Politechnika Wrocławska

Wrocław, 3 sierpnia 2018

dr hab. inż. Lech Madeyski, prof. nadzw. PWr
Katedra Inżynierii Oprogramowania
Wydział Informatyki i Zarządzania
Politechnika Wrocławska

RECENZJA ROZPRAWY DOKTORSKIEJ

Tytuł rozprawy: System do oceny podobieństwa kodów źródłowych w językach funkcyjnych oparty na metodach uczenia maszynowego i agregacji danych

Autor rozprawy: mgr inż. Maciej Bartoszuk

Promotor: dr hab. inż. Marek Gągolewski, prof. PW, prof. PAN

Niniejsza recenzja została opracowana w odpowiedzi na list prof. dr hab. inż. Janiny Kotus, Prodziekana ds. Nauki Wydziału Matematyki i Nauk Informacyjnych Politechniki Warszawskiej, z dnia 27 czerwca 2018 r. informujący, że rada tegoż wydziału, na posiedzeniu w dniu 14 czerwca 2018 r., podjęła uchwałę powołującą mnie na recenzenta rozprawy doktorskiej mgra inż. Macieja Bartoszuka pt. "System do oceny podobieństwa kodów źródłowych w językach funkcyjnych oparty na metodach uczenia maszynowego i agregacji danych".

1 Zakres tematyczny rozprawy

Tematyka rozprawy dotyczy oceny podobieństwa kodów źródłowych w języku R. Naturalnym obszarem zastosowań rozwiązań prezentowanych w dysertacji jest wykrywanie plagiatów w kodach źródłowych (np. dostarczanych przez studentów w ramach zajęć projektowych). Innym, bardziej przemysłowym, obszarem zastosowań może być poprawa jakości kodu poprzez unikanie powtórzeń – tzw. klonów (ang. *code clones*). Oba obszary, choć ten drugi być może nawet częściej, są eksplorowane przez badaczy, o czym świadczy spora liczba publikacji naukowych. Zatem z jednej strony praca wpisuje się w istniejący nurt badań, z drugiej jednak wyróżnia się (co uważam za

mądry krok) skupiając się na języku R, który dostarcza wygodnych mechanizmów tworzenia i manipulacji funkcjami. Prac dotyczących oceny podobieństwa kodów źródłowych, czy detekcji klonów w języku R jest znacznie mniej niż w przypadku takich języków jak Java, z drugiej strony język R w szybkim tempie zdobywa sobie coraz większą popularność, o czym może świadczyć gwałtownie rosnąca liczba pakietów w bazowym repozytorium CRAN (ang. *Comprehensive R Archive Network*), czy publikacji (w tym także książek) poświęconych temu językowi lub go wykorzystujących w celu realizacji badań. Obok języka Python, R jest szczególnie często wykorzystywany w szeroko rozumianym obszarze *data science*, oferując użytkownikom najnowsze rozwiązania z obszaru statystyki, czy uczenia maszynowego. Można spotkać się ze stwierdzeniem, które może w nieco przesadny sposób to ilustruje, że *"jeśli jakichś metod statystycznych nie ma w R, to ich w ogóle nie ma"*. Zatem wybór tego właśnie języka programowania uważam za uzasadniony jego rosnącą popularnością i znaczeniem.

W rozprawie doktorant nie tylko opisał zaproponowane autorskie rozwiązania (m.in. w postaci algorytmów, sposobów reprezentacji funkcji, czy agregacji wyników zwracanych przez różne algorytmy) składające się na system do oceny podobieństwa kodów źródłowych w języku R wykorzystujący metody uczenia maszynowego, ale i przeprowadził ich gruntowną empiryczną ewaluację, a rezultaty prac umieścił w stworzonym przez siebie pakiecie w R, udostępniając jednocześnie serwis internetowy, umożliwiający porównanie kodów źródłowych.

W kontekście powyższych stwierdzeń, zakres i problematykę rozprawy można uznać za w pełni uzasadnioną, aktualną i ciekawą, zarówno w jej aspekcie teoretycznym, jak i praktycznym.

2 Zawartość i układ pracy

Opiniowana rozprawa obejmuje 184 strony i składa się z 6 rozdziałów oraz bibliografii zawierającej 134 pozycje literaturowe. Rozdział pierwszy zawiera krótkie wprowadzenie w przedmiot i zakres rozprawy, cele które postawił sobie doktorant, uzyskane oryginalne rezultaty rozprawy oraz listę publikacji doktoranta zawierających wybrane wyniki badań (co pozwala zorientować się gdzie doktorant umieścił swoje publikacje związane z doktoratem i z kim w trakcie realizacji badań współpracował). W rozdziale ujęto również, jak to się zwykle czyni, krótką prezentację struktury rozprawy i zawartości poszczególnych jej rozdziałów.

Rozdział drugi doprecyzowuje, jakie problemy mogą być rozwiązywane przy użyciu proponowanego systemu, a jakie pozostają poza jego zasięgiem, przywołując definicje plagiatu, wskazując na okoliczności, które mogą towarzyszyć popełnianiu plagiatu, a następnie przechodząc do definicji podobieństwa kodu oraz ich taksonomii. Doktorant wskazuje również na problem braku dobrych zbiorów benchmarkowych, które mogłyby umożliwić porównanie różnych podejść/algorytmów. Moim zdaniem na szczególną uwagę zasługuje próba zdefiniowania możliwych modyfikacji kodu (tzw. ataków na kod) i tworzenia danych testowych za pomocą generatora zbiorów benchmarkowych.

Rozdział trzeci skupia się na metodach porównywania par funkcji korzystając z ciekawego rozwiązania polegającego na oddzieleniu sposobów reprezentacji funkcji w języku R od samych algorytmów ich porównywania, co otworzyło interesujące możliwości w zakresie analizowania istniejących rozwiązań, jak i konstruowania nowych, z czego doktorant sprawnie skorzystał.

W rozdziale czwartym doktorant koncentruje sie na kwestii podejmowania decyzji (na podstawie miar podobieństwa z przedziału $[0, 1]$ zwracanych przez każdy z algorytmów), czy zakwalifikować

daną parę funkcji jako podobną czy nie, rozpatrując ten problem zarówno jako problem klasyfikacji binarnej jak i regresji, proponując również metodę agregacji modeli.

Rozdział piąty stanowi opis pakietu w języku R zawierającego implementację autorskiego algorytmu SimilaR oraz opis serwisu internetowego pozwalającego na porównanie kodów źródłowych.

Rozprawę zamyka rozdział szósty zawierający podsumowanie prac oraz kierunki dalszych badań w omawianym obszarze.

Zaprezentowana rozprawa prezentuje dość szczegółowo najważniejsze rezultaty objęte jej tematyką, w przejrzysty i logiczny sposób. Mimo, iż doktorant nie ustrzegł się pewnych bardzo drobnych potknięć językowych, to poziom edytorski, a także zakres rozprawy i sposób ujęcia prezentowanego materiału pracy oceniam jednoznacznie pozytywnie.

3 Uzyskane wyniki

Do kluczowych wyników prac i osiągnięć doktoranta można zaliczyć:

1. Sformułowanie operacyjnej definicji podobieństwa pary kodów źródłowych (*"Polega ona na wymienieniu możliwych modyfikacji kodu, a następnie stwierdzeniu, że kody A i B są do siebie podobne, jeśli A może powstać przy użyciu dobrze określonych przekształceń B."*).

2. Stworzenie zbiorów benchmarkowych, poprzez wylosowanie funkcji z popularnych pakietów, a następnie przekształcenie losowo wybranych funkcji przy użyciu wspomnianych modyfikacji kodu, co umożliwiło kontrolę rozmiaru, a w szczególności cech badanej grupy funkcji np., procentu funkcji będących klonami (czyli niezbalansowania zbioru) czy liczby dokonywanych przekształceń. Zastosowanie generatora zbiorów benchmarkowych w szczególności pozwala uniknąć subiektywnej i żmudnej oceny podobieństwa funkcji, bo wiemy dokładnie, które funkcje są wynikiem klonowania.

3. Oddzielenie algorytmów porównywania, pewnych ciągów lub grafów, od samej reprezentacji, prezentując dogłębną ocenę skuteczności wszystkich w ten sposób uzyskanych kombinacji. To z kolei umożliwiło zbadanie w jakim stopniu na skuteczność danego rozwiązania składa się przekształcenie kodu do konkretnej reprezentacji (zbadano trzy reprezentacje sekwencyjne tj. litery, tokeny i wywołania funkcji oraz jedną grafową tj. graf zależności programu), a w jakim stopniu algorytm porównywania (zbadano cztery dla postaci sekwencyjnych tj. odległość edycyjna, Smitha–Watermana, zachłannego kafelkowania ciągów i odległość q-gramowa oraz trzy dla reprezentacji grafowych tj. algorytmy McGregora, Weisfeilera–Lehmana (W–L) oraz SimilaR będący autorską modyfikacją algorytmu Weisfeilera–Lehmana).

4. Zbadano, zdaniem doktoranta po raz pierwszy, w kontekście detekcji klonów skuteczność algorytmu o złożoności wielomianowej opartego na metodzie Weisfeilera–Lehmana służącego do znajdowania podobnych grafów.

5. Zaproponowano autorską modyfikację algorytmu Weisfeilera–Lehmana, o nazwie SimilaR dostosowaną do specyfiki problemu znajdowania podobnych kodów źródłowych. Główny autorski wkład wydaje się mieć swoje źródła w obserwacji, że miara podobieństwa spada, gdy dwa wierzchołki otrzymują różne etykiety (ponieważ wektory zliczeń etykiet różnią się pomiędzy sobą) oraz w propozycji, by dwa wierzchołki, pomimo różnych sąsiadów (w sensie

etykiet), mogły otrzymać tę samą etykietę, jeśli ta różnica nie jest zbyt wielka. Wyniki badań empirycznych pokazują lepsze wyniki zmodyfikowanego algorytmu SimilaR w porównaniu z dotychczas zaproponowanymi metodami w sensie miary F.

6. Zaproponowano metodę agregacji wyników zwracanych przez różne algorytmy, aby właściwiej podejmować decyzje o podobieństwie dwóch kodów źródłowych. Jest ona oparta o krzywe i płatki B-sklejane. Podejście to ma tę zaletę, że pozwala zachować dobrze interpretowalną postać modelu. Badania empiryczne pokazały, że (jak należało się spodziewać) lepsze wyniki są osiągane dla zestawu metod, niż każdej z nich użytej pojedynczo. Ponadto zaproponowana metoda agregacji okazała się obiecująca w porównaniu do innych technik uczenia maszynowego, w tym m.in. lasów losowych.

7. W kontrze do powszechnie stosowanego symetrycznego podejścia, w którym zwracana jest jedna wartość, która opisuje, jak podobne są dwie funkcje do siebie (*"jeśli kod A jest podobny w jakimś stopniu do kodu B, to kod B jest tak samo podobny do kodu A"*), doktorant zaproponował podejście niesymetryczne (bazujące na zawieraniu się jednej funkcji w drugiej), w którym metoda porównująca dwie funkcje zwraca parę wartości, które następnie są agregowane przy użyciu t-norm (minimum, produktowej i Łukasiewicza), średniej arytmetycznej lub t-konorm, aby uzyskać lepsze wyniki, niż przy użyciu jednej, symetrycznej wartości podobieństwa. Pokazano, że dzięki odpowiedniej agregacji wyników kilku metod można osiągnąć wyniki lepsze, niż w przypadku podejścia symetrycznego.

8. Stworzone oprogramowanie w postaci publicznie dostępnego pakietu R umieszczonego w repozytorium CRAN oraz serwisu internetowego, dostępnego pod adresem `http://similar.rexamine.com`, umożliwiającego porównywanie kodów źródłowych.

Przedstawiony zakres osiągnięć doktoranta można uznać za pokaźny. Warto również zaznaczyć, że wybrane wyniki badań doktorant opublikował w pięciu publikacjach, które opisuje w rozdziale 1.4 rozprawy. W bazie Scopus można znaleźć siedem publikacji doktoranta, które uzyskały w sumie 20 cytowań.

4 Uwagi

Po lekturze rozprawy doktorskiej nasuwają się oczywiście różne pytania, uwagi, czy komentarze. Nie mają one na celu podważenia uzyskanych wyników, świadczą raczej o tym, że szczegółowo udokumentowane wyniki badań skłaniają do dalszej dyskusji.

1. Spośród wymienionych wyżej kluczowych osiągnięć doktoranta ewentualną dyskusję może wzbudzić kwestia tego, czy doktorant faktycznie jako pierwszy zbadał, w kontekście detekcji klonów, skuteczność algorytmu opartego na metodzie Weisfeilera–Lehmana (w rozprawie pada stwierdzenie: *"W tej pracy zbadano po raz pierwszy w kontekście detekcji klonów skuteczność algorytmu o złożoności wielomianowej opartego na metodzie Weisfeilera–Lehmana"*). Wydaje się, że aby móc coś takiego stwierdzić i uniknąć ewentualnych wątpliwości należałoby pokusić się o przeprowadzenie systematycznego przeglądu literatury (ang. *Systematic Review* lub *Systematic Literature Review*). Jest on formą badania drugiego rzędu (ang. *secondary study*), czyli takiego, które poddaje przeglądowi wszystkie podstawowe badania/publikacje

naukowe (ang. *primary studies*) odnoszące się do konkretnych pytań badawczych w celu zebrania/integracji/syntezy wyników badań. Systematyczny przegląd wyróżnia to, że używa dobrze zdefiniowanej metodyki do identyfikacji, analizy i interpretacji wszystkich dostępnych źródeł informacji odnoszących się do konkretnych pytań badawczych w maksymalnie powtarzalny i bezstronny sposób. Systematyczne przeglądy (mające swoje źródła w medycynie i nurcie nazwanym *Evidence-Based Medicine*) przeniesione zostały na grunt inżynierii oprogramowania przez profesorów Barbarę Kitchenham, Tore Dybå i Magne Jørgensen [1, 2]. Szybko stały się one fundamentem jednego z najważniejszych nurtów inżynierii oprogramowania ostatnich lat, czyli *Evidence-Based Software Engineering*. Systematyczne przeglądy są publikowane przez najbardziej prestiżowe czasopisma inżynierii oprogramowania, takie jak *IEEE Transactions on Software Engineering, Information and Software Technology*, czy *Empirical Software Engineering*.

2. W swojej dysertacji doktorant pominął pracę Li i in. [3] pt. *Detecting Similar Programs via The Weisfeiler-Leman Graph Kernel*, która ukazała się w 2016 roku, co może być swego rodzaju konsekwencją braku systematycznego przeglądu literatury. W streszczeniu brakującej publikacji Li in. [3] na stronach Springera czytamy: *"In this paper, we propose an algorithm to compute program similarity based on the Weisfeiler-Leman graph kernel. Our algorithm is able to operate on different graph-based representations of programs and thus can be applied in different domains. We show the usefulness of our approach in two experiments using data-flow similarity and API-call similarity."* Praca ta wydaje się zatem zawierać pomysły, które można znaleźć w dysertacji: by nowy algorytm oceny podobieństwa programów bazował na algorytmie Weisfeiler-Lemana i mógł operować na różnych reprezentacjach grafowych programu. Na etapie pisania recenzji nie mogłem zapoznać się z pełnym tekstem wspomnianej pracy, ale zaprezentowane w pracy Li i in. pomysły mogą być podobne do rozwiązań prezentowanych w dysertacji. Zakładam, że doktorant mógłby odnieść się do tych kwestii podczas obrony.

3. Przyznam, że w rozdziale 1.2 zatytułowanym *"Hipotezy badawcze"* spodziewałem się hipotez, które stosując metody statystyczne (testowanie hipotez) można odrzucić lub nie, zamiast takich sformułowań, jak np. *"Możliwe jest stworzenie jasno zdefiniowanego podejścia do testowania algorytmów..."*.

4. Czwarty spośród wymienionych przez doktoranta (w rozdziale 1.3) oryginalnych wyników rozprawy to *"Stworzenie algorytmu SimilaR, który przy użyciu grafu zależności programu daje lepsze wyniki, niż dotychczas zaproponowane metody w sensie miary F"*. Osiągnięcie tego celu z wykorzystaniem miary F (ang. *F measure*) wydaje się być nieco dyskusyjnym ze względu na fakt, iż miara F nie bazuje na wszystkich czterech ćwiartkach macierzy konfuzji (ang. *confusion matrix*). Wielu naukowców (m.in. Powers czy Shepperd) wskazuje, że jest to wątpliwa miara oceny klasyfikatorów binarnych:

   - *"Many commonly used measures such as the F-measure (see Table 4) are unsatisfactory due to their not being based on the complete confusion matrix [16]"* [4]
   - *"A common but poorly motivated way of evaluating results of Machine Learning experiments is using Recall, Precision and F-measure. These measures are named for their origin in Information Retrieval and present specific biases, namely that they ignore performance in correctly handling negative examples, they propagate the underlying marginal prevalences and biases, and they fail to take account the chance level performance."* [5].

   W związku z tym dyskusja dotycząca właściwego doboru miar jakości modelu predykcji w problemach klasyfikacji binarnej o niezbalansowanej liczbie klas, zaprezentowana przez dokto-

ranta w rozdziale 3.4 zatytułowanym *"Miary jakości używane przy problemach niezbalansowanych"*, wydaje się, że zasługuje na bardziej wnikliwe potraktowanie, być może zaadresowanie przytoczonych wątpliwości związanych z zastosowaniem miary F. W tym kontekście warto zauważyć, że jedną z rekomendowanych miar, którą można zastąpić miarę F, jest MCC (ang. *Matthews Correlation Coefficient*). Takie rozwiązanie jest sugerowane m.in. przez Shepperda.

5. Praca zawiera empiryczne porównanie różnych algorytmów, z których każdy może wykorzystywać różne reprezentacje programu, a porównania mogą być przeprowadzane na różnych zbiorach benchmarkowych. Biorąc pod uwagę specyfikę pracy zabrakło mi w niej dyskusji nad zagrożeniami dla wiarygodności uzyskanych rezultatów empirycznych (ang. *threats to validity*) [6], zasadności użycia takich, a nie innych metod, czy testów statystycznych [7], sposobów zaadresowania związanych z tym problemów, które realnie wystąpiły, jak i omówienia tych, które zdaniem doktoranta stanowią niewielkie zagrożenie lub wręcz nie stanowią problemu. Przykładowo jednym z zagadnień może być problem porównań wielokrotnych (roz. 3) . Polega on na zwiększonym ponad nominalny poziom istotności ryzyku omyłkowego odrzucenia prawdziwej hipotezy zerowej (popełnienia błędu I rodzaju), przy wykonywaniu wielu porównań tej samej grupy hipotez jednocześnie. Przynajmniej jeden z testów może przypadkiem, dzięki losowej zmienności prób, przekroczyć próg istotności z prawdopodobieństwem równym $\overline{\alpha} = 1 - (1 - \alpha)^m$. Przy typowym, przyjętym również przez doktoranta, poziomie $\alpha = 0.05$, ryzyko popełnienia przynajmniej jednego błędu I rodzaju przekracza 50% już przy liczbie porównań wielokrotnych $m \approx 13$.

6. Zapoznając sie z rozwiązaniami prezentowanymi przez doktoranta niekiedy zadawałem sobie pytanie dlaczego pewne decyzje doktoranta są takie, a nie inne. Nie zawsze były one uzasadniane czy omawiane w kontekście innych alternatyw. Np. jeśli stosując medianę celem doktoranta było uzyskanie pewnej miary centralnej tendencji, czy lokacji (ang. measures of central tendency, measures of central location), których zadaniem jest opisanie zbioru danych za pomocą pojedynczej wartości reprezentującej środek/centrum rozkładu, to mamy do dyspozycji spektrum miar, w tym także tzw. odporne (ang. *robust measures*). Jedną z najbardziej znanych jest oczywiście użyta przez doktoranta mediana. Jakkolwiek jest ona z pewnością miarą odporną, to ma też pewne słabości. W istocie bowiem ignoruje ona wszystkie obserwacje za wyjątkiem jednej lub dwóch. Problematyczna jest też estymacja standardowego błędu mediany. Wilcox [8] szczegółowo omówił jednak inne odporne miary centralnej lokacji bazujące na usuwaniu obserwacji odstających (ang. *removing outliers*) np. średnia ucinana (ang. *trimmed mean*) czy średnia winsorowska (ang. *winsorized mean*), które są relatywnie odporne na wartości odstające, a nie ignorują niemal wszystkich obserwacji, jak ma to miejsce w przypadku mediany. Tego typu alternatywy warto rozważać. Mogą one też okazać się interesującym kierunkiem dalszych modyfikacji prezentowanych rozwiązań.

7. Na stronie 126 można przeczytać: *"Zaproponowaną metodę porównywaliśmy w przypadku klasyfikacji binarnej z następującymi modelami: 1) Naiwny Bayes [14, 114] (funkcja naiveBayes z pakietu e1071 w języku R), 2) drzewo decyzyjne [27] (funkcja rpart z pakietu rpart), 3) metoda k najbliższych sąsiadów [4] dla k = 5 (funkcja knn z pakietu class), 4) regresja logistyczna [35, 60, 96, 125] (funkcja glm z pakietu stats), 5) SVM [31, 37, 118] (funkcja svm z pakietu e1071), 6) lasy losowe [25, 26] (funkcja randomForest z pakietu randomForest)."*

   Przydałaby się bardziej szczegółowa informacja nt. parametrów modeli (np. SVM), chyba, że były stosowane parametry domyślne. Gdyby tak było to jednak warto to wyraźnie za-

znaczyć. Byłaby to też pewna słabość rozwiązań, z którymi proponowane rozwiązanie było porównywane.

8. Na stronie 126 można przeczytać: *"W przypadku wartości z przedziału [0, 1] porównywaliśmy nasze podejście z regresją liniową"*. Wydaje się, że można było pokusić się o użycie w porównaniu szerszej puli modeli regresji, które sprawują się często lepiej, niż regresja liniowa np. model MARS (ang. *Multivariate Adaptive Regression Splines*) będący rozszerzeniem modelu liniowego (koncepcyjnie przypuszczalnie bliski doktorantowi), w którym zmienna objaśniająca w zależności od swoich wartości może mieć różny wpływ na zmienną objaśnianą.

9. Na stronach 127-128 można przeczytać: *"Porównujemy zwykłą regresję liniową z naszą metodą w dwóch wersjach: optymalizowaną dla miary MAD lub RMSE"*. Wydaje się, że warto byłoby uzasadnić sytuację, w której proponowana metoda była optymalizowana pod kątem konkretnych miar, podczas gdy metoda z którą proponowana metoda była porównywana nie tylko nie była optymalizowana, ale i jakieś nowsze rozwiązanie (np. wspomniany MARS) nie było brane pod uwagę.

4.1 Uwagi drobne

1. Na stronie 135 doktorant zastosował metodę PCA (ang. *Principal Component Analysis*). Niestety metoda PCA ma pewne słabości, które warto omówić i ewentualnie rozważyć zastosowanie metody alternatywnej (taką rolę niekiedy pełni metoda PLS).

2. Omówienie rezultatów zawartych w Tabeli 3.3, prezentującej wartości miar jakości dla odległości Levenshteina, wydaje się pomijać najlepszy uzyskany z użyciem F miary rezultat dla funkcji agregującej S_p i sekwencji wejściowej w postaci tokenów: *"Gdybyśmy więc chcieli używać tej metody jako jedynej, prawdopodobnie najlepiej byłoby uruchamiać ją dla tokenów, używając M lub $T_{\text{Ł}}$ jako funkcji agregującej. Miara F jest w tym wypadku na najwyższym poziomie, podczas gdy ta sekwencja wejściowa zajmuje drugie miejsce pod względem czasu działania."*

3. Na stronie 75 można przeczytać: *"Według naszej najlepszej wiedzy nie ma żadnej pracy, która wykorzystywałaby algorytm Smitha–Watermana w kontekście wykrywania podobnych kodów źródłowych."*. Sensownym sposobem na poparcie tej tezy byłoby przeprowadzenie systematycznego przeglądu literatury, w którym jedno z postawionych pytań badawczych dotyczyłoby algorytmów wykorzystywanych w kontekście wykrywania podobnych kodów źródłowych.

4. Pojawia się pytanie natury bardziej ogólnej, czy proponowane rozwiązania dostosowane do specyfiki języka R okazałyby się skuteczne również w innych (i jakich) językach.

5 Podsumowanie

Przedstawione przeze mnie niewątpliwe osiągnięcia doktoranta, nawet w świetle zgłoszonych uwag, pozwalają stwierdzić, że mgr inż. Maciej Bartoszuk wykazał się dużą wiedzą z zakresu metod oceny podobieństwa kodów źródłowych, zarówno algorytmów jak i sposobów reprezentacji programów, a także opanowaniem i właściwym posługiwaniem się warsztatem naukowym. Zadbał też o to, by wyniki jego prac były łatwo dostępne dla potencjalnych użytkowników w postaci pakietu w języku R oraz serwisu internetowego. Wydaje się to bardzo cenne i dobrze wróży na przyszłość.

Uważam, że przedłożona do oceny praca doktorska Pana mgra inż. Maciej Bartoszuka zawiera oryginalne i kompleksowe rozwiązanie ważnego i aktualnego problemu badawczego, pokazuje wysoką, ogólną wiedzę teoretyczną i zdolności praktyczne kandydata oraz umiejętność prowadzenia prac naukowych, a tym samym spełnia wymogi art. 13 ust. 1 ustawy z dnia 14.03.2003 r. z późniejszymi zmianami. Po spełnieniu warunków formalnych wnoszę o dopuszczenie jej do publicznej obrony.

Bibliografia

[1] Barbara A. Kitchenham, Tore Dybå i Magne Jørgensen. "Evidence-Based Software Engineering". W: *ICSE'04: International Conference on Software Engineering*. 2004, s. 273–281.

[2] Barbara Kitchenham, David Budgen i Pearl Brereton. *Evidence-Based Software Engineering and Systematic Reviews*. CRC Press, 2016.

[3] Wenchao Li, Hassen Saidi, Huascar Sanchez, Martin Schäf i Pascal Schweitzer. "Detecting Similar Programs via The Weisfeiler-Leman Graph Kernel". W: *Software Reuse: Bridging with Social-Awareness*. Red. Georgia M. Kapitsaki i Eduardo Santana de Almeida. Cham: Springer International Publishing, 2016, s. 315–330. ISBN: 978-3-319-35122-3.

[4] Martin Shepperd, David Bowes i Tracy Hall. "Researcher Bias: The Use of Machine Learning in Software Defect Prediction". W: *IEEE Transactions in Software Engineering* 40.6 (2014), s. 603–616. DOI: 10.1109/TSE.2014.2322358.

[5] David M W Powers. "EVALUATION: FROM PRECISION, RECALL AND F-MEASURE TO ROC, INFORMEDNESS, MARKEDNESS & CORRELATION". W: *Journal of Machine Learning Technologies* 2.1 (2011), s. 37–63.

[6] Claes Wohlin, Per Runeson, Martin Höst, Magnus C. Ohlsson, Björn Regnell i Anders Wesslén. *Experimentation in Software Engineering: An Introduction*. 2nd. Berlin Heidelberg, Germany: Springer, 2012. ISBN: 978-3-642-29043-5.

[7] Barbara Kitchenham, Lech Madeyski, David Budgen, Jacky Keung, Pearl Brereton, Stuart Charters, Shirley Gibbs i Amnart Pohthong. "Robust Statistical Methods for Empirical Software Engineering". W: *Empirical Software Engineering* 22.2 (2017), s. 579–630. DOI: 10.1007/s10664-016-9437-5. URL: http://link.springer.com/content/pdf/10.1007%2Fs10664-016-9437-5.pdf.

[8] Rand R. Wilcox. *Modern Statistics for the Social and Behavioral Sciences: A Practical Introduction*. CRC Press, 2012.

Lech Madeyski

.....

Lech Madeyski