Systemy uczące się – studium problemów

Marek Kozłowski[1]

[1] Politechnika Warszawska, Wydział Elektroniki i Technik Informacyjnych,
00-665 Warszawa, Polska
{Marek.Kozlowski@stud.elka.pw.edu.pl}

Abstract.

Artykuł dotyczy przekrojowego ujęcia tematyki uczenia się maszyn. Praca jest studium problemów, jakie pojawiły się na obecnym etapie rozwoju systemów uczących. Maszynowe uczenie to dziedzina związana z poszukiwaniem odpowiedzi na pytanie jak konstruować systemy, które automatycznie doskonalą swoje działanie poprzez analizę doświadczenia i nabywanie wiedzy na tej podstawie. Maszynowe uczenie jest z natury dziedziną silnie interdyscyplinarną opartą na takich gałęziach jak: probabilistyka, teoria sterowania, filozofia, biologia i psychologia. Płynny rozwój maszynowego uczenia wymaga rozwiązania pewnych problemów teoretycznych i praktycznych. Problemy teoretyczne są związane z budowaniem abstrakcyjnego modelu systemów uczących, taksonomii. Rodzaje systemów uczących się można klasyfikować według klasycznych kryteriów: metoda reprezentacji wiedzy, sposób używania wiedzy, źródło i postać informacji trenującej, mechanizm nabywania wiedzy. Każdemu z tych kryteriów poświęcimy szczególną uwagę zarysowując wynikający z niego podział. Postaram się też przedstawić nowatorskie kryteria klasyfikacji systemów oparte o dziedziny pokrewne i zastosowania. Zadania praktyczne, wymagające rozwiązania w *machine learning*, dotyczą głównie optymalizacji dokładnościowo – szybkościowej procesu uczenia. Na przykładzie systemów wykrywania włamań przedstawię potencjalne sposoby ulepszania działania systemów uczących. Narzędziami, jakie do tego wykorzystam, będą wstępne przetwarzanie danych – selekcja atrybutów i budowanie systemów hybrydowych. Testy jakościowe systemów uczących przeprowadziłem przy wykorzystaniu algorytmów: *ID3*, *Naive Bayes*, *Bayesian Net* oraz zbioru danych KDD Cup 99.

Keywords: systemy uczące się, machine learning, uczenie się maszyn, EBL, indukcja, dedukcja, intrusion detection systems, preprocessing, feature selection, KDD Cup 99

1. Wprowadzenie

Uczenie się maszyn jest dziedziną związaną z poszukiwaniem odpowiedzi na pytanie jak konstruować systemy, które automatycznie doskonalą swoje działanie poprzez analizę doświadczenia. Naszym celem jest tworzenie takich systemów, które będą stale się ulepszały bez zbytniej ingerencji człowieka.

Prace nad maszynami uczącymi się sięgają lat 50-60 poprzedniego wieku. Najbardziej spektakularnym sukcesem badawczym maszynowego uczenia się stał się w tym czasie opracowany przez Samuela[1] program uczący się gry w warcaby, który doskonaląc swoją strategie na podstawie rozgrywanych partii, osiągnął poziom bardzo dobrego gracza-człowieka. Od tego momentu badania nad systemami uczącymi rozrosły się tematycznie, uległy szerokiemu zróżnicowaniu i przede wszystkim zdobyły dużą popularność. W ten sposób wytworzyła się prężna dyscyplina naukowa. Tę naukę o uczeniu się sztucznych systemów nazywa się czasem maszynowym uczeniem lub uczeniem się maszyn (*machine learning*) – terminów tych używam wymiennie. Najwygodniej jest traktować ją jako gałąź sztucznej inteligencji, chociaż wraz z pogłębianiem się specjalizacji we współczesnej nauce wzrasta także znaczenie badań interdyscyplinarnych i coraz trudniej o prosty hierarchicznie podział. Istotny wkład do badań nad systemami uczącymi się wnoszą takie dziedziny jak m.in. psychologia, statystyka, automatyka, biologia.

W ten sposób stajemy przed dziedziną w pełni ukształtowaną, posiadająca wiele różnorodnych tematycznych obszarów oraz powiązaną z innymi dziedzinami nauki. Jak każda prężnie rozwijająca się nauka tak i maszynowe uczenie posiada wiele problemów i zadań, których rozwiązanie jest konieczne do dalszego rozwoju badań. Powyższe problemy można podzielić na teoretyczne, związane z taksonomią, tworzeniem pewnego abstrakcyjnego modelu *machine learning* oraz na problemy praktyczne koncentrujące się na wydajności i skuteczności.

Problemy teoretyczne systemów uczących dotyczą budowania prawidłowego i zupełnego modelu dziedziny. Niesamowita różnorodność systemów uczących się wymaga wprowadzenia pewnego porządku, który pozwoli nam lepiej zorientować się co jest esencją systemów, gdzie są granice obszarów tej dziedziny, jakie i gdzie są granice całej dziedziny. W tym celu wprowadzę kryteria porównawcze, które dokonują pewnej systematyzacji. Zacznę od opisu klasycznych kryteriów takich jak metoda reprezentacji wiedzy, sposób wykorzystania wiedzy, techniki nabywania wiedzy czy postać informacji trenującej. Następnie spróbuje wprowadzić trochę bardziej subtelne wyznaczniki klasyfikujące systemy uczące się. Wykorzystam interdyscyplinarne dziedziny nauki do pogrupowania i porównania różnych rodzajów uczenia się. Wprowadzę też kryterium stosowalności praktycznej systemów uczących się. Jest to jedno z najbardziej ciekawych podejść do uczenia się maszyn polegające na pokazaniu grup praktycznych problemów, do których rozwiązania stosowane są konkretne klasy systemów. Poruszając ten temat postaram się też pokazać

ograniczenia i sposoby dostrajania technik uczenia się do rozwiązywania określonych klas problemów.

Problemy praktyczne maszynowego uczenia ogniskują się wokół kryteriów jakościowych i ilościowych, takich jak dokładność działania i szybkość. Wykorzystywanie algorytmów uczenia na coraz bardziej newralgicznych polach informatyki(systemy bezpieczeństwa) wymaga od nich wysokiej skuteczności – dokładności wykrywania znanych i nieznanych nadużyć. Z drugiej strony przed systemami stawiamy ograniczenia szybkościowe. Dążymy do optymalizacji procesu trenowania jak i skracania czasu samego etapu stosowania zgromadzonej wiedzy. Dzięki temu sterowanie obiektami zyskuje na większej precyzji i płynności, a wykrywanie włamań pozwala szybciej przeciwdziałać nadużyciom. Drogą do usprawnienia funkcjonowania systemów uczących jest zastosowanie preprocessingu – odpowiedniego przetworzenia danych, oraz tworzenie rozwiązań hybrydowych, jako remedium na cząstkowe wady pojedynczych algorytmów lub technik uczenia.

Najpierw jednak zanim przejdziemy do dokładnej analizy powyższych problemów postaram się w rozdziale 2 przedstawić intuicyjne wyobrażenie o uczeniu się systemów. Pozwoli to w dalszej części pracy lepiej zrozumieć i głębiej spojrzeć na tą dziedzinę nauki. W kolejnych rozdziałach 3 i 4 skupiam się na problemach teoretycznych *machine learning*. Rozdział 5 jest poświęcony analizie problemów praktycznych, skupionych wokół dokładności i szybkości działania systemów uczących. Ostatni szósty rozdział postanowiłem potraktować jako podsumowanie artykułu i wskazać w nim ciekawe drogi przyszłościowych badań w maszynowym uczeniu.

2. Istota systemów uczących się

Analiza systemów uczących się wymaga najpierw dokładnego zrozumienia co kryje się pod hasłem „system uczący się”. Słowo system jest ogólnie znane i mogą się pod nim kryć zarówno organizmy żywe jak i roboty, programy itd. Natomiast co dokładnie oznacza słowo uczenie się jest o wiele trudniej sprecyzować. Wszyscy od najmłodszych lat uczymy się mówić, czytać, jeździć na rowerze. Uczymy się szacować nastrój ludzi na podstawie gestykulacji. Podejmujemy niezliczone ilości prób i popełniamy błędy korygowane przez nauczycieli. Obserwując otoczenie uogólniamy je i odnajdujemy różnego rodzaju zależności. A więc całe nasze doświadczenie jest nasycone uczeniem. Wszystkie nasze cechy intelektualne są wynikiem niezliczonej ilości przenikających się procesów uczenia.

W celu zdefiniowania uczenia się wskażę cztery podstawowe cechy tego procesu. Po pierwsze uczymy się tylko wtedy, gdy następuje zmiana. Po drugie musi to być zmiana prowadząca do poprawy wykonywania jakiejś czynności, w końcu nauka ma prowadzić do doskonalenia. Jednak te dwie cechy to za mało, bo przecież wymiana mikroprocesora w robocie na szybszy nie jest na pewno uczeniem, a następuje tu zarówno zmiana jak i pewna poprawa działania. Kolejna cecha uczenia to autonomiczność – system sam musi zmieniać się na lepsze. Natomiast na koniec musimy jeszcze określić co jest podstawą do uczenia się. Tą podstawą do zmian jest zdobyte doświadczenie. Dysponując tymi czterema elementami: zmiana, poprawa, autonomiczność i doświadczenie możemy sformułować intuicyjną definicję uczenia

się systemów. System uczący się to taki system, w którym na podstawie doświadczeń zachodzą autonomiczne zmiany prowadzące do poprawy jakości jego działania. Definicja powyższa jest nieostra, co raczej nie jest wynikiem złego sformułowania, ale wynikiem naturalnej nieostrości pojęcia uczenia. Najważniejsze jednak, że pozwala zwrócić uwagę na cztery cechy, które w sposób zupełny i intuicyjnie opisują proces uczenia się, w tym i uczenie sie maszyn.

Zbudowaliśmy definicje uczenia się ogólnych systemów, teraz postaram sie wytworzyć pewien też intuicyjny opis uczenia sie systemów komputerowych, czyli uczenia sie programów. Program uczący się to program wykorzystujący pewien abstrakcyjny, sparametryzowany algorytm wykonania zadania. Uczenie się polega na dobraniu na bazie doświadczeń parametrów, tak aby powstał konkretny algorytm wykonania zadania. Uzyskiwane w ten sposób parametry są nazywane wiedzą lub umiejętnościami[8].

3. Kryteria porównawcze systemów uczących się

Problemy teoretyczne systemów uczących dotyczą utworzenia abstrakcyjnego, zupełnego opisu dziedziny *machine learning*. W tym celu musimy najpierw określić, jakie są charakterystyczne grupy systemów uczących i jakie cechy odróżniają je od siebie.

Rodzaje systemów uczących się można porównywać i klasyfikować według szeregu kryteriów, uzyskując różne mniej lub bardziej ciekawe podziały. Opis zacznę od najbardziej klasycznych kryteriów, aby później pokazać własne pomysły na porównywanie systemów.

Najbardziej podstawowe kryteria porównawcze to:

1. Metoda reprezentacji wiedzy
2. Technika nabywania wiedzy
3. Źródło i postać informacji trenującej
4. Sposób wykorzystania wiedzy

3.1 Metoda reprezentacji wiedzy

Z reguły dziedzina zastosowań systemu uczącego dość znacznie zawęża wybór reprezentacji wiedzy. Najczęściej z danym algorytmem uczenia związana jest pewna grupa reprezentacji wiedzy i nasz wybór co do konkretnej realizacji zależy od ograniczeń jakie przed nami stawia praktyczny problem.

Najczęściej wykorzystywane metody reprezentacji wiedzy to: drzewa decyzyjne (dziedzina algorytmów ID3), reguły (indukcja reguł), formuły logiki predykatów (indukcyjne programowanie logiczne), rozkłady prawdopodobieństw (metody probabilistyczne oparte o sieci bayesowskie), automaty skończone (algorytm L*).

Do tradycyjnych podziałów metod reprezentacji wiedzy należy podział na reprezentację symboliczne i subsymboliczne. Reprezentacje symboliczne przechowują informację w postaci zorganizowanych w pewien sposób napisów, którym można przypisać interpretację. Tak przechowywana wiedza może być dość łatwo zapisana w czytelnej dla człowieka postaci. Natomiast reprezentacje

subsymboliczne charakteryzują się tym, iż poszczególne elementy przechowywanej informacji rozpatrywane z osobna nie mają sensownej interpretacji. Są to przede wszystkim zbiory liczb (wagi w sieciach neuronowych) lub ciągi binarne (rodzina algorytmów genetycznych). Takie reprezentacje łącznie niosą ze sobą pewną wiedzę, ale wiedza ta nie może być bezpośrednio wyrażona w postaci zrozumiałej dla człowieka. Jako dobry przykład subsymboliczności mogę podać matryce zegarka elektronicznego. Składa się ona z małych lampek diodowych, które działają w sposób zero-jedynkowy (zapalone lub niezapalone). Łącznie wszystkie te lampki reprezentują określoną godzinę, ale analiza pojedynczego stanu lampki nie niesie żadnej informacji, jest nieinterpretowalna.

3.2 Technika nabywania wiedzy

Na podstawie otrzymanej informacji trenującej system generuje nową lub doskonali posiadaną wiedzę. Mechanizm, zgodnie z którym dokonuje się nabywania wiedzy, jest najczęściej wyznaczony przez metodę reprezentacji wiedzy oraz postać informacji trenującej.

Najbardziej popularne mechanizmy uczenia się to: indukcja, dedukcja, EBL(uczenie przez wyjaśnianie), uczenie przez analogię.

Indukcja polega na uogólnianiu jednostkowej informacji trenującej w celu uzyskania ogólnej wiedzy. Istnieje cała rodzina mechanizmów indukcyjnych, wśród których możemy wyróżnić indukcje drzew decyzyjnych (ID3), indukcje reguł, indukcyjne programowanie logiczne, a nawet podstawy indukcyjne wykorzystują sieci bayesowskie. Można z dużą pewnością stwierdzić, iż indukcyjne nabywanie wiedzy jest najliczniej reprezentowane wśród systemów uczących się.

Dedukcja polega na dojściu do określonego wniosku na podstawie wcześniej ustalonego zbioru prawdziwych przesłanek. Najczęściej zapisywana w postaci zbioru reguł, lub predykatów. W czystej postaci wykorzystywana na szeroką skalę w systemach eksperckich. Dysponują one ustaloną bazą wiedzy (zbiór prawdziwych przesłanek) oraz ściśle określonymi regułami, na podstawie, których za pomocą wnioskowania wprzód/wstecz są wyprowadzane nowe fakty.

EBL(*explanation based learning*) – uczenie przez wyjaśnianie polega na wyjaśnianiu przez system przykładów trenujących, czyli wyprowadzanie ich w drodze logicznej dedukcji z dostępnej jako wiedza wrodzona teorii dziedziny. Odpowiednio uogólnione wyjaśnienia wyznaczają hipotezę ucznia. Uczenie przez wyjaśniane jest określane, przez niektórych autorów, jako kompilacja wiedzy wrodzonej ukierunkowana przykładami trenującymi[4]. Uważam, iż EBL jest jednym z ciekawszych i rzadko poruszanych algorytmów uczenia, dlatego postaram się dokładniej przedstawić operacje jakie w nim występują. Działanie uczenia przez wyjaśnianie można rozłożyć na trzy etapy[5]:

- Wyjaśnianie – wyprowadzenie przykładu trenującego z teorii dziedziny
- Analiza – analiza wyjaśnienia w celu określenia najbardziej ogólnych warunków, przy których dalej jest spełnione to docelowe pojęcie
- Redefinicja – zmiana hipotezy przez dodanie nowej reguły, której poprzednikami są powyżej określone warunki, a następnik zgodny z docelowym pojęciem

Uczenie przez analogię (*analogical reasoning*) polega na założeniu, iż jeśli dwie sytuacje są podobne w pewnych aspektach to z dużym prawdopodobieństwem są podobne też w innych aspektach. Na przykład, jeśli dwa domy mają podobną lokalizację, powierzchnie i jakość wykończenia to z dużym prawdopodobieństwem mają podobne ceny sprzedaży[6]. EBL jest w pewnym stopniu ograniczony przez wnioskowanie dedukcyjne, podczas gdy uczenie przez analogię oferuje bardziej elastyczne użycie istniejącej wiedzy. Standardowy model uczenia przez analogię definiuje źródło jako przykład lub teorię, która jest dobrze rozumiana. Cel nie jest całkowicie zrozumiały. Analogia dokonuje przemapowania pomiędzy odpowiednimi elementami źródła i celu. Kolejnym krokiem jest rozszerzenie tego mapowania na nowe elementy docelowej dziedziny. Badając analogię między wodą a elektrycznością, jeśli wiemy, że ta analogia mapuje natężenie na wielkość przepływu, napięcie na ciśnienie, możemy podejrzewać, iż musi być jakiś odpowiednik dla przepustowości rury. To może doprowadzić nas do zrozumienia pojęcia elektrycznej oporności.

3.3 Źródło i postać informacji trenującej

Klasyczny podział ze względu na informacje trenującą wyróżnia uczenie z nadzorem i uczenie bez nadzoru. W przypadku uczenia z nadzorem system otrzymuje informację określającą w pewien sposób jego pożądane odpowiedzi dla pewnego zbioru danych wejściowych. Są one przykładami zachowania, jakiego się od niego oczekuje. Przy uczeniu bez nadzoru taka instruktażowa informacja trenująca w ogóle nie jest dostępna. Podawane są jedynie dane wejściowe i system ma się nauczyć właściwych odpowiedzi wyłącznie obserwując ich sekwencje. Z tego rodzaju uczeniem mamy do czynienia głównie przy grupowaniu lub innych przekształceniach przestrzeni wejściowej. Istnieją jednak jeszcze inne rodzaje uczenia, które trudno jednoznacznie zakwalifikować do jednej z powyższych grup, a o których warto wspomnieć.

Bardzo popularnym ostatnio jest uczenie ze wzmocnieniem. Polega ono na tym, iż system wykonuje odpowiednie akcje, które są ocenianie w sposób wartościujący. Informacja trenująca nie ma w tym przypadku charakteru instruktażowego, ale przekazuje informacje jak dobre lub złe są dotychczasowe działania. Najlepszym edukacyjnie przykładem takich algorytmów są metody TD uczenia się funkcji wartości.

Uczenie się na podstawie zapytań polega na zadawaniu przez system konkretnych pytań nauczycielowi. Rola nauczyciela sprowadza się do odpowiadania na konkretne pytania, a nie dobierania przykładów.

O uczeniu przez eksperymentowanie mówimy, gdy uczeń gromadzi doświadczenia eksperymentując z otaczającym go środowiskiem. Polega to na wykonywaniu pewnych akcji i obserwowaniu ich konsekwencji.

3.4 Sposób wykorzystania wiedzy

Sposób wykorzystania wiedzy jest z reguły jednoznacznie określony przez metodę reprezentacji wiedzy i cel, czyli stojące przed systemem zadanie. W przypadku indukcyjnych systemów uczących możemy wyróżnić trzy główne zadania:

- *Klasyfikacja* – ustalenie przynależności obiektu do kategorii
- *Grupowanie* – samodzielne tworzenie kategorii w oparciu o podobieństwo
- *Aproksymacja* - odwzorowanie obiektów na zbiór liczb rzeczywistych

Do mniej typowych zadań, których rozwiązania podejmują się systemy uczące należą:

- *Usprawnianie rozwiązywania problemów*(*uczenie się heurystyk*) – można to nazwać też usprawnieniem przeszukiwania przestrzeni stanów w celu znalezienia docelowych stanów. Mają tu szczególne zastosowanie algorytmy EBL. Najczęściej znana jest w tego rodzaju zadaniach pełna i poprawna teoria dziedziny. Sprowadza się ona do definicji skutków wykonania poszczególnych operatorów oraz właściwości stanów docelowych. Zastosowanie EBL ma na celu nauczenie się przyspieszania znajdowania rozwiązań za pomocą makrooperatorów[4]. Makrooperatory są sekwencjami operatorów prowadzących pośrednio do stanów docelowych.
- *Sekwencyjne podejmowanie decyzji* – tutaj zastosowanie mają systemy uczące sie z opóźnionym wzmocnieniem. W celu nauczenia systemu stosuje się odpowiednie nagrody dążąc jednocześnie do maksymalizacji oczekiwanej zdyskontowanej sumy nagród.
- *Modelowanie środowiska* – wykorzystywane są do tego zadania algorytmy uczenia się automatów skończonych.
- *Przedstawienie zebranej wiedzy w czytelny dla użytkownika sposób.*

4. Propozycje nowych kryteriów porównawczych dla systemów uczących się

Przedstawione powyżej kryteria są klasycznymi miarami systemów uczących się, które pojawiają się w wielu pozycjach literatury. Chciałbym przedstawić także inne cechy, które także mogłyby posłużyć za kryteria porównawcze. Są nimi dziedziny pokrewne i zastosowania.

Wcześniej już pisałem, iż *machine learning* jest nauką interdyscyplinarną i korzysta z wyników, inspiracji innych dyscyplin naukowych. Te właśnie dyscypliny nazwałem dziedzinami pokrewnymi. Jest to całkiem pokaźny zestaw nauk pozwalający na znalezienie w nim dla każdego systemu odpowiednich odwołań. Zupełność mapowania systemów uczących się na dziedziny pokrewne przekonała mnie do możliwości zastosowania ich jako nowego kryterium. Przedstawię teraz kilka przykładowych dyscyplin i powiązanych z nimi systemów uczących.

Logika (reguły i predykaty) to dział matematyki, który wyodrębnił się na przełomie XIX i XX wieku. Koncentruje się na analizowaniu zasad rozumowania

oraz pojęć z nim powiązanych z wykorzystaniem sformalizowanych metod. Wykorzystujemy logikę do tworzenia systemów algebraicznych, dzięki którym zamiast słownych określeń stosujemy symbole zdań oraz dokonujemy obliczeń symbolicznych. W systemach uczących się logika jest podstawą wielu symbolicznych metod reprezentacji wiedzy. Dotyczy to przede wszystkim indukcyjnego programowania logicznego, gdzie generowana wiedza jest reprezentowana w postaci formuł logiki predykatów. Silne związki z logiką występują w indukcji reguł oraz w EBL, gdzie stosowane są elementy logicznej dedukcji.

Probabilistyka to dział matematyki zajmujący się zdarzeniami losowymi. Zajmuje się badaniem abstrakcyjnych pojęć stworzonych do opisu zjawisk, które nie są deterministyczne. Jej wykorzystanie w badaniu systemów uczących ma zarówno aspekt teoretyczny(analiza jakościowa algorytmów uczenia) jak i typowo praktyczny. Mechanizmy wnioskowania probabilistycznego są podstawą sieci bayesowskich, które ostatnimi czasy zdobywają coraz większą popularność.

Teoria sterowania to dział nauki i techniki zajmującej się zachowaniem układów dynamicznych w czasie. Jej głównym zadaniem jest budowanie systemów, w których regulator będzie manipulował wejściem układu tak, aby jego wyjście zachowywało się w pożądany sposób. Zastosowanie w tworzeniu modelów sterowania mają aproksymatory funkcji (odmiana indukcyjnego uczenia). Do tworzenia strategii sterowania w środowiskach o dużej zmienności są także wykorzystywane techniki uczenia ze wzmocnieniem.

Psychologia to nauka zajmująca się miedzy innymi fenomenem uczenia się organizmów żywych. We współczesnym uczeniu się maszyn korzenie psychologiczne ma uczenie ze wzmocnieniem. Wartościowanie w postaci liczbowych nagród przypomina podejście psychologów w ich badaniach nad uczeniem się zwierząt.

Neurofizjologia to nauka zajmująca się funkcjonowaniem układu nerwowego. Szczególna jej poddziedzina – neurofizjologia komórek zajmuje się funkcjonowaniem poszczególnych neuronów (komórek nerwowych). Ta dyscyplina jest podstawą sieci neuronowych, oraz przejawia się też w pewnych szczególnych odmianach aproksymacji funkcji.

Wszystkie systemy uczące są tworzone i wykorzystywane w celu rozwiązywania określonych praktycznych problemów. Z reguły każda rodzina systemów charakteryzuje się rozwiązywaniem specyficznej grupy problemów. Możemy stosować stosowalność praktyczną systemów jako kryterium porównawcze. Przedstawię dla przykładu wybrane problemy oraz związane z nimi algorytmy uczenia.

Odkrywanie wiedzy w bazach danych (*data mining*) jest jednym z najpopularniejszych aspektów badań naukowych i korporacyjnych ostatnich lat. Stosuje się w nich głównie indukcyjne algorytmy wzbogacane o różne narzędzia statystyczne. Za wykorzystywaniem indukcji przemawiają olbrzymie zbiory informacji trenującej oraz zadania: klasyfikacji i grupowania.

Problemy automatycznego sterowania mają szerokie zastosowanie w robotyce przemysłowej, biurowej jak i budowaniu bezzałogowych pojazdów. Tutaj szczególnie często korzysta się z aproksymatorów funkcji. Mogą one służyć wyznaczaniu modeli sterowanego obiektu, jak i służyć budowaniu strategii sterowania (oddziaływanie na obiekt zależne od stanu). Stany jak i sterowanie są reprezentowane przez wektory liczb rzeczywistych, co pozwala skutecznie działać aproksymatorom funkcji.

Optymalizacja rozwiązywania złożonych problemów dotyczy tak naprawdę nauki heurystyk przeszukiwania przestrzeni stanów. Ma to zastosowanie w usprawnieniu wykonywania złożonych operacji przez inteligentne roboty. Tutaj szerokie zastosowanie ma uczenie przez wyjaśnianie. Opisywaliśmy już to powyżej, chodzi o znajdowanie drogi na skróty poprzez przestrzeń stanów, za pomocą odpowiednich splotów elementarnych operacji.

Sterowanie robotami w zmiennych środowiskach i bez ingerencji człowieka jest jedną z częściej ostatnio poruszanych kwestii. Związane jest to z dążeniami do utworzenia robotów biurowych, które same nauczą się środowiska pracy i będą funkcjonowały w nim roznosząc dokumenty itp. Do rozwiązywania takich problemów stosuje się uczenie ze wzmocnieniem. Pozwala ono na naukę adaptacyjną otoczenia przez robota za pomocą eksperymentowania.

Wykrywanie włamań sprowadza się do wykrywania działań takich jak: włamania w sieciach wewnętrznych, nieupoważnione operacje na rachunkach bankowych, rozmowy na cudzy rachunek. Do rozwiązywania tego rodzaju problemów można by zastosować czystą indukcję do realizacji klasyfikowania, ale wyniki praktyczne takiego użycia są słabe[7]. Dlatego najlepszym rozwiązaniem jest połączenie sił różnych algorytmów i utworzenie hybrydowego rozwiązania. O tego typu systemach nie mówiliśmy wcześniej, ale są to pomysły bardzo perspektywiczne i dopiero od niedawna intensywnie rozwijane[15].

5. Problemy praktyczne *machine learning*

Do przedstawienia praktycznych problemów występujących w maszynowym uczeniu wykorzystam dziedzinę wykrywania włamań w systemach informatycznych. Postaram się przedstawić ogólny opis systemów wykrywania włamań (*Intrusion Detection Systems*), ich podstawowy podział oraz charakterystykę. Następnie w oparciu o zbiór danych KDD Cup 99 dokonam dokładnej analizy aspektów dokładnościowych i szybkościowych identyfikacji nadużyć na przykładzie wybranych algorytmów uczenia

5.1 Systemy wykrywania włamań

Systemy wykrywania włamań (*IDS - **I**ntrusion **D**etection **S**ystems*) to mechanizm nadzorowania bezpieczeństwa pozwalający na wykrywanie nieautoryzowanych dostępów do systemów lub sieci. IDS jest zdolny do wykrywania wszystkich typów wrogiego ruchu sieciowego i użycia komputerów.
Podstawowe przykłady wykrywanych nadużyć to:

- Network-based attacks - ataki sieciowe na podatne usługi
- Data-driven attacks – wirusy zakodowanie w niewinnie wyglądających danych
- Host-based attacks – nieautoryzowane logowania, eskalacja uprawnień
- Malware – złośliwe oprogramowanie w skład, którego zalicza się trojany, dialery, backdoory i wiele innych.

System wykrywania włamań sieciowych jest dynamicznie monitorującą jednostką, która uzupełnia statyczne właściwości ochronne firewalla. Gromadzone przez system pakiety sieciowe są analizowane pod kątem występowania ataku. Identyfikacja określonego połączenia jako nadużycia najczęściej sprowadza się do włączenia alarmu. W przypadku systemów hostowych mamy do czynienie z analizowaniem sekwencji komend lub wywołań programów na danej maszynie. IDS stosują różne metody analizy zbieranych danych, co doprowadziło do wykrystalizowania się specyficznych grup systemów wykrywania nadużyć.
Systemów wykrywania włamań charakteryzują się podziałem na 4 grupy:

1. *Misuse detection* – wykrywanie w oparciu o sygnatury
2. *Anomaly detection* – wykrywanie anomalii
3. *Compound detection* – rozwiązanie hybrydowe
4. *Ontology detection* – wykrywanie poprzez konstrukcje ontologii

Misuse detection systems. Systemy, których działanie oparte jest na predefiniowanym zbiorze sygnatur ataków. Przeglądając wyspecyfikowane wzorce nadużyć systemy próbują dopasowywać przychodzące pakiety lub sekwencje komend do sygnatur znanych ataków. Decyzje są podejmowane w oparciu o wiedzę zawartą w modelu na temat wrogich procesów (wiedza zawarta w sygnaturach) oraz na podstawie śladów aktywności w systemie. Złośliwe zachowania są definiowane w postaci odpowiednich zbiorów reguł lub diagramów przejść, a następnie konfrontowane z obserwowanym zachowaniem. Główną zaletą tej metody wykrywania włamań jest duża skuteczność identyfikacji znanych ataków. Kolejną zaletą jest brak etapu uczenia, czyli system oparty na sygnaturach rozpoczyna ochronę od razu po zainstalowaniu. Natomiast do kluczowych wad zaliczamy trudności z tworzeniem sygnatur dla wrogiej działalności obejmującej wiele rozciągniętych w czasie dyskretnych zdarzeń. Do słabości *misuse detection* należy także zdolność do wykrywania jedynie zdefiniowanych ataków, co wymaga stałego uaktualniania bazy wzorców nadużyć.
Anomaly detection systems. Systemy tego rodzaju najpierw tworzą bazowy profil normalnej sieciowej lub systemowej aktywności. Następnie każde zachowanie odbiegające od normalności jest traktowane jako prawdopodobny atak. Anomaly detection oferuje kilka znaczących korzyści. Po pierwsze posiada zdolność do wykrywania wewnętrznych włamań. Jeśli użytkownik lub osoba używająca skradzionego konta zacznie wykonywać akcje, które odstają od profilu normalności zostanie wygenerowany alarm. Funkcjonowanie systemu jest oparte na spersonalizowanych profilach, co oznacza, iż włamywacz nie wie jaką aktywność może podjąć bez ryzyka alarmu. Trzecią i najważniejszą zaletą *anomaly detection* jest zdolność do wykrywania dotąd nieznanych ataków. Wrogie działanie generuje alarm, ponieważ odbiega ono od normalnego profilu zachowania, a nie dlatego iż administrator skonfigurował system do poszukiwania specyficznych sygnatur nadużyć. Systemy wykrywania anomalii charakteryzują się też pewnymi wadami. Muszą przechodzić przez fazę uczenia (treningu), aby skonstruować profil normalności. Zbudowanie nie do końca pełnego profilu prowadzi do słabej jakości detekcji włamań. Natomiast uaktualnianie profilu jest czasochłonne. Systemy,

poprzez skoncentrowanie na poszukiwanie odchyleń od normalnych zachowań, są podatne na *false alarms.* Błędne alarmy dzielą się na *false positive* i *false negative*. *False positive* występuje gdy system alarmuje o wykryciu włamania podczas gdy obserwowana aktywność nie jest nadużyciem. Natomiast nie wykrycie rzeczywistego włamania zaliczamy do grupy *false negative*. Ostatni rodzaj błędnych decyzji systemu jest minimalizowany przez obniżanie poziomów definiujących anomalie. Ujemnym skutkiem takiego działania jest duży współczynnik alarmów *false positive*, co mocno pogarsza jakość działania anomaly detection systems. Ostatnią szczególną wadą tych systemów jest możliwość do wyuczenia ich akceptowania wrogich nadużyć jako normalnego zachowania[2].

Compound detection system. System, który jest hybrydą anomaly detection i misuse detection. Hybrydowość przejawia się w mechaniźmie decyzyjnym, który bazuje na analizie normalnego zachowania systemu i jednoczesnym badaniu znanych wzorców włamań. Wykrywanie anomalii pomaga w identyfikacji nowych, nieznanych ataków, podczas gdy misuse detection wykrywa znane nadużycia, oraz próby „złego wytrenowania" systemu. Systemy hybrydowe w porównaniu z poprzednio omawianymi rodzajami systemów charakteryzują się wzrostem liczby pozytywnych alarmów (*true positive*). Oznacza to jednocześnie redukcje błędnych alarmów (*false alarms*)[3].

Ontology detection system. Ontologia dostarcza zbioru termów i relacji, przy pomocy których można łatwo modelować dziedzinę funkcjonowania tworzonego IDS. Podejście wykorzystujące onotologię pozwala ekspertowi zaprojektować *intrusion detection application* bez znajomości podstaw tworzenia systemów wykrywania nadużyć. Zastosowanie ontologii pozwala na wyrażenie IDS w pojęciach dziedziny użytkownika. Prowadzi to do bardziej intuicyjnego procesu projektowania systemu. Z wiedzy zawartej w onotologii łatwiej jest wyprowadzić wiele specyficznych właściwości. Na przykład, system składający się z wielu komponentów jak analizator sygnałów, licznik wychodzących i przychodzących bajtów, zestaw flag. Wszystkie tego rodzaju parametry muszą być dokładnie wyspecyfikowane. Używając ontologii istnieje możliwość wyprowadzenia, że konkretna cecha, obiekt powinna być zamodelowana. Ostatnią zaletą ontology detection systems jest występowanie inteligentnego wnioskowania. Wykorzystywanie onotologii pozwala w IDS na szybkie znajdowanie interesujących relacji między obiektami[9].

5.2 Zbiór danych KDD Cup 99

W 1998 roku DARPA(Defense Advanced Research Projects Agency) zleciła MIT Lincoln Labs opracowanie programu oceny wykrywania włamań. Celem było poznanie i zbadanie aktualnego stanu wykrywania nadużyć. Lincoln Labs stworzyło środowisko, które przez 9 tygodni zbierało surowe dane TCP z lokalnej sieci symulującej U.S Air Force LAN. Zebrane dane treningowe liczyły 4 GB skompresowanych binarnych danych TCP zgromadzonych w trakcie 7 tygodni. Zostały one przetworzone w 5 milionów połączeniowych rekordów. Kolejne 2 tygodnie testowania danych przyniosły następne dwa miliony rekordów połączeniowych.

Połączenie jest sekwencją TCP pakietów zaczynających się i kończących się w ściśle określonych momentach czasu, pomiędzy którymi dane przepływają do i z źródłowego adresu IP do docelowego adresu IP. Każde połączenie jest etykietowane jako normalne lub atak. Ataki są zaliczane do jednej z czterech głównych kategorii: DOS (*denial-of-service*), R2L(*unauthorized access from a remote machine*), U2R(*unauthorized access to local superuser*) i Probing(próbkowanie/skanowanie). Dane zgromadzone przez laboratoria Lincolna zawierają 24 treningowych typów ataków, oraz kolejne 14 dodatkowych w zbiorze testowym[10].

W celu rozróżnienia połączeń normalnych od nadużyć zostały zdefiniowane wysoko-poziomowe cechy, które tworzą rekordy połączeniowe. Cechy są pogrupowane w kilku kategoriach:

- cechy typu *same host* – badają tylko połączenia z ostatnich 2 sekund, które mają ten sam docelowy host jak aktualne połączenie
- cechy typu *same service* - badają tylko połączenia z ostatnich 2 sekund, które mają tą samą usługę co aktualne połączenie
- cechy oparte na hoście – skonstruowane w oparciu o okno 100 połączeń do tego samego adresu
- cechy typu *content features* – zostały zbudowane w oparciu o wiedzę ekspertów, aby sprawniej poszukiwać podejrzanych zachowań w porcjach danych.

5.3 Eksperyment

Przy użyciu przykładowych trzech algorytmów uczenia: ID3(C.4.5), klasyfikatora Naive Bayes i sieci bayesowskich postaram się przedstawić podstawowe, praktyczne problemy *machine learning*. Problemy dokładności, szybkości klasyfikacji rozpatruje w kontekście wykrywania nadużyć, przy użyciu wcześniej opisanego zbioru KDD Cup 99.

Najpierw w skrócie przedstawię używane algorytmy uczenia:

- ID3 (inductive decision trees) – jest to struktura drzewiasta, w której węzły wewnętrzne zawierają testy na wartości atrybutów. Z każdego węzła wewnętrznego wychodzi tyle gałęzi ile jest możliwych wyników testu w tym węźle. Liście zawierają decyzje o klasyfikacji obiektów.
- Naive Bayes – jest prostym probabilistycznym klasyfikatorem. Jego działanie oparte jest na założeniu o wzajemnej niezależności cech. Ogólna reguła klasyfikacji sprowadza się do określenia najbardziej prawdopodobnej kategorii(C) przy zadanych cechach(f_i), można ją tak sformułować[16]:

$$\text{classify}(f_1, \ldots, f_n) = \text{argmax}_c \; p(C = c) \prod_{i=1}^{n} p(F_i = f_i | C = c)$$

- Sieci bayesowskie – skierowany acykliczny graf, w którym wierzchołki reprezentują zdarzenia, a łuki związki przyczynowe pomiędzy zdarzeniami. Służy przedstawieniu zależności między zdarzeniami, przy założeniu niezależności danego zdarzenia od wszystkich innych, które nie są jego potomkami.

Jednym z głównych sposobów optymalizacji procesu uczenia jest *preprocessing* danych. Pozwala on na lepsze sformułowanie, opisanie przykładów, a to z kolei przyczynia się do poprawy jakości uczenia. W swoim eksperymencie zajmuje się jedną z najbardziej efektywnych metod *preprocessingu* – selekcją atrybutów[17]. Zadaniem jej jest wskazanie istotnych dla procesu uczenia cech, a tym samym wyeliminowanie nieistotności i redundancji z wyjściowych zbiorów. Metody selekcji cech dzielą się na dwie podstawowe grupy:

- Filter – uniwersalne metody selekcji oparte na specyficznych metrykach do oceny i wyboru cech np. Correlation Feature Selection, Consistency Feature Selection, reduktory zbiorów przybliżonych[14].
- Wrapper – metody oceniające jakość zbioru atrybutów przy użyciu konkretnego algorytmu uczenia. Tak otrzymane zbiory cech są szyte pod dany algorytm np. ID3[11].

Do badania wpływu *preprocessingu* na dokładność i szybkość klasyfikacji wykorzystuje metodę selekcji atrybutów o nazwie *correlation based feature selection*. Metoda ta dokonuje oceny jakości kolejnych podzbiorów cech używając heurystyki oceny korelacji między cechami oraz kategorią. Wysokie noty są przypisywane do zbiorów atrybutów, które charakteryzują się wysokim skorelowaniem z kategorią a niską wewnętrzną korelacją[12]. Do generowania kolejnych podzbiorów cech używam algorytmu przeszukiwania – *best first search*. Charakteryzuje się on tym, iż tworzy nowe zbiory bazując na dodawaniu lub usuwaniu pojedynczych cech. Posiada zdolność do nawracania, aby odkrywać nowe możliwości, gdy obecna ścieżka poszukiwań nie daje żadnej poprawy.

Przeprowadzenie selekcji *Correlation Feature Selection* na zbiorze KDD Cup 99 ograniczyło zbiór cech z wyjściowych 41 do 17. Według miary korelacji zbiór atrybutów, ponad dwukrotnie mniejszy niż początkowy, jest najlepszym zestawem cech opisujących włamania. Logiczną konsekwencją zmniejszenia wymiarowości przestrzeni przykładów jest skrócenie czasu budowania modelu i przyspieszenie procesu właściwej klasyfikacji obiektów ze zbioru testowego. Powstaje jednak wątpliwość czy uproszczenie opisu przykładów nie odbije się istotnie na dokładności klasyfikacji.

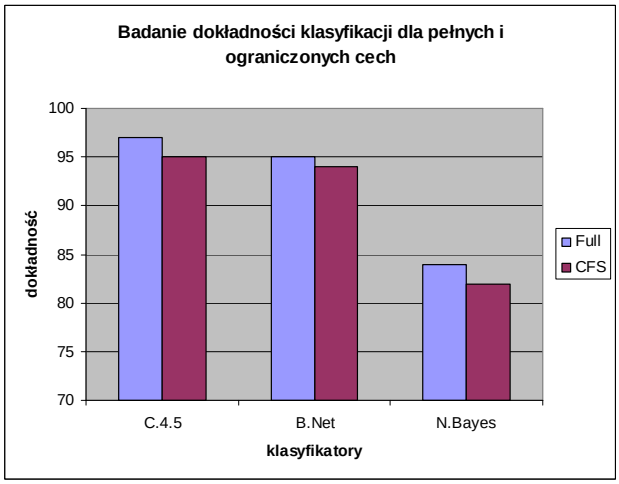

Fig. 1. Dokładność wykrywania nadużyć w zależności od stosowanego zbioru cech. Na fioletowo oznaczone wyniki dla pełnego zbioru cech(41 atrybutów), na purpurowo wyniki dla ograniczonego zbioru(17 atrybutów).

Na wykresie Fig.1 widać, iż zastosowanie ograniczonego zbioru cech tylko nieznacznie pogorszyło dokładność klasyfikacji. Odchylenia wynoszą maksymalnie 3%. Jest to tak mała wielkość, iż można by ją uznać za margines pomyłki. Systemy wykrywania nadużyć są jednak rygorystycznymi aplikacjami, więc uznajemy minimalne pogorszenie skuteczności działania jako koszt optymalizacji.

Zmniejszenie ilości cech polepszyło szybkość uczenia. Badanie czasu budowania modelu oraz czasu klasyfikacji zbioru testowego pozwoliły dokładnie określić skale optymalizacji.

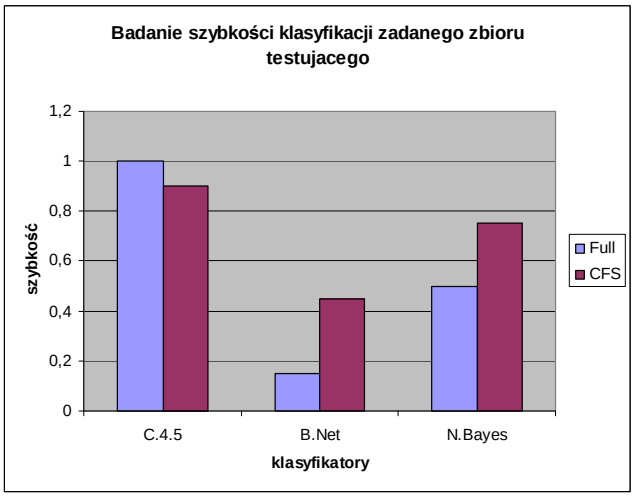

Fig. 2. Szybkość klasyfikacji przykładów w zależności od stosowanego zbioru cech. Na fioletowo oznaczono wyniki dla pełnego zbioru cech(41 atrybutów), na purpurowo wyniki dla ograniczonego zbioru(17 atrybut). Najszybsza klasyfikacja (najkrótsza) znormalizowana do 1.

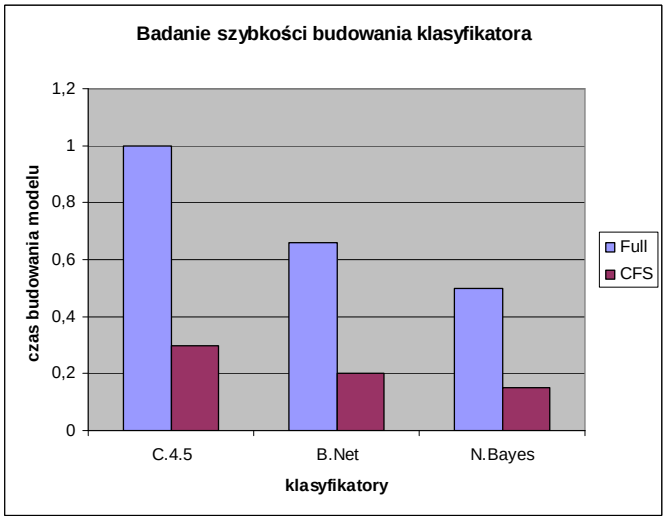

Fig. 3. Szybkość(czas) budowania klasyfikatora w zależności od stosowanego zbioru cech. Na fioletowo oznaczono wyniki dla pełnego zbioru cech(41 atrybutów), na purpurowo wyniki dla ograniczonego zbioru(17 atrybutów). Najdłuższy czas znormalizowany do 1.

Na wykresach Fig.2 i Fig.3 widać, iż optymalizacja zbioru cech wpływa na skrócenie czasu budowania modelu jak i samej klasyfikacji. Szczególnie czas budowania modeli ulega kilkakrotnemu zmniejszeniu. W przypadku szybkości klasyfikacji tylko drzewa decyzyjne po zastosowaniu ograniczonego zbioru cech nie poprawiają swoich wyników szybkościowych. Wynikiem tego może być specyficzność konstrukcji drzewa, gdzie mniejsza liczba cech nie oznacza zawsze prostszego, płytszego drzewa.

Podsumowując, zastosowanie selekcji atrybutów doprowadziło do bardzo dużej optymalizacji w zakresie szybkości klasyfikacji zbioru testowego i w zakresie budowania modelu. Jednocześnie ograniczony zbiór cech nie spowodował poważnego zmniejszenia dokładności klasyfikacji. Strata na dokładności jest nie porównywalna z zyskiem jaki odnosimy w kryteriach szybkościowych. Rodzi się pytanie jak wykorzystać korzyści powyższego *preprocessingu*. Moja odpowiedź to systemy hybrydowe.

Systemy hybrydowe to połączenie różnych technik, algorytmów w jeden układ. W *machine learning* możemy w ramach systemów hybrydowych łączyć różne techniki uczenia (np. indukcja i dedukcja) lub łączyć różne algorytmy uczenia w ramach tego samego sposobu nabywania wiedzy. Przykładem takiego systemu może być scalenie używanych przeze mnie algorytmów: naive bayes, bayesian net i ID3. Moduł decyzyjny może realizować głosowanie między algorytmami, lub stosować aproksymacje wagową ostatecznej kategorii. Dzięki selekcji atrybutów zarówno budowanie modelu jak i klasyfikacja są kilkakrotnie szybsze. W tym samym czasie, co pierwotnie realizowaliśmy klasyfikację za pomocą jednego algorytmu, możemy przeprowadzić kilka takich operacji przy użyciu różnych algorytmów. Ustalenie finalnej kategorii za pomocą głosowania polega na wyborze kategorii większościowej

wśród wyników pojedynczych metod. Z kolei aproksymacja wagowa jako wagę może używać dokładności wykrywania nadużyć. Ta metoda, która ma większą dokładność ma większy wpływ na ostateczny wybór kategorii docelowej. W ten sposób opierając się na kilku algorytmach możemy polepszyć końcową skuteczność klasyfikacji. Obecny rozwój systemów uczących charakteryzuje się małym przyrostem nowych algorytmów znacząco polepszających dokładność klasyfikacji. Dlatego najlepszym według mnie sposobem na optymalizacje skuteczności działania jest budowanie hybryd wzbogaconych o wstępne przygotowanie danych[13].

6. Spojrzenie w przyszłość

Machine Learning jest dość młodą dziedziną, ale już o w pełni wykrystalizowanych fundamentach i nurtach. Przez ostatnie kilkadziesiąt lat zostały stworzone różne rodzaje systemów dysponujące konkretnymi podstawami matematycznymi i modelami zastosowań. Kolejnym krokiem w nurcie teoretycznym uczących się maszyn powinno być utworzenie ogólnej teorii uczenia. Na tym poziomu rozwoju według mnie należy poszukiwać pewnego ogólnego opisu tego co udało się osiągnąć do tej pory. Usystematyzownie i wpisanie rodzajów uczenia w abstrakcyjny model pozwoli poznać nowe cechy systemów i określi nowe drogi poszukiwań.

W nurcie praktycznym nadchodzi czas systemów hybrydowych. Systemy tego rodzaju łączą różne algorytmy w ramach tej samej techniki lub scalają różne rodzaje uczenia. Na przykład można łączyć indukcje i EBL, indukcje z uczeniem ze wzmocnieniem. Takie systemy pozwalają pozbyć się słabych stron konkretnego uczenia. Jeden z takich systemów przedstawiłem przy opisie praktycznych problemów. Warunkiem koniecznym właściwego rozwoju hybryd jest szeroko rozumiany *preprocessing* danych. Odpowiednie odfiltrowanie zupełnych i istotnych danych pozwoli na szybkie i skuteczne uczenie. Daje to możliwość efektywnej pracy hybryd. Kluczowym aspektem badań jest też poszukiwanie nowych reprezentacji wiedzy, pozwalających na przekazywanie większej ilości informacji. Dobrym przykładem jest zastępowanie prostych reguł zdaniowych logiką predykatów, które pozwalają wyrażać jednocześnie związki między wieloma obiektami.
Ciekawym pomysłem na tworzenie systemów uczenia jest użycie ontologii. Pozwala to nam się wspiąć na wyższy poziom abstrakcji i usprawnić proces budowania systemów. Przechodząc od dokładnie zbudowanego modelu dziedziny problemu do niskiego poziomu aplikacji uczącej.

Z pewnością jest jeszcze wiele innych aspektów, które wymagają nasilenia badań, chociażby poprawa szybkości przetwarzania zbiorów trenujących, ale według mnie podstawowe wskazałem. Mam nadzieję, że zasugerowane kryteria porównawcze lepiej pozwoliły zrozumieć systemy uczące i ich rosnącą rolę w otaczającym nas świecie.

References

1. Samuel A.L.: Some studies in machine learning using the game of checkers, IBM Journal on Research and Development, 1959, 3, s. 210- 222.
2. Patcha A., Park J.: An overview of anomaly detection techniques: Existing solutions and latest technological trends, Computer Networks 51, 2007, s. 3450-3470.
3. Shon T., Moon J.: A hybrid machine learning approach to network anomaly detection, Information Sciences 177, 2007, s. 3779-3821.
4. Cichosz P.: Systemy uczące się, WNT, 2007, s. 620- 634.
5. Mitchell T.M.: Machine Learning, McGraw-Hill, 1997.
6. Luger G.F, Stubblefield W.A: Artificial Intelligence, Addison Wesley Longman, 1998, s. 646-649.
7. Fawcett T., Provost F.: Adaptive Fraud Detection, Data Mining and Knowledge Discovery 1, 1997, s. 219-316.
8. Russel S., Norvig P.: Artificial Intelligence: A Modern Approach, Prentice Hall, 2003.
9. Hung S.-S., Shing-Min Liu D.: A user-oriented ontology-based approach for network intrusion detection, Computer Standards & Interfaces 30, 2008, s.78-88.
10. Stolfo S., Wei F., Prodromidis A.: Cost-based Modeling for Fraud and Intrusion Detection: Results from the JAM Project, DARPA Information Survivability Conference, 2000.
11. Hall M., Smith L.: Feature Selection for Machine Learning: Comparing a Correlation-based Filter Approach to the Wrapper, Springer, 1995.
12. Hall M.: Correlation-based Feature Selection for Machine Learning, Ph.D diss. Dept. of Computer Science, Waikato Univ.
13. Peddabachigari S. et al.: Modeling intrusion detection system using hybrid intelligent systems, Journal of Network and Computer Applications 30, 2007, s.114-132.
14. Liu Ciu-Juan: The application of rough sets on network intrusion detection, Proceedings of Sixth International Conference on Machine Learning and Cybernatics, Hong Kong, 2007.
15. Depren O., Topallar M., Anarim E.: An intelligent intrusion detection system for anomaly and misuse detection in computer networks, Expert Systems with Applications 29, 2005.
16. Panda M., Patra M.: Network Intrusion Detection using Naïve Bayes, International Journal of Computer Science and Network Security, 2007.
17. Guyon I., Elisseeff A.: An Introduction to Variable and Feature Selection, Journal of Machine Learning Research 3, 2003.