SCIS 2019 2019 Symposium on
Cryptography and Information Security
Shiga, Japan, Jan. 22 - 25, 2019
The Institute of Electronics,
Information and Communication Engineers

PE 表層情報を用いた機械学習による静的マルウェア検知
Using surface information from PE files for Static Machine-Learning-Based Malware Detection

茂木裕貴 *
Yuki Mogi

あらまし impfuzzy のような fuzzy hash は，類似したファイルが類似したハッシュ値になることから，マルウェアのクラスタリングや亜種の特定に使われている．また peHash もマルウェアのクラスタリングに用いられる．一方で、これらを直接マルウェアの検知に利用する研究はこれまでほとんどなかった．そこで本研究では，こうしたハッシュ値がマルウェアの検知に有効かどうかを FFRI Dataset 2018 を用いた機械学習による分類器の作成により検討し，以下の結果を得た．1. 先行研究で報告されている通り，fuzzy hash や peHash を含まない PE 表層情報はマルウェアの検知に有効である．2. fuzzy hash や peHash にのみ基づいた分類器は，それらを含まない PE 表層情報に基づいた分類器より性能が低かった．3. fuzzy hash や peHash を含まない PE 表層情報に基づいた分類器と fuzzy hash 及び peHash に基づいた分類器を組み合わせることで，単体での性能より高い性能を示した．4. fuzzy hash や peHash を含まない PE 表層情報に基づいた分類器をパラメーターチューニングを行った場合であっても，fuzzy hash 及び peHash に基づいた分類器と組み合わせることで，特に False Positive Rate が低くなるようなしきい値の設定で有効であった．

キーワード マルウェア, 静的検知, ファジーハッシュ，機械学習

1 はじめに

マルウェアの検知はサイバーセキュリティにおいて重要であり，これまで様々な研究が行われている．特にマルウェアが動作するのを待たず，静的に検知する方法は，近年発展の著しい機械学習が応用され成果が報告されている [1, 2, 3, 4, 5]．一方で fuzzy hash や peHash はマルウェアのクラスタリングや亜種の推定に使用されているが，これを直接機械学習し，マルウェアの検知を行うような研究はほとんどなかった．しかしクラスタリングや亜種の推定に使えるのであれば，機械学習により，そうした類似したマルウェアのグループと，類似していない非マルウェア（以下クリーンウェア）は分類できると考えられる．そこで本研究ではこれを検証し，fuzzy hash や peHash がマルウェアの検知に直接貢献するかどうか確認するため，FFRI Dataset 2018 を用いて実験を 4 つ行った．FFRI Dataset 2018 は MWS Dataset 2018 の一部として提供されている [6]．FFRI Dataset 2018 では，マルウェア 29 万件，非マルウェア（以下クリーンウェアと呼ぶ）21 万件に対しハッシュ値及び表層解析データを提供している．これを用いることで，本研究は fuzzy hash のマルウェアの静的検知への有効性を検討する．

本稿の構成は以下の通りである．まず第二章で関連研究について述べる．第三章から第六章にかけ，各章で実験の内容及びその結果について報告する．それをふまえ，第七章で結論及び Future Work について述べる．

2 関連研究

fuzzy hash とは context triggered piecewise hash (CTPH) とも呼ばれ [7]，類似した文章やファイルが類似した値になるようなハッシュ値である．このアイデアはスパムを検知するため，SpamSum [8] において述べられ，それをもとに [9] で CTPH として定式化された．また impfuzzy は JPCERT/CC が開発し，マルウェアの種類の判別の実験において，ssdeep や imphash より高い精度をみせた [10]．peHash はマルウェアのクラスタリング手法として [11] で提唱された．ハッシュ値の異なる実装が複数存在するため，FFRI Dataset 2018 では pehash [12] を用いて 6 種類の peHash の値を提供している．

こうしたハッシュ値を直接マルウェアの検知に利用す

* 株式会社 FFRI 〒 150-0013 東京都渋谷区恵比寿 1 丁目 18 番 18 号．FFRI, Inc., 1-18-18, Ebisu, Shibuya-ku, Tokyo 247-8501, Japan. yuki.mogi@ffri.jp

る研究はほとんどないが，愛甲 [13] は ssdeep をマルウェアの検知に利用した．ただし，その他の impfuzzy といった他の fuzzy hash や peHash を使った検知や，これらを機械学習した静的検知は行われていない．

機械学習を用いてマルウェアの静的検知を行う研究は数多く存在する．E. Raff ら [2] は PE ファイル全体を 1 次元 CNN にかける手法 MalConv を提案した．PE ファイル全体ではなく一部の特徴量に着目する研究もある [3, 4, 5]．H. S. Anderson ら [1] はこうした研究で重要であると報告された特徴量を提供する Ember データセットを作成した．さらに同論文ではデータセットの元となった PE ファイルを MalConv で分類した結果と，データセットで提供される特徴量に基づいた LightGBM [14] によって分類した結果の比較が行われた．PE ファイル全体を用いた MalConv が AUC 0.9982 だったのに対し，PE 表層情報 に基づいた LightGBM が AUC 0.9991 と高い精度を出し，適切に PE ファイルから抽出された PE 表層情報の有効性が示された．

3　実験 1 PE 表層情報によるマルウェア検知

3.1　実験 1 の内容

本実験では fuzzy hash や peHash を含まない PE 表層情報に基づいてマルウェアの検知が高い精度で行えるかどうか確認を行った．この PE 表層情報は FFRI Dataset 2018 で提供されている，pefile [15] から得られたダンプファイルから抽出した．マルウェア 29 万件，クリーンウェア 21 万件のあわせて 50 万件それぞれに対し feature hashing trick [16] により 450 次元の特徴ベクトルを作成した．これ用いて Random Forest [17] 及び [1] でも用いられている LightGBM でマルウェアとクリーンウェアの分類を行った．feature hashing trick 及び Random Forest は scikit-learn [18] を用いて実装を行い，LightGBM は Microsoft により提供されている LightGBM パッケージ [14] を用いた．ハイパーパラメータについてはチューニングをしていない．これについては実験 4 で検証を行う．分類は，50 万件のうち 9 割の 45 万件を訓練データ，残りの 5 万件をテストデータとして評価を行った．

3.2　実験 1 の結果

表 1: 実験 1 の結果（Accuracy 及び AUC）

分類器	Random Forest	LightGBM
Accuracy (%)	98.77	97.56
AUC	0.9986	0.9968

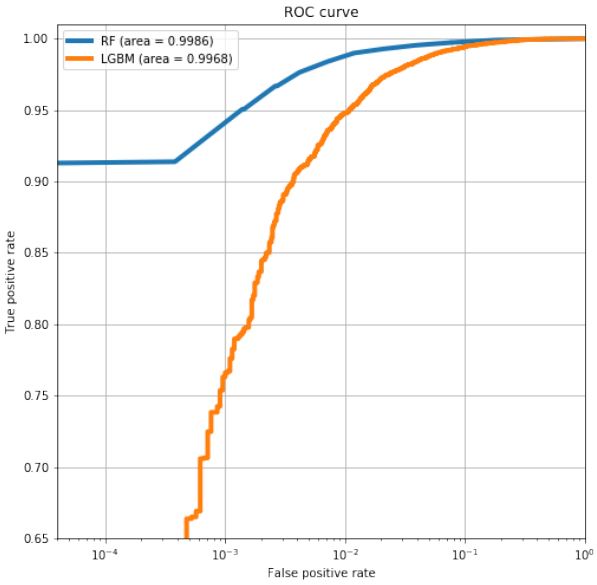

図 1: 実験 1 の結果（Low FPR における ROC 曲線）．ここで Random Forest の ROC 曲線は横に伸びているのではなく，原点と直線でつながっている．

実験の結果を表 1 にまとめる．また ROC 曲線を図 1 にまとめる．Random Forest が Accuracy，AUC ともに LightGBM より高い値となった．データセットが異なるため単純な比較はできないものの，Ember データセットを用いた MalConv より AUC は高く，高い精度で分類ができたと言える．先行研究での他のデータセットと同様に，FFRI Dataset 2018 においても PE 表層情報が検知へ有効であることが確認された．これは，PE 表層情報が，普遍的にマルウェアの検知に有効であることを示唆する．

一方で，図 1 をみると，Random Forest の ROC 曲線がある点を境に横に伸びているようにみえる．これは横に伸びているのではなく，原点と直線でつながっている．即ちある一定の値より FPR が下げられないことを示している．具体的には，False Positive Rate (FPR) 0.03812 より下げられず，その時の True Positive Rate (TPR) は 91.38 % であった．

4　実験 2 fuzzy hash によるマルウェア検知

4.1　実験 2 の内容

本実験では fuzzy hash と peHash に基づいてマルウェアの検知が高い精度で行えるかどうか確認を行った．ここでは FFRI Dataset 2018 で提供されているハッシュ値のうち，ssdeep，impfuzzy，Totalhash，EndGame を用いた．ここで Totalhash 及び EndGame は peHash の実装を指し，それぞれ [19]，[20] における実装が基となっている．これらを hash の文字を数字に対応させ変

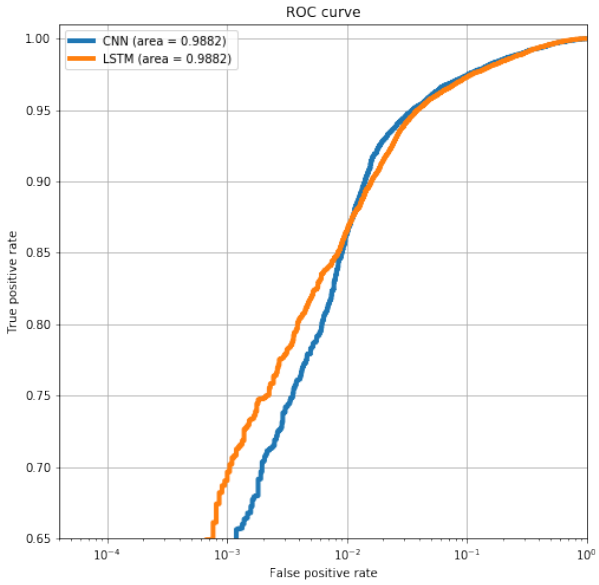

図 2: 実験 2 の結果（Low FPR における ROC 曲線）

換し，0 埋めにより長さを調節した上で 500 次元のベクトルを作成した．例えば abcde... というハッシュ値があった場合，それを変換したベクトルは [1, 2, 3, 4, 5, ...] となる．これを 自然言語処理で高い精度を出している Character Level CNN [21] 及び LSTM [22] によって実験 1 と同様にマルウェアとクリーンウェアの分類を行った．分類は，50 万件のうち 9 割の 45 万件を訓練データ，残りの 5 万件をテストデータとして評価を行った．

4.2 実験 2 の結果

表 2: 実験 2 の結果（Accuracy 及び AUC）

分類器	CNN	LSTM
Accuracy (%)	95.04	95.44
AUC	0.9882	0.9882

実験の結果を表 2 にまとめる．また ROC 曲線を図 2 にまとめる．Accuracy に関しては LSTM の方が良い結果が出たが，AUC はともに同じであった．ROC 曲線からは，Low FPR では LSTM の方が TPR が高く，一方 High FPR では CNN の方が TPR が高いことが読み取れる．一方で，いずれの場合も実験 1 で用いた分類器より Accuracy，AUC ともに劣っており，fuzzy hash 及び peHash はそれらを含まない PE 表層情報と比較すると検知に有効ではないことが分かる．

5 実験 3 fuzzy hash や peHash も含めた PE 表層情報によるマルウェア検知

5.1 実験 3 の内容

本実験では，実験 1 で作成した，fuzzy hash や peHash を含まない PE 表層情報に基づいた分類器及び実験 2 で作成した fuzzy hash と peHash に基づいた分類器を組み合わせることで，検知性能が向上するかを確認した．fuzzy hash と peHash に基づいた分類器が単体で検知にそれほど有効でなくとも，他の分類器と組み合わせることでそれぞれの単体での性能より検知性能が向上すれば，fuzzy hash や peHash はマルウェアの検知に貢献するといえる．組み合わせる手法としては実験 1 で作成した分類器をパラメーターチューニングした分類器の予測した確率と，実験 2 で作成した分類機の出力した確率を 9:1 の重みで足し合わせる方法をとった．分類は，実験 1 ，実験 2 と同様に，50 万件のうち 9 割の 45 万件を訓練データ，残りの 5 万件をテストデータとして評価を行った．

5.2 実験 3 の結果

表 3: 実験 3 の結果（Accuracy 及び AUC）

実験 1 の分類器	Random Forest		LightGBM	
実験 2 の分類器	CNN	LSTM	CNN	LSTM
Accuracy (%)	98.84	99.87	97.73	97.73
AUC	0.9989	0.9989	0.9973	0.9974

実験の結果を表 3 にまとめる．また ROC 曲線を図 3 及び図 4 にまとめる．比較のため，ROC 曲線の図には実験 1 の分類器の ROC 曲線もプロットしている．いずれの分類器も実験 1 の分類器より，Accuracy，AUC ともに向上した．特に実験 1 の Random Forest と実験 2 の LSTM を組み合わせた分類器は，しきい値を変えることで TPR を比較的高く保ったまま FPR を 0.01 % より小さくできる．このとき FPR は 0.009523 % で，TPR は 84.87 % であった．検知技術を実用するにあたっては，ユーザーの利便性のため高い TPR を保ちつつ FPR を下げる事が重要である．これを実現するのに fuzzy hash 及び peHash が貢献することを，この結果は意味する．

6 実験 4 パラメーターチューニングを行った場合のマルウェア検知

6.1 実験 4 の内容

本実験では，実験 1 で作成した，fuzzy hash や peHash を含まない PE 表層情報に基づいた分類器にパラメー

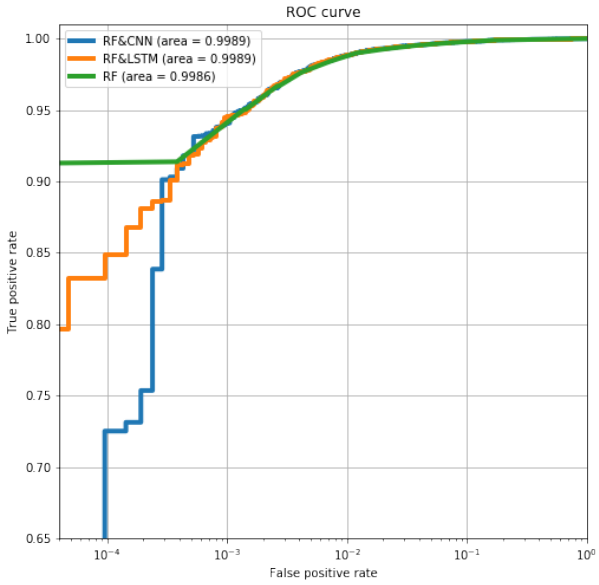

図 3: 実験 3 の結果（Low FPR における ROC 曲線・Random Forest）．ここで Random Forest の ROC 曲線は横に伸びているのではなく，原点と直線でつながっている．

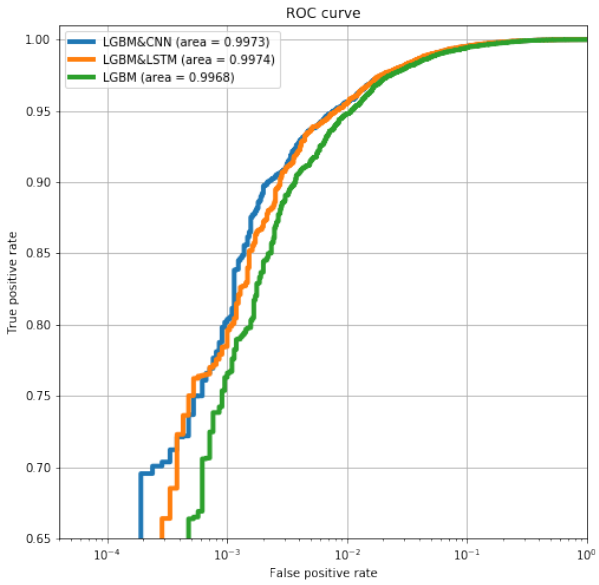

図 4: 実験 3 の結果（Low FPR における ROC 曲線・LightGBM）

ターチューニングを行った上で実験 2 で作成した fuzzy hash と peHash に基づいた分類器を組み合わせることでマルウェアの検知が高い精度で行えるかどうか確認を行った．実験 1 で作成した分類器のハイパーパラメーターはライブラリのデフォルトパラメーターを用いており，パラメーターチューニングを行うことで性能が向上すると考えられる．一方，それにより実験 2 で作成した分類器を組み合わせても性能が向上しない可能性がある．この場合，パラメーターチューニングによって fuzzy hash や peHash を用いる必要がなくなることになる．従ってパラメーターチューニングを行った場合であってもなお実験 2 で作成した分類器と組み合わせることが有効かどうかを確認する必要が生じる．そこで実験では，実験 1 で作成した分類器にパラメーターチューニングを施した．パラメーターチューニングの方法としては，グリッドサーチにより最適なハイパーパラメーターを求め，それを適用した．分類器を組み合わせる方法としては，実験 3 と同様に，実験 1 で作成した分類器をパラメーターチューニングした分類器の予測した確率と，実験 2 で作成した分類機の出力した確率を 9:1 の重みで足し合わせる方法をとった．分類は，実験 1 及び実験 2 及び実験 3 と同様に，50 万件のうち 9 割の 45 万件を訓練データ，残りの 5 万件をテストデータとして評価を行った．

6.2 実験 4 の結果

表 4: 実験 4 の結果（Random Forest）

チューニングした分類器	Random Forest		
実験 2 の分類器	なし	CNN	LSTM
Accuracy (%)	99.10	99.08	99.09
AUC	0.9992	0.9993	0.9993

表 5: 実験 4 の結果（LightGBM）

チューニングした分類器	LightGBM		
実験 2 の分類器	なし	CNN	LSTM
Accuracy (%)	98.84	98.85	98.86
AUC	0.9988	0.9989	0.9990

実験の結果を表 4 及び表 5 にまとめる．また ROC 曲線を図 5 及び図 6 にまとめる．比較のため，ROC 曲線の図には実験 1 の分類器の ROC 曲線もプロットしている．パラメーターチューニングを行った LightGBM については，実験 2 で作成した分類器を組み合わせることで Accuracy，AUC ともに向上した．一方で，パラ

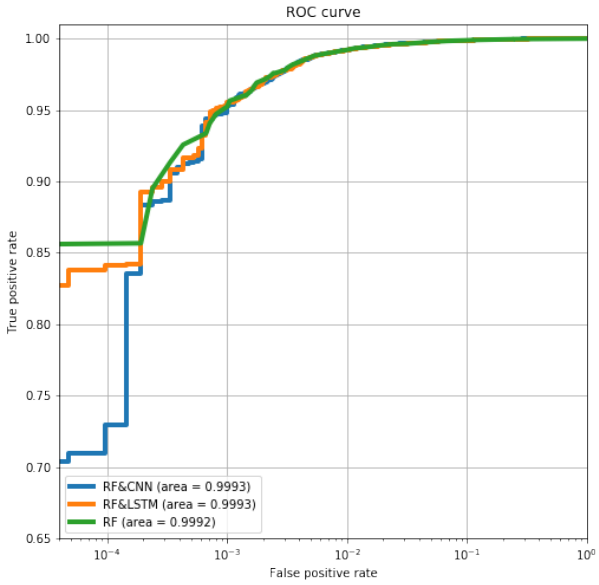

図 5: 実験 4 の結果（Low FPR における ROC 曲線・Random Forest）．ここで Random Forest の ROC 曲線は横に伸びているのではなく，原点と直線でつながっている．

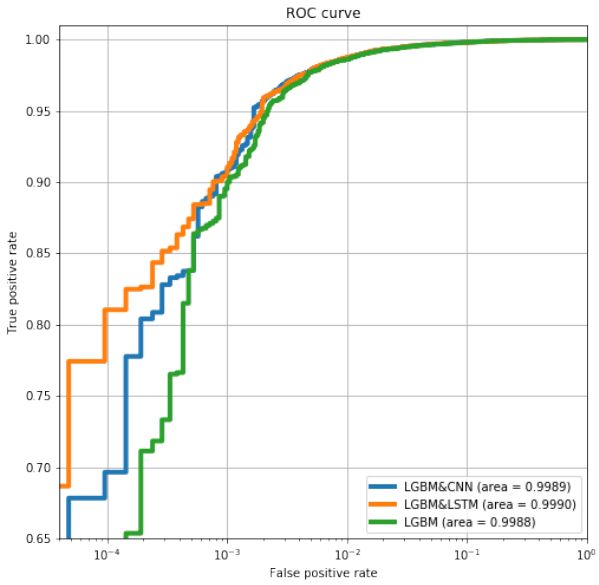

図 6: 実験 4 の結果（Low FPR における ROC 曲線・LightGBM）

メーターチューニングを行った Random Forest については，実験 2 で作成した分類器を組み合わせることで Accuracy が若干低下した．一方，AUC については改善した．特にしきい値を変えることで，TPR が 0 にならず FPR を 0.01 % より下げられる．実験 2 の LSTM を組み合わせた分類器は，FPR 0.9529 % のとき TPR 84.12 % であった．実験 3 と同様に，特に TPR を多少犠牲にしてでも 0.01 % より FPR を低く保ちたいとき，TPR を高く保つために fuzzy hash や peHash が有効であることが分かる．

7 結論

以上の実験より，以下のことが明らかになった．

1. PE 表層情報は，マルウェアの検知に普遍的に有効である可能性がある．
2. fuzzy hash 及び peHash は，それ単体ではそれらを含まない PE 表層情報ほど検知に有効ではない．
3. fuzzy hash 及び peHash を含まない PE 表層情報に基づいた分類器と fuzzy hash 及び peHash に基づいた分類器を組み合わせることで，それぞれ単体での性能より性能が向上する．
4. パラメーターチューニングを行った場合，Accuracy が若干下がることもあるが，Low FPR での性能が向上する．

今後の課題としては，2 つ挙げられる．まず，FFRI Dataset 2018 については，愛甲 [13] により Concept-Drift 及び Scale-free 性が確認されている．こうした性質を考慮すると，時系列に沿って学習させていき，検知率がどう変化していくかを調べることには意味がある．次に，本研究により，fuzzy hash 及び peHash がマルウェアの検知に有効であることが判明したが，これらのハッシュ値はマルウェアの検知を目的とはしておらず，単体ではさほど高い検知性能は出なかった．そこでそれ単体で高い精度で検知ができるハッシュ値（仮に ML Hash と呼ぶ）が考えられれば，データセットのポータビリティという観点で重要である．誰かがマルウェアを解析した際，SHA256 などの通常のハッシュ値に加え，ML Hash を併記すれば，それを用いてデータセットを拡張することができる．また演算に要する時間が短ければ，エンドポイントでのマルウェアの検知において効果的である．

参考文献

[1] H. S. Anderson, P. Roth, “Ember: an open dataset for training static PE malware machine learning models,” arXiv preprint arXiv:1804.04637, 2018.

[2] E. Raff, J. Barker, J. Sylvester, R. Brandon, B. Catanzaro and C. Nicholas, “Malware Detection by Eating a Whole EXE,” arXiv preprint arXiv:1710.09435, 2017.

[3] J. Saxe, K. Berlin, “Deep Neural Network Based Malware Detection Using Two Dimensional Binary Program Features,” Malicious and Unwanted Software (MALWARE), 2015 10th International Conference, pages 11—20, IEEE, 2015.

[4] M. Z. Shafiq, S. M. Tabish, F. Mirza, and M. Farooq, “A framework for efficient mining of structural information to detect zero-day malicious portable executables,” Technical Report, TR-nexGINRC-2009-21, January, 2009, available at https://www.researchgate.net/publication/242084613_A_Framework_for_Efficient_Mining_of_Structural_Information_to_Detect_Zero-Day_Malicious_Portable_Executables

[5] K. Raman, “Selecting Features to Classify Malware,” InfoSec Southwest 2012, 2012, available at http://2012.infosecsouthwest.com/files/speaker/materials/ISSW2012_Selecting_Features_to_Classify_Malware.pdf

[6] 高田雄太, 寺田真敏, 松木隆宏, 笠間貴弘, 荒木粧子, 畑田充弘, “マルウェア対策のための研究用データセット～MWS 2018 Datasets～,” 情報処理学会, Vol.2018-CSEC-82, No.38, 2018 年 7 月

[7] ssdeep, https://ssdeep-project.github.io/ssdeep/index.html, accessed: 2018-12.

[8] README, https://www.samba.org/ftp/unpacked/junkcode/spamsum/README, accessed: 2018-12.

[9] J. Kornblum, “Identifying almost identical files using context triggered piecewise hashing,” Digital Investigation Volume 3, Supplement, September 2006, Pages 91–97, Elsevier, 2006.

[10] Import API と Fuzzy Hashing でマルウエアを分類する ～impfuzzy～(2016-05-09), https://blogs.jpcert.or.jp/ja/2016/05/impfuzzy.html, accessed: 2018-12.

[11] G. Wicherski, “peHash: A Novel Approach to Fast Malware Clustering,” LEET ’09, 2009, available at https://www.usenix.org/legacy/event/leet09/tech/full_papers/wicherski/wicherski.pdf

[12] pehash, https://github.com/knowmalware/pehash, accessed: 2018-12.

[13] 愛甲健二, “Concept Drift と Scale-free の性質を持つデータセットに対する検知の自動化手法,” CSS 2018, 2018.

[14] LightGBM, https://github.com/Microsoft/LightGBM, accessed: 2018-12.

[15] pefile, https://github.com/erocarrera/pefile, accessed: 2018-12.

[16] K. Weinberger, A. Dasgupta, J. Langford, A. Smola and J. Attenberg, “Feature hashing for large scale multitask learning,” Proceeding ICML ’09 Proceedings of the 26th Annual International Conference on Machine Learning, pages 1113–1120, ACM , 2009.

[17] L. Breiman, “Random Forests,” Machine Learning, October 2001, Volume 45, Issue 1, pages 5—32, Kluwer Academic Publishers.

[18] scikit-learn, https://scikit-learn.org/stable/, accessed: 2018-12.

[19] viper, https://github.com/viper-framework/viper, accessed: 2018-12.

[20] pehashd, https://github.com/endgameinc/pehashd, accessed: 2018-12.

[21] X. Zhang, J. Zhao, Y LeCun, “Character-level Convolutional Networks for Text Classification,” arXiv preprint arXiv:1509.01626, 2016.

[22] S. Hochreiter, J. Schmidhuber, “LONG SHORT-TERM MEMORY,” NEURAL COMPUTATION 9(8), pages 1735–1780, 1997, available at https://www.researchgate.net/publication/13853244_Long_Short-term_Memory