE Information Theory Workshop, pp. 4-pp. IEEE (2005)
21. Kovalcik, T.: Digital forensics of cryptocurrency wallets. Master's thesis, Halmstad University, School of Information Technology (2022)
22. Krizhevsky, A., Sutskever, I., Hinton, G.E.: ImageNet classification with deep convolutional neural networks. In: Advances in Neural Information Processing Systems, vol. 25 (2012)

23. Laricchia, F.: Global mobile OS market share 2012–2022 (2022). https://www.statista.com/statistics/272698/global-market-share-held-by-mobile-operating-systems-since-2009/
24. Lee, A.W.C., Chung, J., Lee, M.: GNHK: a dataset for English handwriting in the wild. In: Lladós, J., Lopresti, D., Uchida, S. (eds.) ICDAR 2021. LNCS, vol. 12824, pp. 399–412. Springer, Cham (2021). https://doi.org/10.1007/978-3-030-86337-1_27
25. Marti, U.V., Bunke, H.: The IAM-database: an English sentence database for offline handwriting recognition. Int. J. Doc. Anal. Recogn. **5**(1), 39–46 (2002)
26. Montanez, A.: Investigation of cryptocurrency wallets on iOS and Android mobile devices for potential forensic artifacts. Marshall University Research Project (2014)
27. Nakamoto, S.: Bitcoin: a peer-to-peer electronic cash system (2008). https://bitcoin.org/bitcoin.pdf
28. Prakash, G., Koshy, J.: Google play store apps (2021). https://www.kaggle.com/datasets/gauthamp10/google-playstore-apps
29. Singh, S.S., Karayev, S.: Full page handwriting recognition via image to sequence extraction. In: Lladós, J., Lopresti, D., Uchida, S. (eds.) ICDAR 2021. LNCS, vol. 12823, pp. 55–69. Springer, Cham (2021). https://doi.org/10.1007/978-3-030-86334-0_4
30. The Monero Project: Monero - secure, private, untraceable (2022). https://www.getmonero.org/
31. Ukwen, D.O., Karabatak, M.: Review of NLP-based systems in digital forensics and cybersecurity. In: 2021 9th International Symposium on Digital Forensics and Security (ISDFS), pp. 1–9, Elazig, Turkey (2021). https://doi.org/10.1109/ISDFS52919.2021.9486354
32. Wigington, C., Tensmeyer, C., Davis, B., Barrett, W., Price, B., Cohen, S.: Start, follow, read: end-to-end full-page handwriting recognition. In: Proceedings of the European Conference on Computer Vision (ECCV), pp. 367–383 (2018)
33. Wood, G.: Ethereum: a secure decentralised generalised transaction ledger (2014). https://github.com/ethereum/yellowpaper
34. Xu, Q., Ibrahim, G., Zheng, R., Archer, N.: Toward automated categorization of mobile health and fitness applications. In: Proceedings of the 4th ACM MobiHoc Workshop on Pervasive Wireless Healthcare, pp. 49–54 (2014)
35. Ye, Q., Doermann, D.: Text detection and recognition in imagery: a survey. IEEE Trans. Pattern Anal. Mach. Intell. **37**(7), 1480–1500 (2014)
36. Young, P., Lai, A., Hodosh, M., Hockenmaier, J.: From image descriptions to visual denotations: new similarity metrics for semantic inference over event descriptions. Trans. Assoc. Comput. Linguist. **2**, 67–78 (2014)