

Automated Detection of Password Leakage from Public GitHub Repositories

Runhan Feng, Ziyang Yan,
Shiyan Peng, Yuanyuan Zhang

44th International Conference on Software Engineering (ICSE2022)

発表者：松尾 和輝

背景

- セキュリティ意識の欠如のために、認証情報をGitHub上に公開してしまうミスが増加^{[2] [3]}
 - 漏洩したパスワードが悪用される可能性
- 認証情報の漏洩を検知する研究が数多く行われている^{[10][50][67]}
 - APIキーのような構造化された認証情報のみを対象、正規表現を用いて検知
- 人が作成するパスワードの漏洩を検知する研究は行われていない
 - パスワードの多様性、言語の違いが障壁となる

[2] Github kills search after hundreds of private keys exposed. <https://it.slashdot.org/story/13/01/25/132203/github-kills-search-after-hundreds-of-private-keys-exposed>, 2013. [Online; Accessed on 02/04/2022].

[3] Aws urges devs to scrub secret keys from github. <https://developers.slashdot.org/story/14/03/24/0111203/aws-urges-devs-to-scrub-secret-keys-from-github>, 2014. [Online; Accessed on 02/04/2022].

[10] About secret scanning. <https://docs.github.com/en/github/administering-a-repository/about-secret-scanning>, 2022. [Online; Accessed on 02/04/2022].

[50] Meli, M., McNiece, M. R., and Reaves, B. How bad can it get? characterizing secret leakage in public github repositories. In NDSS (2019).

[67] Sinha, V. S., Saha, D., Dhoolia, P., Padhye, R., and Mani, S. Detecting and mitigating secret-key leaks in source code repositories. In 2015 IEEE/ACM 12th Working Conference on Mining Software Repositories (2015), IEEE, pp. 396–400.

目的とアプローチ

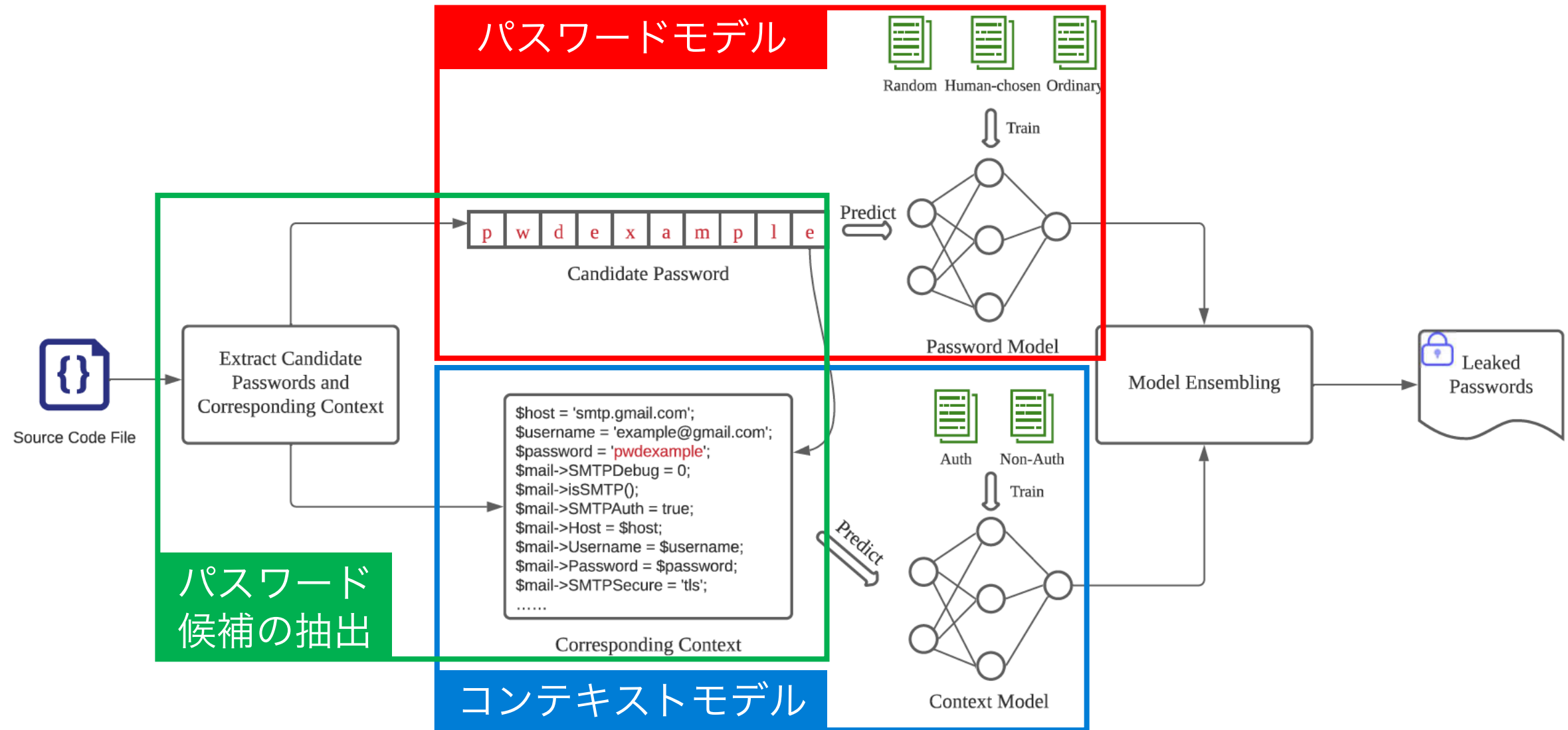
- 目的

- 異なるプログラミング言語で書かれたコードファイル中のパスワードを自動検出するためのPassFinderを提案

- アプローチ

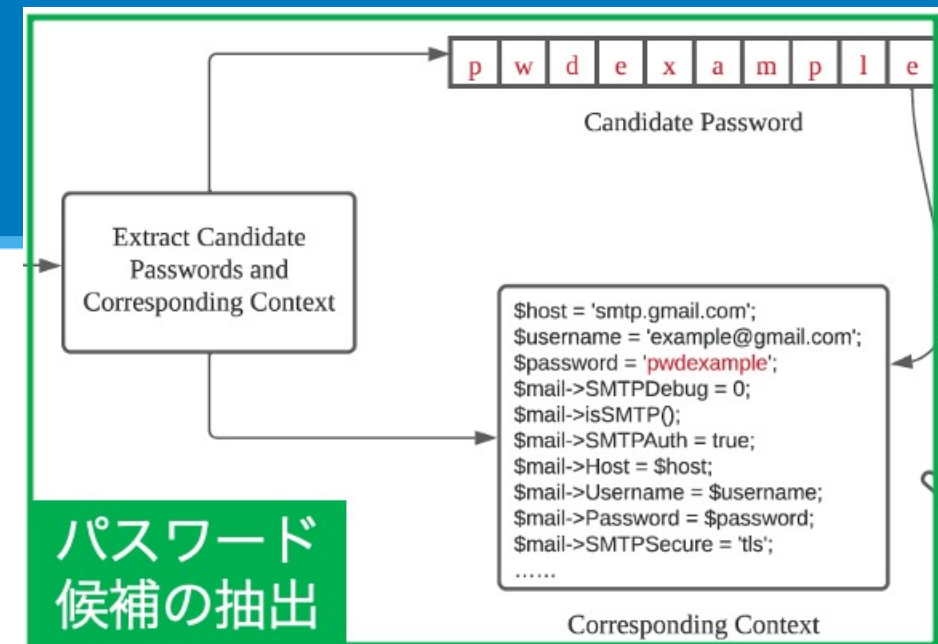
- 人間が作成したパスワードと、機械が生成したランダムパスワード、通常の文字列を分類するモデルを構築（パスワードモデル）
- 文字列の周辺情報を用いて、パスワードであることを識別するモデルを構築（コンテキストモデル）
- 両モデルの出力を組み合わせることで、パスワードであることを識別する

モデル概要図

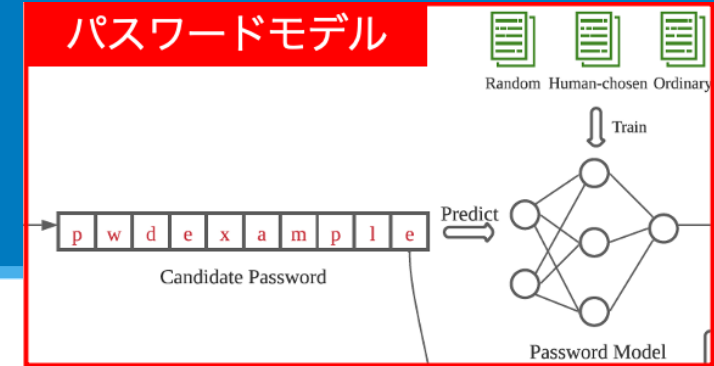


パスワード候補の抽出

- 入力をファイルとする
 - ファイル全体の前処理
 - 非ASCII文字の削除
 - 低頻出語彙の置換 を行う
- 引用符で囲まれた文字列をパスワードの候補とする
 - 6から30文字までの文字列が対象
- 特定した文字列をキーワードモデルに，パスワード候補の前後6行のコードスニペットをコンテキストモデルに与える



パスワードモデルの学習



人が生成したパスワード，機械が生成したパスワード，通常
の文字列の3分類問題

- データセット

- 人が生成したパスワード

- 過去にリークされたLinkedIn^[1]のデータから5,800万件を収集

- 機械が生成したパスワード

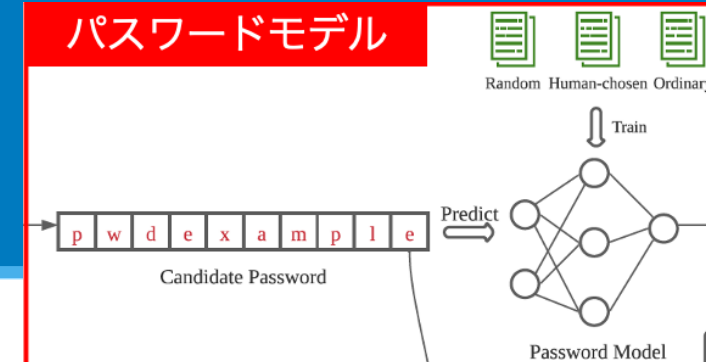
- 擬似乱数生成器により，6から30文字の長さで各長さごとに100万件，合計で2,500万件の文字列を生成

- 通常 of 文字列

- GitHubでよく使われる10種類のプログラミング言語^[5]について，それぞれスター数上位3000件のリポジトリを収集，合計3万件のリポジトリを取得
 - 引用符で囲まれた1600万件の長さ6から30のユニークな文字列を抽出

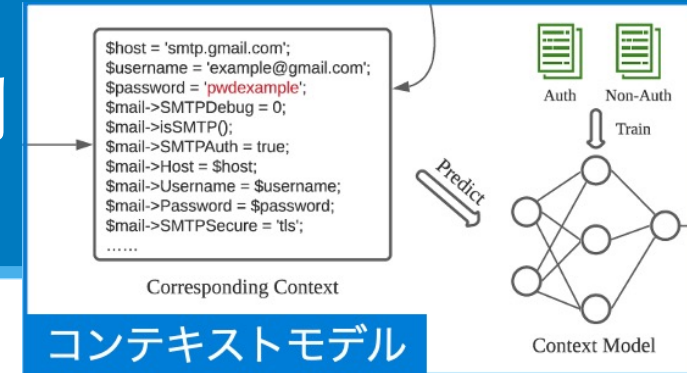
[1] LinkedIn leaked passwords. <https://hashes.org/public.php>, 2012. [Online; Accessed on 02/04/2022].

パスワードモデルの学習



- 各文字列は、文字単位でトークン化, one hot encodingで符号化
 - 訓練データ, テストデータ, 検証データに8 : 1 : 1の割合で分割
 - テストセットでハイパーパラメータの微調整を行い, 最後に検証セットで評価
- アルゴリズム
 - TextCNN

コンテキストモデルの学習

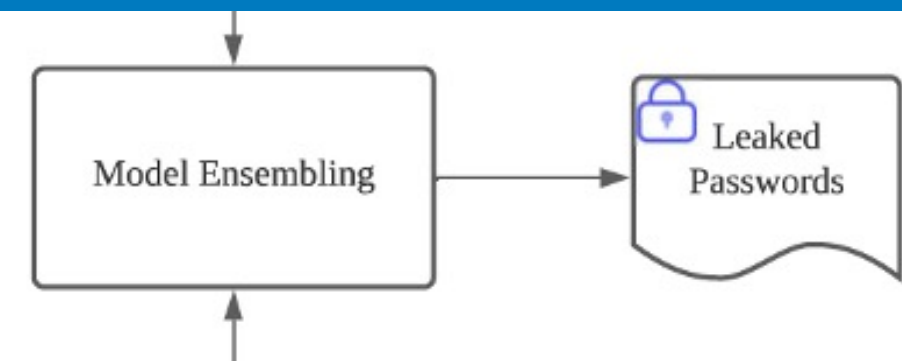


人が生成したパスワード，通常の文字列の2分類問題

- データセット

- 認証のために収集したコードスニペットは，各言語600件，計6000件
- 認証に関係のないコードスニペットは，各言語600件，計6000件
- ラベリングを実施
- トークン化，符号化，アルゴリズムはキーワードモデルと同様
- 10分割交差検証を実施

モデルのアンサンブル



- パスワードモデルとコンテキストモデルの結果を組み合わせる
 - モデルアンサンブル^[37]の技術を使用
- パスワードモデルとコンテキストモデルの両方ともパスワードであると予測した場合にのみ、パスワードであると識別

[37] Dietterich, T. G. Ensemble methods in machine learning. In International workshop on multiple classifier systems (2000), Springer, pp. 1–15.

リサーチクエスチョン

- RQ1：パスワードモデルとコンテキストモデルはそれぞれの程度有効か？
- RQ2：PassFinderはどの程度効果的にパスワードを検出することができるか？
- RQ3：コンテキストモデルは言語間検証においてどの程度有効か？

RQ1 | 結果 個別モデルの有効性

- パスワードモデルについて
 - 平均して96.35%のPrecision, 97.29%のRecall, 96.79%のF1スコアを達成

Table 1: Performance of the Password Model

	Precision	Recall	F1-score
Human-chosen	98.54%	96.86%	97.69%
Random	99.21%	98.92%	99.06%
Ordinary	91.29%	96.08%	93.62%
Marco-Average	96.35%	97.29%	96.79%

RQ1 | 結果 個別モデルの有効性

- コンテキストモデルについて
 - Naive Bayes(NB), K-Nearest-Neighbors(KNN), Support Vector Machine(SVM), Logistic Regression(LR)と比較
 - 提案手法が最も精度が良かった

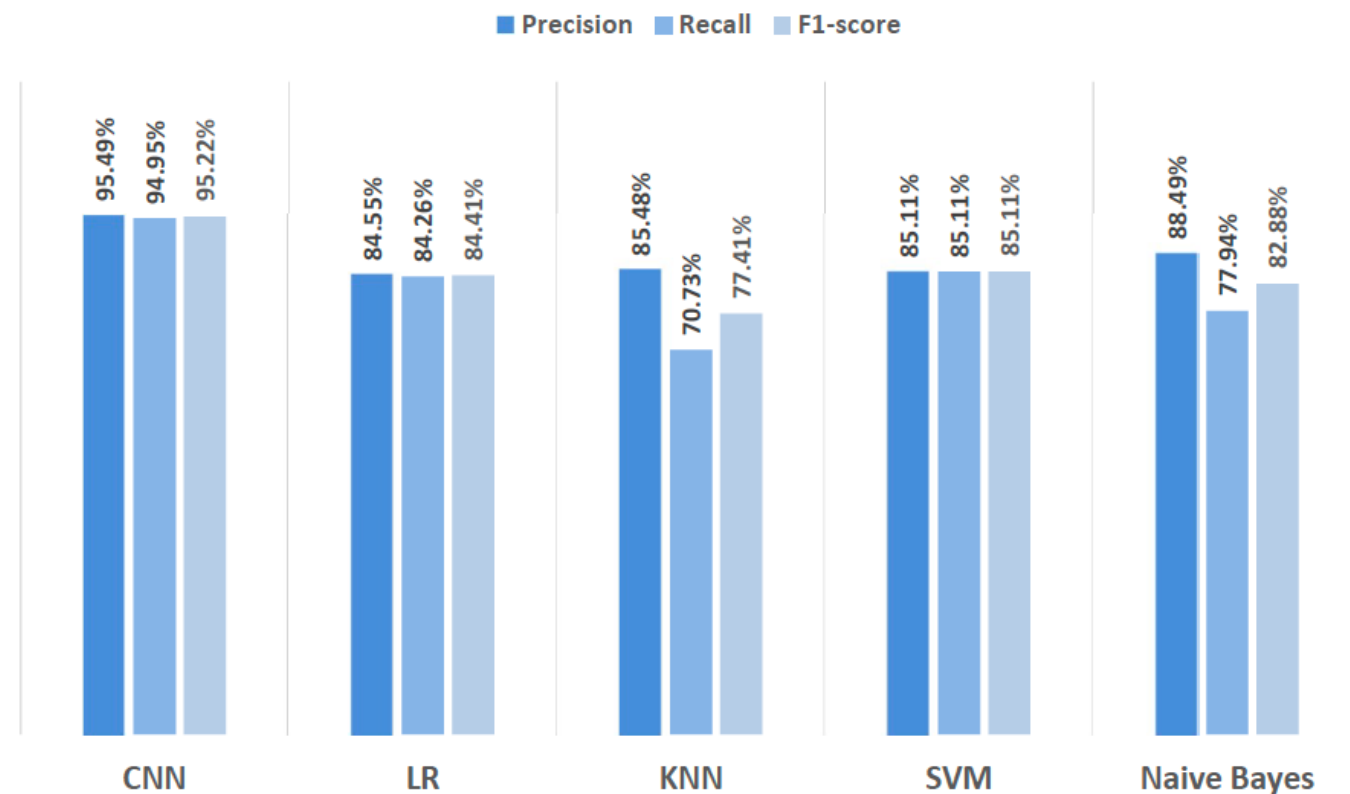


Figure 4: Performances of different classification methods

RQ2 | 結果 モデルの有効性

- APIで収集した5,000件のファイルから，395件のパスワードを特定，アノテーションを実施
- detect-secrets^[7]をベースラインとし，比較を行う
- 正規表現ベースの手法より提案手法の方が，精度が高い

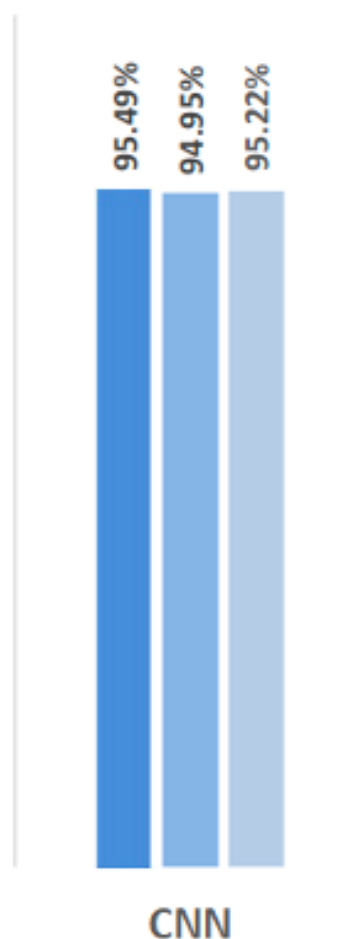
	PassFinder	Regex-based
Detected passwords	390	3,621
# True Positives	318	224
# False Positives	72	3,397
# False Negative	77	171
Precision	81.54%	6.19%
Recall	80.51%	56.71%
F1-score	81.02%	11.16%

[7] Solarwinds github password. https://www.theregister.com/2020/12/16/solarwinds_github_password/, 2020. [Online; Accessed on 02/04/2022].

RQ3 | 結果 言語間における有効性

- 9つの言語を学習用に，残りの1つをテスト用にし，コンテキストモデルをそれぞれ学習
- バランス良く学習させたモデルよりも2%ほど精度が低下
 - 言語横断的な環境でも高い精度を誇る

■ Precision ■ Recall ■ F1-score



Languages	Precision	Recall	F1-score
PHP	94.62%	94.62%	94.62%
JavaScript	95.59%	95.43%	95.51%
Ruby	95.29%	96.10%	95.69%
Python	94.41%	93.93%	94.17%
Java	94.76%	95.08%	94.92%
C#	95.72%	90.40%	92.98%
C++	91.81%	87.16%	89.42%
TypeScript	92.46%	89.32%	90.86%
Golang	87.94%	93.27%	90.53%
C	88.89%	89.64%	89.26%
Average	93.15%	92.50%	92.80%

まとめ

- 目的
 - GitHubリポジトリにおけるパスワード漏洩の検出手法の構築
- アプローチ
 - キーワードと、周辺の情報を活用した分類器の学習
- 結果
 - 既存手法と比較して、提案手法は高い精度でパスワードを検出可能であることが示された

所感

- 漏洩したパスワードをデータセットとしていることに驚いた
 - 倫理という単語が何回か出てきており、とても繊細な研究であると感じた
- パスワードをコミットしてしまうなんて不注意にも程があると思う