機械学習を用いたテキスト分類による ライセンス特定のためのルール作成プロセス支援

東　裕之輔
和歌山大学 システム工学研究科
s151039@sys.wakayama-u.ac.jp

眞鍋　雄貴
熊本大学 大学院自然科学研究科
y-manabe@cs.kumamoto-u.ac.jp

大平　雅雄
和歌山大学 システム工学部
masao@sys.wakayama-u.ac.jp

要旨

近年では，*OSS* を利用したソフトウェア開発が盛んに行われており，*1* つのソフトウェアに多数のライセンスが含まれている場合がある．ライセンス違反を回避するために，ソースファイル単位でのライセンス特定を行うことが必要である．最も精度の高いライセンス特定ツールである *Ninka[5]* は，未知ライセンスの存在を検出することができる．しかし，未知のライセンスが何であるかは利用者が目視により確認する必要がある．また，大量の未知ライセンスが出力される場合もある．このような未知ライセンスを特定するには，ライセンスルールを手作業で作成する必要があり，複雑で労力がかかる．本研究では，ライセンスルール作成支援を目的とし，*Debian v7.8.0* のソフトウェアパッケージを入力として *Ninka* が未知ライセンスと判定したファイル群に対して *K-means* 法を適用し，ライセンス記述の分類精度を定性的および定量的評価を行った．また，*Random Forest* 法を用いて，類似ライセンスの推定をおこなった．実験の結果，ライセンスのルール作成プロセスにおいて，機械学習を用いたテキスト分類が有用であることがわかった．

1. はじめに

生産性向上の手段として，ソフトウェアの再利用は必要不可欠なものとなっている [9]．特に近年では，オープンソースソフトウェア（以降，OSS と呼ぶ）を利用したソフトウェア開発が盛んに行われている．ソフトウェアを再利用する際には，著作者からライセンスを得ることが義務付けられている．OSS の場合，一般的には，ソースコードのコメントアウト部分に明示されている，ライセンスを遵守することで OSS の再利用が可能となる．

ただし，一つのソフトウェアが複数の異なるソフトウェアを再利用し構成される場合，複数のライセンスが，一つのソフトウェアに共存することが想定される．OSS を含めたライセンスには不整合問題というものが存在する．ライセンス不整合問題とは，異なるライセンスの条項間に矛盾した条項が存在し，両方のライセンスに違反している状態のことを指す．ライセンスに違反すると，最悪の場合，訴訟問題に発展する恐れがある．例えば，2007 年に GPL 違反で提訴され VoIP 電話機の仮販売差止め処分に至った Skype 社などの事例がある [1]．ライセンス違反の回避を目的として，ライセンスをソースファイル単位で機械的に特定する手法やツールが複数提案されている．German らの研究 [5] では，Ninka と呼ばれるライセンス特定ツールが，現時点で最も精度の高いツールであることを示している．ただし，ライセンス記述が検出された場合に，他の手法やツールは何らかの既知ライセンスを出力として返す（誤判別）するのに対して，Ninka は，未知ライセンスの存在が検出されたことを出力する．しかし，未知ライセンスそのものは，目視により確認する必要がある．

本研究では，まず，特定を行ったファイルの内，Ninka

[1] OSS ライセンスの比較および利用動向ならびに係争に関する調査:http://www.ipa.go.jp/files/000028335.pdf

が未知ライセンスと判別したソースファイルが，どの程度の割合を占めているかを予備的に調査する．調査の結果，特定を行ったソースファイルの内，未知ライセンスファイルが大量に含まれていることがわかった．すなわち，Ninka は誤検知そのものは少ないが，未知ライセンスとして判定したライセンスを多く含むことがわかった．次に，大量の未知ライセンスファイルをすべて目視により調査しライセンスを特定するには相当な労力がかかるため，未知ライセンスファイルをクラスタリング（分類）し，クラスタの代表的なファイルを目視するのみでライセンスを特定できるかどうかを調査した．本研究では，Debian v7.8.0 のソフトウェアパッケージを入力として，Ninka が判別した未知ライセンスファイル集合に対して K-means 法を適用し，ライセンス記述の分類精度を定性的および定量的評価を行った．また，Random Forest 法を用いて，類似ライセンスの推定をおこなった．実験の結果，ライセンスのルール作成プロセスにおいて，機械学習を用いたテキスト分類が有用であることがわかった．

2. 用語

本章では，本論文で用いる用語を定義する．

ソフトウェアライセンス ソフトウェアの利用許諾である．ソフトウェアの利用者は，ライセンスを遵守することで，ソフトウェアを利用することができる．ライセンスはソフトウェア全体だけではなく，ソースファイルについても決まっている．ライセンスには条項の修正を受けた，複数のバージョンが存在する．例えば，GPL には現在 3 種類のバージョンが存在するが，全て別のライセンスである．本論文では，ライセンスが複数のバージョンを持つ場合，特定のバージョンを示すために接尾辞 v を用いる．また本論文では “or later” を接尾辞+で表現する．“or later” とは，例えば，GPL のライセンス記述に “either version 2 of the License, or (at your option) any later version” と記載されている場合，version 2 以降のバージョンのうちのどれかでの再頒布が可能となる．表 1 に本論文で使用するライセンスの名前と略称を示す．

ライセンス記述 ライセンスの明示を行っている文章である．主にソースコードのコメントアウト部分にて明示される．ライセンス記述には，ライセンス全文が載っているものと，ライセンス全文への参照が記述されているものがある．

表 1. 本論文で使用する一般的なライセンスとその略称

略語.	説明
Apachev2	Apache License version 2.0
BSD4	初期の BSD License，BSD4clause License
BSD3	BSD3clause License
BSD2	BSD2clause License
CeCill	Cea Cnrs Inria Logiciel Libre License
CDDL	Common Development and Distribution License
CPLv1.0	Common Public License version 1.0
EPLv1.0	Eclipse Public License version 1.0
FreeType	FreeType License
GPLv1	General Public License version 1
GPLv2	General Public License version 2
GPLv3	General Public License version 3
GPLnoVersion	GPL with no version indicated
LGPLv2.1	Lesser General Public License version 2.1
LGPLv3	Lesser General Public License version 3
LibraryGPLv2.0	Library General Public License version 2.0
MIT/X11	MITLicense，MIT が X11 のためにリリースしたライセンス
MITold	昔の MIT/X11
MPLv1.1	Mozilla Public License version 1.1
Python	Python License
PostgreSQL	Postgre SQL License
SameAsPerl	Perl に適応されているライセンス
SeeFile	別のファイルへの参照
ZLIB	ZLIB/libpng license
publicDomain	パブリックドメインとされているライセンス

未知ライセンスファイル Ninka がライセンス特定の際，未知のライセンスと判別したソースファイルである．

未知ライセンス記述 未知ライセンスファイルのライセンス記述である．

類似ライセンス 既存ライセンスをベースに作られたライセンスであり，ライセンス記述が類似しているライセンスである．

ライセンスルール 正規表現などのルールベースによるライセンスの特定に必要な知識．

3. ライセンス特定手法

ソフトウェアが複数の異なるソフトウェアを再利用し構成される場合，複数のライセンスが，一 つのソフトウェアに共存することが想定される．ライセンスには，不整合問題が存在する．ライセンス不整合問題とは，異なるライセンスの条項間に矛盾した条項が存在し，両方のライセンスに違反している状態である．

ライセンスの不整合問題を回避するため，ソースファイル単位でのライセンス特定を行う必要があるが，全

てのソースファイルのライセンスを目視で確認するのは相当な労力がかかる．そのため，正規表現などのルールベースでソースファイルのライセンスを機械的に特定する手法やツールが複数提案されている．Ninka[5] は，ライセンス記述を複数の文に分解したライセンス文を特定し，ライセンス文で構成される既知ライセンスを特定結果として出力する．Fossology[6] は，b-SAM[2] アルゴリズムを用いて既知のライセンス記述とソースコード中のコメントを比較し，最も類似している既知ライセンスを特定結果として出力する．他にも，単純な正規表現を利用したソフトウェアライセンス特定ツールも存在する．OSLC[3] はソースコード中のコメントとライセンス記述間での最長一致列を類似度とする．次に最も類似度が高かったライセンスを特定結果として出力する．Ohcount[4] は各ライセンス記述に対応した正規表現を用いて，ソースコード中のコメントにマッチした正規表現からライセンスを特定する．

現時点で，どの手法が最も有用なのか検証するため，German ら [5] は各ツールを Debian v5.0.2 からランダムに選出した 250 のソースファイルのライセンスを各ツールで特定し，その性能の評価を F-尺度法を用いて行った．その結果，最も F 値が高かったツールは Ninka であった．ただし，ライセンス記述が検出された場合に，他の手法やツールは何らかの既知ライセンスを出力として返す（誤判別）するのに対して，Ninka は，未知ライセンスが存在が検出されたことを出力する．しかし，未知のライセンスそのものは手作業によりライセンスの確認を行わなければならない．そこで本研究では，未知ライセンスファイルに着目した分析を行う．

4. 予備調査

Ninka が特定したソースファイルの内，未知ライセンスファイルがどの程度の割合を占めているかを予備的に調査する．データセットとして，Firefox v25.0[5] のソースファイルを選んだ．対象とする言語は Java (.java), C (.c , .h), C++ (.cpp, .cxx, .cc, .hxx), Lisp (.el), Perl (.pm and .pl), Python (.py), Ruby (.rb), ShellScript (.sh), Makefile (.mk) である．これらの拡張子を条件にソースファイル集合を作成した結果，Firefox v25.0 では 22,413 ファイルとなった．表 2 に作成した Firefox v25.0 のソースファイル集合のライセンスを Ninka で特定したライセンスの内，上位 10 件のライセンスを示す．

表 2. FireFox v25.0 のライセンス（上位 10 件を表示）

Ninka 特定結果	ファイル数
NONE	1,390
spdxBSD3	959
Apachev2	752
FreeType	481
BSD3	240
MPL1_1andLGPLv2_1,MPLv1_1	204
SeeFile	143
MITX11	135
spdxBSD2	90
MITmodern	60
UNKNOWN	17,678

特定に成功したライセンスのうち，未知ライセンスファイルは 17,678 ファイル存在し，特定を行ったソースファイルの内，約 79%を占めていた．予備調査の結果，Firefox v25.0 は，大量の未知ライセンスファイルが含まれていることがわかったが，大量の未知ライセンスファイルのライセンスを目視で特定するのは相当な労力がかかる．そのため，未知ライセンスファイルのライセンスを特定できなければ，実用的とはいえない．未知ライセンスを特定するには，新しくライセンスルールを作成する必要がある．

5. ライセンスルール作成プロセスの問題点

5.1. ライセンスルール作成プロセス

Ninka のライセンスルール作成プロセスは以下の 4 つの手順から成る．

1. 未知ライセンスファイルを調査し，追加する未知ライセンス記述を決定する．
2. 未知ライセンス記述を複数のライセンス文に分解する．
3. 既知なライセンス文が存在しないか調査し，ユニークなライセンス文を決定する．

[2] http://fossology.org/symbolic_alignment_matrix
[3] http://sourceforge.net/projects/oslc/
[4] http://github.com/blackducksw/ohcount
[5] ftp://ftp.mozilla.org/pub/mozilla.org/firefox/releases/25.0/source/irefox-25.0.source.tar.bz2

4. 新規に追加するライセンス文の正規表現を作成し，ライセンスルールを追加する．

手順 1 では，ライセンス特定ツールに追加すべきライセンスを決定するため，未知ライセンスファイル集合を調査する．ここでは，未知ライセンスファイル数に対して多くを占めているライセンスがライセンス特定ツールに追加すべきライセンスである．

手順 2 では，未知ライセンス記述を複数のライセンス文に分解する．このプロセスは Ninka を用いて行う．Ninka はライセンス特定時にオプションを指定することで中間生成ファイル “ファイル名.goodsent”（以降では，goodsent ファイルと呼ぶ）を出力する．goodsent ファイルはライセンス記述を抽出し，Ninka によってライセンス文に分解されたライセンス記述が出力されているファイルである．追加すべきライセンスのソースファイルに対して Ninka を実行し，goodsent ファイルを出力させることができれば，手順 2 は完了する．

手順 3 は，手順 2 で分解したライセンス文の一部は，既知である場合がある．同じライセンス文が違う名前で登録されることになってしまい，正しいライセンス特定が行えなくなるため，ライセンス特定ツールにとって既知であるライセンス文を調査し，追加するライセンス文と，同一のライセンス文を明らかにする必要がある．Ninka の場合，ライセンス文は表記ゆれを考慮した拡張正規表現で登録されているため，単純な検索だけでは，このプロセスは行うことはできない．

最後に，手順 4 では，新しく追加するライセンス文の正規表現を作成し，ライセンスルールを追加する．

5.2. ライセンスルール作成プロセスの問題点

ライセンスルール作成プロセスにおける問題点として，ライセンスルール作成プロセスは労力がかかると考えられる．特に労力がかかる手順は，手順 1，手順 3 がある．手順 1 は，未知ライセンス集合のライセンスを目視で確認し，未知ライセンスファイル集合に存在するライセンスの内訳を明らかにする必要がある．手順 3 は，ライセンス特定ツールの知識の調査を行わなければならない．ライセンスには，ライセンス記述が酷似している類似ライセンスが多数存在する．図 1 の Python と PostgreSQL のような類似ライセンスは他にも存在するため，ライセンス記述の違いを見極めなければならないライセンスを全て探す必要がある．

（PostgreSQL の類似箇所）
Permission to use, copy, modify, and distribute this software and its documentation for any purpose, without fee, and without a written agreement is hereby granted, provided that the above copyright notice and this paragraph and the following two paragraphs appear in all copies.

（Python の類似箇所）
Permission to use, copy, modify, and distribute this software and its documentation for any purpose and without fee is hereby granted, provided that the above copyright notice appear in all copies and that both that copyright notice and this permission notice appear in supporting documentation, and that the name of Stichting Mathematisch Centrum or CWI not be used in advertising or publicity pertaining to distribution of the software without specific, written prior permission.

図 1. 類似ライセンス例（Python と PostgreSQL の例）

以上により，ライセンスルール作成プロセスを自動化できることが望ましい．しかし，自動化のためには，未知ライセンスファイル集合からライセンスルールに追加すべきライセンスを抽出することがまず必要になる．追加すべきライセンスを抽出する手段の 1 つとして，機械学習を用いたテキスト分類が挙げられる．テキスト分類を行うことで，類似する未知ライセンス集合を抽出することが容易になると考えられる．ただし，機械学習でのテキスト分類をライセンスに対して行った研究はまだない．ライセンスルール作成プロセス自動化の支援を行うには，まず，機械学習を用いたテキスト分類の限界を明らかにする必要がある．そこで本研究では，機械学習を用いたテキスト分類が，既存 OSS ライセンス特定手法のルール作成プロセスの支援にどれくらい有用かを明らかにすることを目的とした分析を行う．

6. 分析方法

本章では，ライセンスルール作成プロセスである手順 1，手順 3 の支援に，機械学習を用いたテキスト分類がどれくらい有用であるかを明らにするための分析方法を説明する．

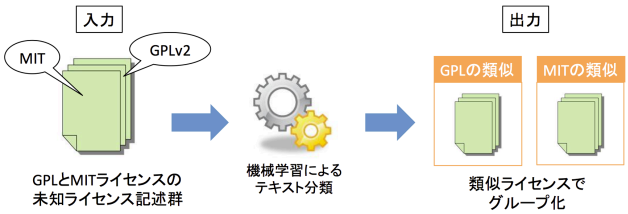

図 2. 分析方法の概要

具体的には，図 2 のように，GPL や MIT で構成されている未知ライセンス記述群を入力とし，GPL と MIT の 2 つの類似ライセンスのグループにクラスタリングすることがどれくらいできるかどうかを明らかにする．

機械学習を用いたテキスト分類を行うには，まず，未知ライセンス記述を単語文書行列 (TDM) を用いて数量化する前処理が必要である．

次に，手順 1 を支援するため，前処理後の未知ライセンス記述群に対して K-means 法によるクラスタリングを行う．クラスタリングにより，一つのライセンスから成るクラスタを生成することができれば，追加すべきライセンスを探すための調査が容易に行えると考えられるため，手順 1 を支援することができる．

次に，手順 3 を支援するため，前処理後の未知ライセンス記述に対して Random Forest 法による類似ライセンスの推定を行う．未知ライセンス記述の類似ライセンスを推定することができれば，ライセンス特定ツールの知識内に存在する既知のライセンス文を把握することが容易に行えるようになると考えられるため，手順 3 を支援することができる．

本実験で用いる K-means 法のクラスタ数，Random Forest で推定を行う類似ライセンスは，OSI が発表している主要なライセンスを参考に決定した．本実験で推定を行う類似ライセンスを表 3 に示す．OSI が発表している主要なライセンスとは，人気でかつ広く使用されているライセンス [6]13 種類と MPLv1.1 を含めた 14 種類である．MPLv1.1 を含めた理由は，MPLv1.1 はマルチライセンスを採用した古くから用いられているライセンスであり，ソフトウェア企業が，独自のライセンスを作成する際の雛形とすることが多いからである．K-means 法の最適クラスタ数は不明なため，本研究で用いる主要なライセンスに従って 14 とし，Random Forest 法の学習データは表 3 の OSS プロジェクトから，100 ファイル分のライセンス記述をコメントアウト単位で抽出した．

表 3. 推定を行う類似ライセンス及び取得先

ライセンス	ライセンス記述の取得先	取得した日付
Apachev2	OpenOffice v3.4.0	2014/11/16
BSD3	j-gitv3.5.1	2014/10/14
BSD2	OpenSSh v6.7p1	2014/11/19
GPLv1	Gcc v2.9.0	2014/11/21
GPLv2	Gcc v3.0.1	2014/11/29
GPLv3	Gcc v4.9.2	2014/11/16
LGPLv2.1	Glibc v2.20	2014/11/20
LGPLv3	OpenOffice v3.3.0	2014/10/18
LibraryGPLv2.0	Gcc v4.9.2，Glibc v2.2.0，FireFox v9.0	-
MIT	FireFox v33.0	2014/11/15
MPLv2.0	FireFox v33.0	2014/11/15
MPLv1.1	FireFox v9.0	2014/11/20
CDDLv1.0	zfs-fuse v0.6.9	2015/2/2
EPLv1.0	Eclipse MARS	2014/11/19

以降では，分析方法の各処理（前処理，クラスタリング，類似ライセンス推定）の詳細を説明する．

6.1. 前処理：TDM の構築

文章を数量化する方法の一つとして，単語文書行列 (Term-DocumentMatrix: TDM) を構築する方法がある．TDM とは全文書に現れた単語一つ一つが行，各文書を列とした行列であり，総単語数を N，総文書数を M とすると $N \times M$ の行列となる．その [i , j] 番目の要素は，単語 i が文書 j に現れる回数に対応する．次に TDM を構築するまでの過程と処理を順に説明する．

最初に改行の除去を行う，未知ライセンス記述の改行を空白に置換する．次に，未知ライセンス記述の数値の英単語化を行う．ライセンスには複数のバージョンが存在することを 2 章で述べた．このバージョンの値を学習させたいため，数値を英単語として扱う．1 → one，2 → two のように 1–9 までの数値を英単語に置換する．

次に，未知ライセンス記述から英語の文書によく出てくる単語，例えば，“i”，“me”，“my” などの人称代名詞や “and”，“but” などの接続詞などを除去するため，ストップワードの設定を行う．ストップワードには，R のテキストマイニングパッケージ [7] で利用でき

[6] http://opensource.org/licenses/category

[7] http://cran.r-project.org/web/packages/tm/tm.pdf

る $stopwords(kind = "en")$ で表示される単語 174 語を設定する．最後に，R のテキストマイニングパッケージで利用できる関数 $TermDocumentMatrix$ と設定したストップワードを用いて TDM を構築する．ただし，Random Forest 法においては，実験に使用した計算機環境 [8] では，計算時間が膨大となったため，学習データと未知ライセンスファイルから構築した TDM から，1 回のみ出現した単語を削除し，再構築した TDM を用いる．

6.2. K-means 法によるテキスト分類

未知ライセンス記述群のテキスト分類に，教師なし機械学習アルゴリズムである K-means 法 [8] を用いる．テキスト分類に K-means を用いた理由は，未知ライセンス記述に対して機械学習を用いたテキスト分類を行った例がなく，クラスタリング結果が予想できないことから，複雑なクラスタリングアルゴリズムを使用すると結果の考察が困難になると考えたからである．本研究で用いる K-means 法は，クラスタ数を 14，類似度はユークリッド距離とした．

6.3. Random Forest 法による類似ライセンス推定

クラスタの類似ライセンスの推定として Random Forest 法 [2] を用いる．テキスト判別手法に Random Forest を用いた理由は，多値分類アルゴリズムの中で精度が最も高いことで知られているためである．クラスタの類似ライセンスは，各クラスタの未知ライセンス記述の類似ライセンス推定結果を多数決することにより決定する．Random Forest 法により，未知ライセンス記述群から構築した TDM を，表 3 から取得した学習用のライセンス記述から構築した TDM を学習し，類似しているライセンスを出力する．

7. 実験

本実験は，機械学習を用いたテキスト分類がライセンスルール作成プロセスの支援にどれくらい有用であるかどうかを明らかにすることが目的である．6 章で述べた分析方法を用いて，OSS ライセンスルール作成支援プロセス手順 1，手順 3 にどれくらい有用かどうかを明らかにするための実験を行う．本実験では Debian v7.8.0 の各ソフトウェアパッケージから 1 つのソースファイルをランダムに選び，Ninka でライセンスを特定できなかった未知ライセンス記述群をデータセットとして抽出する．

7.1. 評価基準

本研究では特に労力がかかるプロセスは，手順 1, 手順 3 であると述べた．以下の 2 点が確認できれば，手順 1 を支援できるとする．

- 評価基準 A. ライセンス記述が存在しないクラスタは存在するか
- 評価基準 B. 一つのライセンスで 75%以上を占めているクラスタは存在するか

評価基準 A は，ライセンス記述が存在しないクラスタが存在すれば，明らかに調査の対象外な未知ライセンス記述を除外できる可能性があるということである．評価基準 B は，一つのライセンスで 75%以上を占めているクラスタが存在すれば，クラスタの選別などを行うことで，容易に追加すべきライセンスを抽出できそうであると考えられる．評価基準 B の判定を行うとき，未知ライセンス記述数が 10 以下のクラスタは除外する．評価基準 A,B を満たしていれば，テキスト分類が OSS ライセンス特定ツールのライセンスルール作成プロセス支援に有効であるとする．

次に，以下を確認できれば，手順 3 を支援できるとする．

- 評価基準 C. 目視により特定したクラスタのライセンスの類似ライセンスであったクラスタが 50%以上存在したか

Random Forest で推定した，クラスタの類似ライセンスが，目視により特定したクラスタのライセンスの類似ライセンスであったクラスタが 50%以上存在すれば，類似ライセンス推定を全く行わない状況と比べると，違いを見極める必要のある類似ライセンスを探すのに有用であるとする．なお，類似ライセンスかどうかは，ライセンス記述と一致している文章が存在すれば類似ライセンスとする．

7.2. データセット

本実験では Linux ディストリビューションである Debian v7.8.0 のソフトウェアパッケージのソースファイル [9]

[8] OS:OS X Yosemite, CPU:2.3GHz Intel Core i7, メモリ:16GB

[9] http://ftp.riken.jp/pub/Linux/debian/debian-cd/7.8.0/source/iso-dvd/

の未知ライセンス記述群をデータセットとして抽出した．Debian には 48,559[10] のソフトウェアパッケージを含んでいるため，多数のライセンスを含んでいる．本実験では “.zip”， “.tar.gz” ， “.tar.xz” ， “.tar.bz2” をソースコードアーカイブと見なし解凍した．ソースファイルの拡張子は，Gonzalez ら [7] によって報告された Debian でよく使われているプログラミング言語 8 種類のうち，シェルスクリプトと PHP を除く，6 種類のプログラミング言語の拡張子，Java (.java), C (.c), C++ (.cpp , .cxx , .cc), Lisp (.el ， .jl), Perl (.pm , .pl), Python (.py) を選んだ．これらのプログラミング言語で書かれている 12,725 パッケージから各 1 ファイルをランダムで抽出し，Ninka でライセンス特定を行った結果，ソースファイル 12,725 ファイルのうち未知ライセンスファイルは 2,838 ファイルであった．この未知ライセンスファイル集合から goodsent ファイルに書かれている文章を未知ライセンス記述群として取得しデータセットとする．

7.3. 実験結果

7.3.1 K-means のクラスタリング結果

未知ライセンス記述群を 14 のクラスタに分類した．クラスタの未知ライセンスを目視で特定し，各クラスタで最も多かったライセンスのうち上位 3 つと，クラスタのデータ数をまとめたものを表 4 に示す．表 4 において，ライセンス記述が無かったクラスタは，クラスタ 1，9，13 の 3 クラスタで確認できた．一つのライセンスで 75%以上を占めているクラスタは，クラスタ 4，7，12 の 3 つのクラスタで確認できた．特に，クラスタ 7 に関しては，ZLIB が 95%以上を占めている．以上から，K-means 法によるクラスタリングは，評価基準 A，B を満たしているため，ライセンスルール作成プロセスに有用であるとする．

7.3.2 Random Forest 法の類似ライセンス推定結果

Random Forest 法による各クラスタの類似ライセンス推定結果を表 5 に示す．各クラスタのライセンスは，各クラスタの未知ライセンス記述のライセンス特定，あるいは推定した結果から多数決を行い決定する．目視により特定したクラスタのライセンスと Random Forest 法により推定したライセンスが類似していた場合，推定に成功したとし，クラスタのライセンスが Random Forest 法が推定可能なライセンスではなかった場合とクラスタのファイル数が 10 以下の場合は推定不可とし評価には含めない．推定が可能であった 6 クラスタのうち，4 クラスタは類似ライセンスの推定に成功した．その精度は 4/6 = 約 67%（小数点第 2 位四捨五入）である．クラスタ 12 に関しては Random Forest 法により推定したクラスタの類似ライセンスは，目視で確認したクラスタの類似ライセンスと一致している．

次に，類似ライセンスが一致しているクラスタ 12 以外の類似ライセンスの推定が成功したものに関して類似箇所を示す．

LGPLv3+ と GPLv2+の類似箇所

LGPLv3 と GPLv2 は大部分が同一の記述であり，異なる箇所は，バージョンの数字の違いと，“GNU General Public License” と “GNU Lesser General Public License” の違いの 2 箇所のみである．

MIT と ZLIB の類似箇所

MIT と ZLIB の類似箇所はいくつか存在する．MIT の免責条項 “THE SOFTWARE IS PROVIDED ‘AS IS’, WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED,” と ZLIB の冒頭にある免責条項 “This software ‘as-is’, without any express or implied warranty.” の箇所では，大文字表記と小文字表記の点に違いがあるが，文章は類似している．もう一つは MIT の冒頭にある “Permission is hereby granted” と ZLIB の “Permission is granted” である．

BSD2 と BSD3 の類似箇所

表 4. K-means 法によるクラスタリング結果

クラスタ	目視によるライセンスの内訳	データ数
1	NONE	23
2	GFDL ライセンス全文	1
3	NONE，GPL，GPLv2	382
4	SameAsPerl（248/312），GPL，GPLv2	312
5	Apachev2，Cecil，LGPLv2.1+	55
6	GPLv2+，GPL，LGPLv2.1+	278
7	Zlib（41/46），SOFA，Other	43
8	BSD3，BSD2，BSD-style	85
9	NONE	2
10	CC の全文	1
11	CPL のライセンス全文	1
12	MIT（105/118），BSD4	118
13	NONE	1
14	Copyright，SeeFile，ライセンス名のみ	1536

[10] https://packages.debian.org/stable/allpackages?format=txt.gz

表 5. Random Forest 法による類似ライセンス推定

クラスタ	目視の結果	正解集合	推定結果	推定の成否
1	NONE	なし	MIT	-
2	GFDL	GFDL	MPLv1.1	-
3	NONE	なし	MIT	-
4	SameAsPerl	GPLv1，GPLv2，GPLv3，LGPLv2.1，LGPLv3	MIT	失敗
5	Apachev2	Apachev2，BSD2，BSD3	MIT	失敗
6	GPLv2+	GPLv1，GPLv2，GPLv3，LGPLv2.1，LGPLv3	LGPLv3	成功
7	ZLIB	MIT	MIT	成功
8	BSD3	Apachev2，BSD2，BSD3	BSD2	成功
9	NONE	なし	MIT	-
10	CC	CC	BSD2	-
11	CPLv0.5	なし	BSD2	-
12	MIT	MIT	MIT	成功
13	NONE	なし	MIT	-
14	Copyright	なし	MIT	-

BSD2 と BSD3 は大部分が同一の記述であり，この 2 つのライセンス記述の異なる箇所は “3.Neither〜” 条項が存在するか否かのみである．

以上により，評価基準 C を満たしているので，Random Forest 法を用いた類似ライセンスの推定は，ライセンスルール作成プロセス支援に有用ではないかと考えられる．

8. 考察

8.1. ライセンスルール作成プロセス支援の可能性

データセットのクラスタリングを行った結果，1 つのライセンスで 75%以上を占めているクラスタが 3 つのクラスタで確認できた．特にクラスタ 7 においては，ZLIB ライセンスが 95%以上を占めていた．しかし，各クラスタのライセンスの内訳は，クラスタ内の未知ライセンス記述を目視で確認しなければ，把握することができない．各クラスタのライセンスの内訳を把握するには，同一ライセンスであるライセンス記述の定義を行い，ライセンス特定ツールに追加すべきライセンスを各クラスタから抽出することができれば，ライセンスルール作成プロセスの手順 1 を支援することが可能である．また，クラスタ 14 に 1536 の未知ライセンス記述がクラスタリングされた．クラスタ 14 は，Copyright のみや，ライセンス名のみなど，比較的短い文章が多数存在した．ライセンス名のみの記述については，ライセンス名が未知であった場合，ライセンスルールに追加するなど，未知のライセンス名の抽出にも用いることができる．

各クラスタの類似ライセンスの推定を行った結果，約 67%の精度で，類似ライセンスを推定することができた．本実験では，推定を行うライセンスを 14 種類に設定し，類似ライセンスの推定を行った．しかし，未知ライセンスファイル集合には，14 種類以外のライセンスが多数含まれていることが考えられる．推定を行うライセンスの種類を増やすことで，精度の向上が期待できる．また，ライセンス記述がないクラスタが確認できたことから，ソースコードを学習し，ライセンス記述がないクラスタの推定が可能であると，ライセンスルール作成プロセスの手順 1 の支援においても有用であると言える．

8.2. 類似ライセンスの推定結果

本実験で行った類似ライセンスの推定では，Random Forest 法の予測結果は MIT とされることが多かった．クラスタ単位では 14 クラスタ中 9 クラスタに，2,838 の未知ライセンス記述の内，2,415 ファイルが MIT ライセンスと推定された．MIT は類似ライセンスが多いことでも知られている．Fedora Project はソースファイル上で見つかった 36 種類の MIT の類似ライセンスを報告している [11]．また MIT ライセンスには，BSD，Apachev2，ZLIB などにも記述される免責条項が存在する．以上のことから MIT ライセンスの類似ライセンスであると推定されるライセンスが多いのではないかと考えられる．

[11] https://fedoraproject.org/wiki/Licensing:MIT

8.3. 企業におけるライセンス特定ツールの利用シナリオ

ライセンス特定ツールは，特定精度が 100%でない限り，企業の法務部門などではすべてのライセンスを目視で確認する必要があり，ライセンス特定ツールの利用自体の意義が不明確な場合がある．

我々は，現時点でライセンス特定ツールを利用せず目視ですべてのライセンスを確認している企業にも，ライセンス特定ツールを利用することに意味があると考えている．すべてのライセンスを目視で確認する場合，本論文で紹介したように軽微な表記ゆれを見落とす可能性があり，目視であっても特定精度が 100%になるとは限らない．そのため現状では，2 名以上の担当者でダブルチェックするなどして対応していると思われる．

ライセンス特定ツールによる特定結果は，人為的ミスを防ぐためのダブルチェックとしても用いることができると我々は考えている．ライセンス特定ツールと目視による特定結果に差異があれば，いずれかの方法に誤りがある可能性が高い．特に，本論文の実験で用いた Debian のように，膨大な数のライセンスを確認しなければならない状況においては，目視による確認にもミスが増えると考えられる．ライセンス特定ツールをダブルチェックの手段として利用することで，作業コストの大きな削減にもつながると期待できる．

8.4. 制約

本研究では，OSI の主張する主要なライセンスと MPLv1.1 を含めた 14 ライセンスのみを用いて分析を行った．そのため本研究の実験結果は一般性が高いとは言えない．K-means 法によるテキスト分類時のクラスタ数を，それぞれのプロジェクトのライセンス分布に基づいたクラスタ数にするとクラスタリングや Random Forest 法の類似ライセンス推定の精度向上が期待できる．

本研究では，支援すべきライセンス特定ツールのルール作成支援プロセスとして Ninka のみを選んでいる．そのため本研究の実験結果は一般性が高いとはいえない．特に，Ninka が未知ライセンスを判別する仕組みに着目しているため，他のライセンス特定ツールのルール作成支援プロセスと異なる部分が生じる場合がある．しかし，どのようなライセンス特定ツールもライセンス記述を調査する部分は共通しているはずなので，本実験結果が部分的には当てはまると考えられる．

9. 関連研究

9.1. ライセンス特定に関する研究

本論文でも紹介したライセンス特定に関する研究は，他にも行われている．例えば，Tunnanen ら [10] は，ライセンスを指定する記述と適合する正規表現を作成し，どの正規表現に適合するかによりライセンスを特定する方法を提案している．ただし，特定されていないソースファイルについては，利用者自身がライセンスの正規表現を作成しなければならない．眞鍋ら [11] は，ライセンス特定支援に関する研究を行っている．ライセンス記述には共通した記述が用いられることに着目し，頻出文字列を提示することで作成すべきライセンス記述のための正規表現を作成支援する手法を提案している．本研究では，K-means 法による単語の出現頻度による類似度に基づいたテキスト分類を行っている部分で，眞鍋らのアプローチとは異なっている．

9.2. ライセンス不整合問題に関する研究

ライセンス不整合問題を解決するために，Agawal ら [1] は，ライセンスの条項をタプルで表現したライセンスメタモデルを構築し，ArchStudio4 上で義務の衝突，利用できる権利を計算する手法を提案している．また，German ら [4] は，ライセンス不整合を起こした場合，ライセンス不整合をどのように解消するか，ライセンサーとライセンシの観点から解決方法をまとめている．これらの研究と本研究は，ライセンス不整合問題を問題意識として持っているが，本研究では，ライセンス特定ツールにおけるライセンスルール作成支援を行うことで，ライセンス不整合問題を解決しようとしている．

9.3. ライセンスとソフトウェア開発者に関する研究

ソフトウェア開発者のライセンスに対する認識を分析した研究がいくつか行われている．例えば，Colazo ら [3] は，コピーレフト性のある OSS ライセンスを用いる OSS プロジェクトはコピーレフト性のない OSS ライセンスの OSS プロジェクトと比べて，開発者の地位やコーディング量，活動の永続性が高く，開発期間が短くなることを示している．Sojer ら [9] は，ソフトウェア開発者にソフトウェアの再利用に関するインタビューとライセンスに関するクイズを実施したところ，65%のソフトウェア開発者は，ソフトウェアの再利用に関する知識を，

主にインターネットのような非公式な情報を元に勉強していることがわかった．また，ライセンスに関するクイズから開発者自身のライセンスに関する知識を過剰評価していることがわかった．この結果から，効果的なソフトウェア再利用に関する勉強方法が確立されていないことを示した．これらの分析結果を踏まえ，本研究では，ソフトウェア開発者のライセンスに関する知識が不十分であることを前提とし，ソフトウェア開発者が容易にライセンスルールの作成を行えるようにするための支援を最終目標にしている．

10. おわりに

本研究では，機械学習を用いたテキスト分類がルール作成支援にどれくらい有用かどうかを明らかにすることを目的として，K-means 法によるテキスト分類，Random Forest 法による類似ライセンスの推定を Debian v7.8.0 のソースファイルから抽出した未知ライセンス記述群に対して行い結果を分析した．本研究ではルール作成支援プロセスを 4 つに分割して定義し，手順 1 の追加すべきライセンス記述の調査，手順 3 の既知ライセンス文の調査を支援対象とした．ライセンス記述を K-mean 法を行うと追加すべきライセンスルールを容易に発見できる可能性があることを述べた．Random Forest 法でクラスタの類似ライセンスの推定を行うと，約 67%の精度で類似ライセンスを推定できることを述べた．

今後の課題としては，実用的な手法を提案するため，分類するクラスタ数や，推定を行う類似ライセンスの種類を増やすことで，より細分化が可能なクラスタリングや，より精度の高い類似ライセンス推定を可能にすること．また，同一ライセンスであるライセンス記述の定義を行い，ライセンス特定ツールに追加すべきライセンスを各クラスタから抽出可能にすることなどが挙げられる．

参考文献

[1] Thomas A. Alspaugh, Hazeline U. Asuncion, and Walt Scacchi. Analyzing software licenses in open architecture software systems. In *Proc. of ICSE '09 Workshop on Emerging Trends in FLOSS '09*, pp. 54–57, May 2009.

[2] Leo Breiman. Random forests. *Machine Learning*, Vol. 45, No. 1, pp. 5–32, Oct. 2001.

[3] Jorge Colazo and Yulin Fang. Impact of license choice on open source software development activity. *American Society for Information Science and Technology*, Vol. 60, No. 5, pp. 997–1011, May 2009.

[4] Daniel M. German and Ahmed E. Hassan. License integration patterns: Addressing license mismatches in component-based development. In *Proc. of ICSE '09*, pp. 188–198, May 2009.

[5] Daniel M. German, Yuki Manabe, and Katsuro Inoue. A sentence-matching method for automatic license identification of source code files. In *Proc. of ASE '10*, pp. 437–446, Sep. 2010.

[6] Robert Gobeille. The fossology project. In *Proc. of MSR'08*, pp. 47–50, May 2008.

[7] Jesus M. Gonzalez-Barahona, Gregorio Robles, Martin Michlmayr, Juan José Amor, and Daniel M. German. Macro-level software evolution: A case study of a large software compilation. *Empirical Software Engineering*, Vol. 14, No. 3, pp. 262–285, June 2009.

[8] J. A. Hartigan and M. A. Wong. A k-means clustering algorithm. *Royal Statistical Society*, Vol. 28, No. 1, pp. 100–108, 1979.

[9] Manuel Sojer and Joachim Henkel. License risks from ad hoc reuse of code from the internet. *Communications of the ACM*, Vol. 54, No. 12, pp. 74–81, Dec. 2011.

[10] Timo Tuunanen, Jussi Koskinen, and Tommi Krkkinen. Retrieving open source software licenses. In *OSS '06*, pp. 35–46, June 2006.

[11] 眞鍋雄貴, 市井誠, 早瀬康裕, 松下誠, 井上克郎. コメント中の頻出文字列を用いたソフトウェアライセンスの特定支援. ソフトウェア工学の基礎 XIV, 日本ソフトウェア科学会 FOSE2007, pp. 105–114, Nov. 2007.