Vol.2020-DPS-182 No.38
Vol.2020-CSEC-88 No.38
2020/3/13

脆弱性データベースを使用した脅威分析方法 －LDA分類器とコサイン類似度を用いたトピックモデル分析による攻撃事例と大規模脆弱性DBの突合について－

小柳 洋貴[1]　寶木 和夫[2]　三科 雄介[2]　梅澤 克之[1,2]

概要：製品開発の初期段階からセキュリティを考慮した設計を行うことは重要である．これを支援する脅威分析支援ツールは多々開発研究されている．しかし開発初期段階の自然言語で書かれる各種設計書などに内在する脅威を判定することは難しい．また扱えるデータ数に制限があるものも少なくない．そして過去の攻撃事例に酷似した攻撃も珍しくない．そこで本論文では，トピックモデルの手法の 1 つである Latent Dirichlet Allocation(LDA) を用いて大規模な脆弱性データベース (DB) と既存の攻撃事例を突合し，その攻撃で使われた脆弱性に類似する既存の脆弱性を抽出する手法を提案する．具体的には，LDA にて脆弱性情報と攻撃事例の文書に対してマッチングを行い類似する脆弱性情報を見つけ出す手法を提案する．

Threat Analysis Method using Vulnerability Database －Attack case and large-scale vulnerability DB matching by topic model analysis using LDA classifier and cosine similarity－

1. はじめに

昨今製品開発において，開発初期段階からセキュリティを考慮することが求められている．これを補助するために，開発の初期段階に支援を行えるツールの提供も増えている．米国 Microsoft 社が提供する“Microsoft Threat Modeling Tool”という脅威分析ツールもその 1 つである [1]．しかしこれは，開発の初期段階で自然言語により記述された文書の脅威分析には向かない傾向にある．さらにこれは，複数の脆弱性を用いて行われる攻撃に対して不向きである．このような課題に対して，国立研究開発法人産業技術総合研究所が開発したセキュリティ要件分析ツール (TACT)[2] を用いてトピックモデルによる文書間の類似度を算出することで自然言語で書かれた文書を突合分析し，類似度の高い脆弱性を発見する研究 [3][4] が行われている．そこでは，一つの自動車の設計情報や脆弱性特性に関して自然言語で記述した文書と，自動車に限らず広く脆弱性について自然言語で記述した文書との間でトピックモデルによる 2 文書間突合分析が行われた．その結果，その自動車にマッチする新たな脆弱性を導出し得るという有意な効果を得ている．ただし，後者の脆弱性のデータベース空間を増減する効果については，十分な分析が行われていなかった．

本研究では Python のトピックモデル用のライブラリを使用し，自然言語記述の類似度に基づく脆弱性分析の可能性を検証することを目的としたツールを制作する．制作されたツールを用いて，大規模脆弱性データベース（以下大規模脆弱性 DB とする）と論文 [4] で使われているテスラ社への自動車への攻撃の論文 [5] の BROWSER HACKING に関する節 (以下本提案中では比較用文書と呼ぶ) をマッチング処理し，類似度を算出する．その後，大規模脆弱性 DB から一部を抜粋したデータと全データでの該当した脆弱性情報から差を比較し，本提案で作成したツールの有効性を示す．2 章では本提案に関連する技術や使用するツールなど事前の説明を記述する．3 章では本提案の手法の紹介，4 章では提案手法に則って文書を分類した際の結果を記述する．5 章では結果からの考察を行い，6 章にてまとめを示す．7 章で本提案により生じた課題を記述する．

[1] 湘南工科大学, 神奈川県藤沢市辻堂西海岸 1-1-25
Shonan Institute of Technology, 1-1-25 Tsujido-Nishikaigan, Fujisawa, Kanagawa 251-8511, Japan
[2] 国立研究開発法人産業技術総合研究所, 東京都江東区青海 2-4-7
National Institute of Advanced Industrial Science and Technology (AIST), 2-4-7 Aomi, Koto-ku, Tokyo 135-0064, Japan

ⓒ 2020 Information Processing Society of Japan

Vol.2020-DPS-182 No.38
Vol.2020-CSEC-88 No.38
2020/3/13

2. 関連研究

2.1 大規模脆弱性 DB

米国 MITRE 社は CVE(Common Vulnerability and Exposure：共通脆弱性識別子)[6] を公開している．従来から脆弱性情報に関するデータベースはあったが，ベンダー各社や製品ごとに情報が提供されていた．そのため同一の脆弱性に対する情報であっても判別することが難しいことが多々あった．そこで米国からの支援を受け，MITRE 社により 1999 年に脆弱性情報を一意に特定できるよう CVE が提案された．現在は CERT/CC や IBM など多くの主要な脆弱性情報サイトと連携し，多くの脆弱性情報を統合して提供している．公式サイトからは csv，html，text，xml の 4 種のフォーマットで，Unix compressed，Gzipped，Raw のデータ圧縮フォーマットから入手することができる．本提案中では csv 形式のデータを用いる．データ構成は Name，Status,Description，Reference，Phase，Votes，Comments からなる．このうち Name と Description を抽出し用いる．Name には一意な ID が含まれ，Description には脆弱性に関する自然言語 (英語) で記述された説明文が入っている．

2.2 LDA(Latent Dirichlet Allocation)

文書には潜在的なトピックがあると仮定し，文書内の単語はそのトピックから生起されその単語集合が文書であるという考え方がある．この文書の集合から潜在的なトピックを推定しようというものがトピックモデルと呼ばれる．トピックモデルの中に潜在的意味解析と呼ばれるものがある．この中の 1 つに LDA がある．潜在的意味解析で LDA の手法が提案されるまでには以下の流れがある．まず基礎として LSA(Latent Semantic Analysis) というものがある．これは文書を単語の出現回数でベクトル化したものを TF-IDF により重み付けする（TF-IDF の詳細は 2.3 に記す）．重みづけしてあるベクトル群は膨大なため，特異値分解により近い意味を持つ者同士を近似することで，ベクトル空間を削減する．つまり単語から潜在的トピックに変換していることに相当する．このできたベクトル群を用いて，文書のクラスタリングや文書分類をしていくものが LSA である．しかし LSA はトピックに意味づけが数学的にはされるが，文の意味や単語の意味に関連した意味づけはされないという課題がある．LSA に確率の概念を追加したものに pLSA(Probabilistic LSA) というものがある．これは文書はある確率モデルに基づいて生成されるという考え方であり，pLSA は各文書における各トピックの重要度の割合と各トピックにおける各単語の重要度の割合を仮定する．この仮定されたモデルのパラメータを一般的に EM アルゴリズムと呼ばれる手法を用いて更新していき，最も尤度の高い確率モデルを生成する．pLSA の利点として値がすべて確率なので 0 から 1 で出力され，各単語が複数のトピックに属することができる．pLSA の理解を簡単にするためグラフィカルモデルを図 1 に示す．使われる記号の説明を表 1 に記す．pLSA では M 個の文書の中から各文書 d について考え，文書中の潜在的なトピック分布 c を推定する．推定される c からトピックに含まれる単語 w を割り当てる．この単語へ割り当てを N 単語分，M 個の文書分繰り返すことでモデルを作成している． LDA は David M. Blei 氏の論文 [7] によって提案された．LDA ではパラメータの値を 1 つに決定するのではなく，事前分布を用意し，パラメータの事後分布を推定する．このときに事前分布には主にディリクレ分布を，事後分布には多項分布を一般的に仮定する．LDA のグラフィカルモデルも pLSA と同様にイメージを容易にするため，図 2 に示す．LDA のグラフィカルモデルに使われる記号説明は表 2 に示す．LDA のモデルはトピック分布 θ がハイパーパラメータである事前分布 α から生成され，生成された θ から文書内の各単語のトピック分布 z が生成される．もう 1 つのハイパーパラメータである β からはあるトピックの語彙の出現確率が生成されてる．ここまでの z と φ により w が生起される．pLSA と LDA の違いは，LDA では各文書とトピックごとの単語分布に対してそれぞれディリクレ分布をハイパーパラメータとし与えていることにある．結果的に LDA は pLSA に比べ汎化性能が高くなることが多く，様々なものに活用することが可能である． LDA ではパラメータを解析的に求められない際のために，サンプリング手法の 1 つであるギブスサンプリングが採用されている．

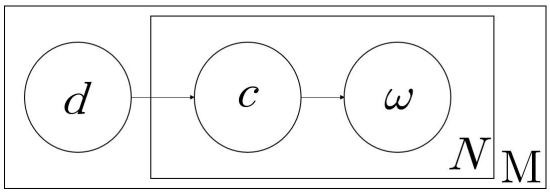

図 1 pLSA のグラフィカルモデル

表 1 pLSA のグラフィカルモデルにおける記号表

記号	説明
M	文書数
N	各文書の単語数
d	各文書
c	トピック分布
w	トピックごとの単語分布

2.3 TF-IDF

TF-IDF は TF と IDF という二つの概念を組み合わせたものである．TF は Term Frequency の略であり，単語の出現頻度を表すものである．以下この節で登場する記号は表 3 に示す．TF は式 (1) によって導出できる．

$$tf_{i,j} = \frac{n_{i,j}}{\sum_k n_{k,j}} \tag{1}$$

ⓒ 2020 Information Processing Society of Japan

Vol.2020-DPS-182 No.38
Vol.2020-CSEC-88 No.38
2020/3/13

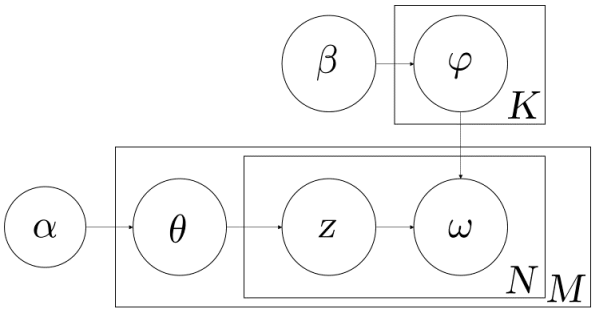

図 2　LDA のグラフィカルモデル

表 2　LDA のグラフィカルモデルにおける記号表

記号	説明
M	文書数
N	各文書の単語数
α	各文書ごとのトピック分布の事前分布の ハイパーパラメータ
β	トピックごとの単語分布の事前分布の ハイパーパラメータ
θ	各文書ごとのトピック分布
z	単語ごとのトピック分布
w	トピックごとの単語分布
φ	トピックごとの単語分布
K	トピック数

表 3　式 (1),(2) の記号

記号	説明
i, j, k	添え字
$n_{i,j}$	文書 d_i における t_i の出現回数
$n_{k,j}$	文書 d_i における単語の出現回数
$\|D\|$	総文書数
t_i	i 番目の単語
$\|\{d : d \ni t_i\}\|$	単語 t_i を含む文書数

IDF は Inverse Document Frequency の略であり，こちらはある単語の含まれる文書の割合の逆数を表す．式 (2) のように導出する．

$$idf_i = \log \frac{|D|}{|\{d : d \ni t_i\}|} \tag{2}$$

導出された TF と IDF を乗算することで TF-IDF を得ることができる．以下に簡単な例を示す．次の 2 つの文書があることを仮定する．

- 文書 1:ジャガイモと人参と玉ねぎとジャガイモ
- 文書 2:ほうれん草とジャガイモとさつま芋とほうれん草と白菜

このような 2 つの文書に対し TF の値は表 4，5 のようになる． 次に IDF の値を計算すると表 6 となる．導出した TF と IDF を乗算することで表 7 を得ることができる．

表 4　文書 1 における TF の値

tf(ジャガイモ, 文書 1)=$\frac{2}{4}$=0.5
tf(人参, 文書 1)=$\frac{1}{4}$=0.25
tf(玉ねぎ, 文書 1)=$\frac{1}{4}$=0.25

表 5　文書 2 における TF の値

tf(ほうれん草, 文書 2)=$\frac{2}{5}$=0.4
tf(ジャガイモ, 文書 2)=$\frac{1}{5}$=0.2
tf(さつま芋, 文書 2)=$\frac{1}{5}$=0.2
tf(白菜, 文書 2)=$\frac{1}{5}$=0.2

表 6　文書 1 と文書 2 における IDF の値

idf(ジャガイモ)=$\log_2(\frac{2}{2})$=0.0
idf(人参)=$\log_2(\frac{2}{1})$=1.0
idf(玉ねぎ)=$\log_2(\frac{2}{1})$=1.0
idf(ほうれん草)=$\log_2(\frac{2}{1})$=1.0
idf(白菜)=$\log_2(\frac{2}{1})$=1.0
idf(さつま芋)=$\log_2(\frac{2}{1})$=1.0

表 7　文書 1 と文書 2 における TF-IDF の値

tfidf(ジャガイモ, 文書 1)=tf(ジャガイモ, 文書 1)・idf(ジャガイモ)
=0.5 · 0.0=0.0
tfidf(人参, 文書 1)=tf(人参, 文書 1)・idf(人参)
=0.25 · 1.0=0.25
tfidf(玉ねぎ, 文書 1)=tf(玉ねぎ, 文書 1)・idf(玉ねぎ)
=0.25 · 1.0=0.25
tfidf(ほうれん草, 文書 2)=tf(ほうれん草, 文書 2)・idf(ほうれん草)
=0.4 · 1.0=0.4
tfidf(ジャガイモ, 文書 2)=tf(ジャガイモ, 文書 2)・idf(ジャガイモ)
=0.2 · 0.0=0.0
tfidf(さつま芋, 文書 2)=tf(さつま芋, 文書 2)・idf(さつま芋)
=0.2 · 1.0=0.2
tfidf(白菜, 文書 2)=tf(白菜, 文書 2)・idf(白菜)
=0.2 · 1.0=0.2

2.4　評価指標

トピックモデルの評価指標は数多く提案されている．その中に Perplexity と Coherence がある．Perplexity とは分岐数または選択肢の数を表し，確率の逆数で表される．つまりどれほどの選択肢の中から単語が選択されているかを表しており，予測性能と考えることができる．そのため一般的に値は低いほど良いとされる．Coherence はトピックの品質を表す指標として使われ，人間にとって解釈がしやすいかどうかを表す．しかし Coherence に対する定義は明確にされていないのであるのが現状である．Coherence について最初に人間による評価方法 [8] が提案され，その後自動評価方法 [9] が発表されている．一般的に Coherence は高いほど良いとされる．

3.　LDA モデルの作成と文書間類似度の算出方法

3.1　提案の概要

本提案では表 8 のような環境で提案の検証を行った．使用したデータは表 9 の通りである．本提案で用いる比較用文書で主として使われた脆弱性は CVE-2011-3928 である．

ⓒ 2020 Information Processing Society of Japan

表 8　実験環境データ

名称	環境内容
OS	Oracle VM VirtualBox ＋ Ubuntu18.04LTS
CPU	Intel core i5 7500 ＋（VirtualBoX により 3core に制限）
メインメモリ	8GB （VirtualBox により 16GB より制限）
補助記憶装置	VDI 100GB
GPU	Intel HD Graohics 620
グラフィックメモリ	512MB
Python バージョン	Pyrhon 3.6.9

表 9　使用データ

データ名	件数	説明
小規模データ	200 件	CVE2011-3928 を中心とする前後 100 件のデータ
中規模データ	4398 件	2011 年度分の全 CVE データ
大規模データ	119479 件	1999 年から 2019 年 7 月 25 日までのデータ

小規模データとは，従来研究 [3] で使用されたデータであり，CVE-2011-3928 を中心とした前後 200 件のデータを指す．中規模データは，CVE-2011-3928 も含まれる 2011 年度分の CVE データを指す．大規模データは全ての CVE データを指す．これらのデータを用いて LDA モデルの作成し，CVE の文書群にトピック分布を与え作成したモデルを使い比較用文書にもトピック分布を付与する．その後文書間の類似度を算出し，文書を分類する．

3.2　Gensim による LDA モデルの作成

本提案では Python のライブラリである Gensim を用いて LDA を実装する．Gensim は LGPL のライセンスで提供され，教師なしトピックモデリングと自然言語のためのオープンソースライブラリである．本研究では既存研究 [10] で公開されているソースコードを利用した．データ読み込み部や，ストップワードと呼ばれる文書の助詞や副詞などを除く処理等を適宜追加したものを利用する．LDA 生成プロセスの詳細は以下のとおりである．

(1) CVE の Description の行を配列データとして読み込む
(2) 全ての単語に対して Python のライブラリである nltk.corpus の wordnet クラスの morphy メソッドを用いて単語を原形に戻す
(3) 読み込んだ文書群からストップワードを nltk.corpus の stopwords クラスの words メソッドで english を指定し，文書から自動で取り除く
(4) 残った文書データから任意の出現回数以下（本研究では 2 回以下）の単語を除去する
(5) (3) までの処理を終えた文書群から corpora クラスの Dictionary メソッドにより単語辞書を作成する
(6) 作成した辞書と文書群から各文書を dictionary クラスの doc2bow メソッドにより単語のベクトルに変換する（この処理をコーパスの作成と呼ぶ）
(7) Gensim の ldamodel クラスの LdaModel メソッドに作成した辞書とコーパス，パラメータを指定して LDA モデルを作成する

パラメータの 1 つに 1 文書当たりの持ち得るトピック数の項目がある．このトピック数の決定は 3.3 に記述する．上記の手順で作成するコーパスは，文書を単語の出現回数で表した BoW(Bag of Words) と呼ばれる形式のベクトル表現である．本提案では BoW の他に 2 章で述べた TF-IDF でのコーパスも用いる．TF-IDF によるコーパスの作成手順は以下の通りである．

(I) 前述の (1) から (4) までの処理を行う
(II) Gensim の models クラスにある TfidfModel メソッドにより TfidfModel を作成する
(III) (6) で作成したコーパス（BoW コーパス）を Tfidf-Model に渡すことで TF-IDF コーパスを得る

以上により生成される 2 つのコーパスを用いて実験を行う．

3.3　トピック数の決定

前節までの手順により LDA モデルは作成可能であるが文書に存在するトピック数を決定する必要がある．今回は実験的にトピック数を求めた．トピック数 2 から 19 までのトピック数を評価したグラフを図 3,4 に示す．図 3,4 は全ての CVE データを用いて，出現頻度 2 以下の単語を取り除く設定にしたときの，LDA モデルの評価を行ったグラフである． 図 3 を見ると BoW コーパスでは，トピック数 3 の時点で Perplexity が低く Coherence が高い状態で差が最大である．よって数値的に見た場合，グラフ内のトピック数毎の LDA モデルとしてはトピック数 3 が良好であると言える．同様の視点で図 4 の TF-IDF コーパスのグラフを見ると，こちらはトピック数 6 が良好だと判断することができる．本提案ではこれらの評価結果より，良好だと思われるトピック数を LDA モデルの作成の際（生成プロセ

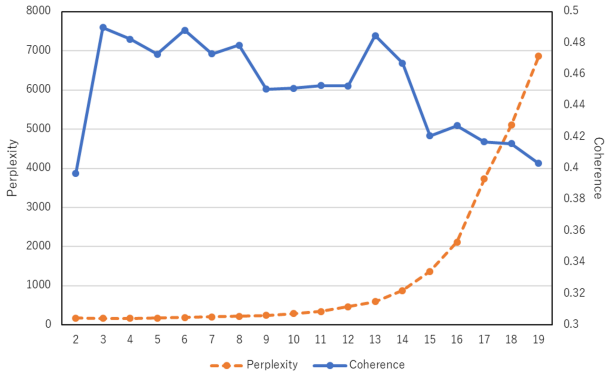

図 3　Perplexity と Coherence のグラフ (BoW コーパス)

ⓒ 2020 Information Processing Society of Japan

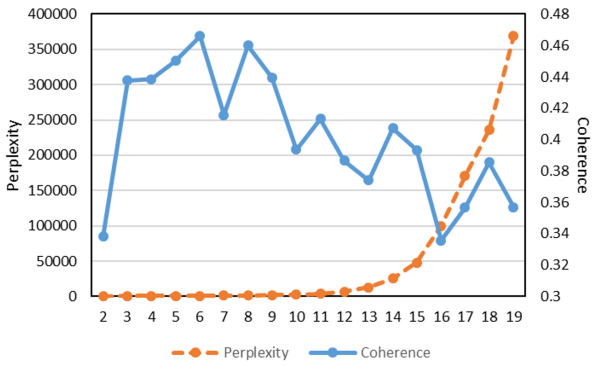

図 4　Perplexity と Coherence のグラフ (TF-IDF コーパス)

スの (7) のパラメータに該当）に用いる．

3.4　攻撃事例へのトピック分布の付与

比較用文書にもトピック分布を割り当てる必要がある，LDA モデルは 3.2 に示した CVE によるモデルを利用できる．LDA モデルを作成後の流れは以下のとおりである．

(i) 比較用文書を読み込む
(ii) (1)〜(4) までの処理を行う
(iii) Gensim の dictionary クラスの doc2bow メソッドを使い比較用文書を単語のベクトルにする
(iv) LdaModel クラスの get_document_topics メソッドにベクトル化した文書を引数として与え確率分布を割り当てる

3.5　文書間の類似度の算出

LDA モデルの作成と比較用文書へのトピック分布の割り当て終了後，CVE の脆弱性情報と比較用文書の文書間でコサイン類似度を求める．コサイン類似度は各文書のベクトルの内積によって求められる．

3.6　脆弱性情報の抽出

CVE-2011-3928 は今回用いている比較用文書における主として用いられた脆弱性であり，本提案では必ず検出されなければならない脆弱性である．よって CVE-2011-3928 と比較用文書とのコサイン類似度を閾値として設定する．設定した閾値以上である脆弱性数をカウントする．

4.　評価

4.1　評価結果

評価結果を表 10 に示す．従来ツールとは，従来研究 [2] で使用されたツールである．従来ツールを用いた場合は総件数 200 件に対して，比較用文書と類似していると判定された脆弱性は 200 件中 29 件（全体割合が 14.9 %）であった．提案ツールで TF-IDF コーパスを用いて，小規模データを用いた場合が 200 件中 125 件（全体割合が 62.5%）と小規模データを用いた場合，従来ツールより提案ツールの方が結果が悪化してしまっている．これは使用しているストップワードのリストや使用している品詞，トピック数やその他のパラメータの違いによるものと思われる．大規模データを用いた場合は 119479 件中 1514 件（全体割合 1.3%）であった．このことから部分的なデータでなくより多くのデータを用いた方が良いことが伺える．

5.　考察

5.1　データ件数による精度の差異

データを増やすことにより，精度の高いマッチングを行えた．しかしながら 1 万件程度ではそれほど劇的に精度が向上するわけではなく，10 万件規模でのデータの増加が必要であると考えられる．また本実験では CVE の脆弱性データベースを用いたため，脆弱性に特化した情報が得られた．そのため潜在的な脆弱性情報を比較用文章とマッチングしやすかったと考えられる．

5.2　コーパスによる精度の差異

実験結果から本論文の方式では TF-IDF を用いたコーパスの方が精度が高いことを示せた．これは CVE の Description が文書というには短いことから，単語自体が少ないことが要因と考えられる．単純な単語の出現頻度で行列を表す BoW での表現では情報量が不足する．しかし TF-IDF ならば単純な単語の出現頻度以外の要素も加味するため元の文書の長さが同じでも，コーパスが持つ情報量が多くなる．そのため精度に差が出たと考えることができる．逆にこの結果を踏まえると，CVE 単体だけでは 1 文書ごとのデータが不足しているということが言える．よってこれ以上の精度の向上やモデルの品質を上げるためには，Description の行に関連する研究やその脆弱性に対して詳しく述べている文書を加える必要があるとも言える．この問題をどのように解決していくかは今後の課題となる．

6.　まとめと今後の課題

本研究では Gensim による LDA モデルを製作することで，大量の文書を扱うことを可能にした．実験結果より，CVE の一部のデータと全てのデータを用いた場合では全てのデータを用いた方がより精度が高くなり検出精度に差が出ることを示せた．今回は LDA モデルを作成する際にトピック数を手動で決定しモデル作成を行ったが，今後は，人の手を介さずに決定することで人的リソースの削減とヒューマンエラーによる脆弱性の発生を抑えることが必要である．これについては Hierarchical Dirichlet Process[11] を使うことで解決できると考えられる．他にも本提案中では比較用文書における正解の脆弱性が決まっているため閾値を決定できたが，実際に不明な時点で決定するために，指標となるものを考える必要がある．また現在の CVE の Description だけでは文書量としては短いため，補助的に

ⓒ 2020 Information Processing Society of Japan

表 10 各データの該当件数

ツール名	使用データ	トピック数	総件数（件）	該当件数（件）	該当割合（ %）
従来ツール	小規模データ	不明	200	29	14.5
提案ツール（BoW コーパス）	小規模データ	5	200	30	15.0
	中規模データ	6	4398	4118	93.6
	大規模データ	6	119479	20890	17.5
提案ツール（TF-IDF コーパス）	小規模データ	2	200	125	62.5
	中規模データ	6	4398	2628	59.8
	大規模データ	7	119479	1514	1.3

別の文書を追加して用いることができるようにすることが必要である．さらに Gensim の使用によりメモリのリソースを抑えて実験を行ったが，全ての CVE データでトピック数 10 前後の LDA モデルの作成に 3 分程度かかった．今後データが増えた際トピック数の決定に時間を要することが考えられるため高速化する手段を考える必要が出てくる可能性がある．今回の提案では比較用文書として [5] を用いたが，他の事例でも同様の検証を行い評価を行う必要がある．

参考文献

[1] Microsoft: The STRIDE Threat Model，入手先 ⟨https://msdn.microsoft.com/ja-jp/ee823878⟩ (2020.01.15).
[2] 半田剣一, 大崎人士, 竹内泉, (2017) ”セキュリティ要件分析支援ツール TACT.: 情報処理学会ソフトウェア工学研究会ウィンターワークショップ 2017・イン・飛騨高山予稿集 (2017)
[3] 梅澤克之，三科雄介，田口研治，寶木和夫.: 脆弱性データベースを使用した脅威分析方法の提案, 暗号と情報セキュリティシンポジウム (SCIS2018) 予稿集, 1C2-6, Jan. 2018.
[4] 梅澤克之, 三科雄介, Sven Wohlgemuth, 寶木和夫.: 脆弱性データベースを使用した脅威分析方法, 暗号と情報セキュリティシンポジウム (SCIS2019)
[5] S. Nie et al.: FREE-FALL: HACKING TESLA FROM WIRELESS TO CAN BUS, Briefing,Black Hat USA 2017, July 2017.
[6] MITRE Corporation: CVE - Common Vulnerability and Exposure, 入手先 ⟨https://cve.mitre.org/⟩ (2020.01.15).
[7] D. Blei, A. Ng, and M. Jordan.: Latent Dirichlet Allocation, in Journal of Machine Learning Research(2003), pp. 1107-1135.
[8] Jonathan Chang, Sean Gerrish, Chong Wang, Jordan L Boyd-Graber,and David M Blei.: Reading tea leaves: How humans interpret topicmodels, In Advances in NIPS, pp. 288-296, 2009.
[9] Newman, D., Lau, J. H., Grieser, K. and Baldwin, T.: Automatic Evaluation of Topic Coherence, pp. 100-108 (2010).
[10] Latent Dirichlet Allocation (LDA) ゆるふわ入門 - あらびき日記, 入手先 ⟨https://abicky.net/2013/03/12/230747⟩ (2020-01-15).
[11] Teh, Y. W.; Jordan, M. I.; Beal, M. J.; Blei, D. M. (2006).: Hierarchical Dirichlet Processes, Journal of the American Statistical Association. 101 (476), pp. 1566-1581.

ⓒ 2020 Information Processing Society of Japan