INFORMATYKA

Efekty uczenia się i treści programowe zajęć:

Nazwa zajęć**: *Analiza danych sportowych***

Po zakończeniu zajęć i potwierdzeniu osiągnięcia efektów uczenia się student:

- Zna tematykę badawczą związaną z analizą i wnioskowaniem z dziedzinowych źródeł danych. Potrafi przygotować środowisko programistyczne do przetwarzania danych tekstowych, obrazów i wideo oraz wykonywać w nim podstawowe operacje.
- Potrafi budować repozytoria danych cyfrowych – w tym pobierać je wybranymi metodami, analizować i budować strukturę danych do przetwarzania.
- Potrafi stosować metody czyszczenia i integracji danych pochodzących z różnych źródeł.
- Potrafi stosować zaawansowane algorytmy przetwarzania danych, statystyki oraz prowadzić proces dedukcyjny.
- Potrafi stosować oraz interpretować metryki związane ze „skutecznością" zdarzeń w spotkaniach sportowych w różnych dyscyplinach.
- Potrafi wykonywać segmentację i rozpoznawanie obrazów.
- Potrafi analizować materiały wideo w celu śledzenia obiektów.
- Potrafi wykrywać i rozpoznawać tekst na obrazach.
- Potrafi dobierać, stosować oraz modyfikować biblioteki wizualizacji danych w zależności od potrzeb analityka danych.
- Zna i potrafi stosować najnowsze osiągnięcia w zakresie analizy danych sportowych, np. w celu sugerowania decyzji w czasie rzeczywistym na podstawie danych historycznych.

Treści programowe dla zajęć:

- Wprowadzenie do przetwarzania danych sportowych: przygotowanie środowiska programistycznego (Azure) – przegląd modułów; podstawowe operacje związane z pozyskiwaniem i zapisywaniem danych, budowanie repozytoriów.
- Omówienie istotnych cech opisujących zdarzenia sportowe w poszczególnych dyscyplinach (i wybranych ligach) – koszykówka – NBA, hokej na lodzie – NHL, piłka nożna i tenis. Analiza dostępnych repozytoriów danych:
    - Tekstowych,
    - plików video,
    - posegmentowanych obrazów.
- Metody czyszczenia oraz integracji danych pochodzących z różnych (niezależnych) źródeł.
- Przygotowanie danych do wnioskowania, zastosowanie metod statystycznych, w tym szeregów czasowych.
- Metryki analizy danych o zdarzeniach sportowych. Omówienie oraz wybrane implementacje
    - metryki stosowane w sportach zespołowych,
    - metryki rozproszenia graczy podczas gier zespołowych,
    - metryki charakteryzujące zawodników (m. in. Entropia Shannona, podłużne i boczne przemieszczenia w sytuacji meczowej, entropia Kołmogorowa, eksploracja przestrzenna),
    - metryki stosowane w ocenie taktyk drużyn sportowych.
- Segmentacja i rozpoznawanie obrazów: segmentacja obrazów, rozpoznawanie wybranych cech obiektów, klasyfikacja obrazów.
- Analiza wideo: przepływ optyczny, śledzenie obiektów, analiza równoważnych klas zdarzeń.
- Ręczne oraz półautomatyczne tagowanie zdarzeń sportowych w zakresie zarówno danych graficznych, jak i danych video. Wykrywanie i rozpoznawanie tekstu.
- Wizualizacja danych ze zdarzeń sportowych oraz analizy zachowania/zaangażowania zawodników podczas treningów i zawodów sportowych. Odwzorowanie boisk, symulacje oraz agregacja zdarzeń podobnych.
- Przedstawienie wybranej interpretacji danych sportowych w postaci opracowania analitycznego. Zastosowanie wybranych narzędzi i bibliotek do budowy systemu rekomendacyjnego.

Nazwa zajęć**: *Bezpieczeństwo oprogramowania***

Po zakończeniu zajęć i potwierdzeniu osiągnięcia efektów uczenia się student:

- Rozumie problematykę związaną z bezpieczeństwem oprogramowania. Rozumie, że odpowiedzialność za użytkowania oprogramowania użytkownika i twórcy oprogramowania nie może być wyłączna.
- Zna różnice pomiędzy błędami występującymi w oprogramowaniu a problemami związanymi z bezpieczeństwem.
- Potrafi podać przykłady niebezpieczeństw związanych z oprogramowaniem.
- Zna problemy związane z oprogramowaniem wielowątkowym.
- Potrafi stosować wybrane narzędzia automatyzujące problem wyszukiwania problemów bezpieczeństwa w oprogramowaniu.
- Zna problemy związane z bezpieczeństwem przechowywania danych w systemach komputerowych.
- Zna problemy związane ze wstrzykiwaniem kodu.
- Zna wybrane metody ochrony przed niebezpiecznymi programami dostępne w systemach operacyjnych.
- Zna problematykę kodu samomodyfikującego się oraz wybranych zabezpieczeń stosowanych w oprogramowaniu.

Treści programowe dla zajęć:

- Granice odpowiedzialności użytkownika i twórcy oprogramowania. Rodzaje problemów występujących w oprogramowaniu.
- Błędy bezpieczeństwa w oprogramowaniu. Dlaczego „bezbłędne” oprogramowanie niekoniecznie jest bezpieczne?
- Przykładowe problemy związane z bezpieczeństwem, występujące w oprogramowaniu (np. Timing attack). Zastosowania kryptograficzne, losowość.
- Obsługa blokad w oprogramowaniu wielowątkowym. Blokady plików w różnych językach programowania.
- Valgrind (Cachegrind, Memcheck, Helgrind, Massif) jako przykład narzędzia wspomagającego znajdowanie błędów w oprogramowaniu.
- Problem przechowywania wrażliwych danych (jak je bezpiecznie przechować / usunąć), pliki tymczasowe, obrazy pamięci, hibernacja systemu, środowisko wykonania programu, zagadnienia sprzętowe.
- Problemy bezpieczeństwa WWW (XSS, CSRF, XSRF) – narzędzia ataków i sposoby ochrony.
- SQL injection (i inne problemy typu "injection").
- LSM (SELinux, AppArmor), Grsecurity – zasady działania i tworzenie polityk bezpieczeństwa. Capabilities w Linuksie. Zabezpieczenia środowiska (chroot, sandbox, code signing, exec shield, dm-verity).
- Modyfikacje kodu w trakcie wykonania programu. Zabezpieczenia gier. Zabezpieczenia przed kopiowaniem.

Nazwa zajęć**: *Cybernetyczne działania wojenne***

Po zakończeniu zajęć i potwierdzeniu osiągnięcia efektów uczenia się student:

- Potrafi zrozumieć znaczenie cyberbezpieczeństwa dla całościowego problemu procesu zapewnienia odpowiedniego poziomu bezpieczeństwa państwa w XXI w.
- Potrafi zauważyć możliwość wykorzystania przez obce państwa lub struktury niepaństwowe domeny cyber do działań szpiegowskich oraz wojskowych wymierzonych w jego państwo.
- Potrafi zrozumieć możliwość wykorzystania przez obce państwa lub struktury niepaństwowe domeny cyber do działań szpiegowskich względem własnych zasobów informacji, a także zasobów znajdujących się po stronie instytucji/firmy, w której pracuje.
- Potrafi zauważyć rolę i znaczenie domeny cyber dla całościowego systemu obronnego państwa w warunkach nowych sposobów prowadzenia działań zbrojnych i uderzeń niekinetycznych w toku operacji hybrydowych.
- Potrafi zrozumieć zróżnicowane postawy względem cyberbezpieczeństwa definiowane przez system polityczny danego państwa oraz formy zabezpieczenia jego interesów narodowych.
- Rozumie potrzebę przekładania własnych działań zawodowych i pozazawodowych na stworzenie odpowiedniego poziomu cyberbezpieczeństwa oraz odporności systemu obronnego państwa.
- Rozumie wymóg poznawania innych niż techniczne aspektów cyberbezpieczeństwa.

- Rozumie potrzebę współpracy z osobami zajmującymi się cyberbezpieczeństwem z różnych perspektyw naukowych oraz zawodowych.

Treści programowe dla zajęć:

- Nowa domena działań, czyli jak państwa oraz sojusze obronne postrzegają cyberprzestrzeń w ujęciu wywiadów oraz wojska
- Kryptologia i rozwój SIGINT-u jako kluczowego źródła pozyskiwania danych wywiadowczych w XX i XXI w., czyli o tym, czy cyberszpiegostwo staje się niezbędne
- Cyberprzestrzeń pełnoprawną domeną prowadzenia działań zbrojnych, czyli jak współczesne konflikty zbrojne przechodzą do cyberprzestrzeni – od Iraku 1991 do Górskiego Karabachu 2020
- C4ISTR, czyli gdzie cyberbezpieczeństwo zaczyna decydować o współczesnych sukcesach oraz porażkach sił zbrojnych
- Chińsko-amerykańska „wojna" o cyberprzestrzeń, czyli jak kształt rywalizacji mocarstw globalnych jest definiowany względami cyberbezpieczeństwa
- Państwo Izrael, czyli przykład państwa, w którym wojsko i służby specjalne rozumieją strategiczne potrzeby oraz możliwości wykorzystania domeny cyber na Bliskim Wschodzie
- Cyberprzestrzeń i WRE czyli jak zrodził się rosyjski pomysł na ograniczenie dysproporcji względem państw NATO

Nazwa zajęć**: *Ekstrakcja informacji***

Po zakończeniu zajęć i potwierdzeniu osiągnięcia efektów uczenia się student:

- Rozumie sposób działania tradycyjnych wyszukiwarek tekstowych. Zna sposoby przetwarzania tekstu na potrzeby wyszukiwarek tekstowych (lowercasing, tokenizacja, tworzenie wektorów TF-IDF). Potrafi wyjaśnić sposób, w jaki rankingowane są dokumenty w tradycyjnej wyszukiwarce (TF-IDF scoring, Okapi BM25). Zna pojęcie inverted index i jego znaczenie w optymalizacji zapytań.
- Umie przetwarzać tekst w celu stworzenia wyszukiwarki tekstowej przy pomocy gotowych bibliotek. Potrafi stworzyć ranking dokumentów dla zapytania.
- Zna popularne silniki wyszukiwarek. Zna sposoby ewaluacji wyszukiwarek internetowych (ludzka ewaluacja, cumulative gain i jej pochodne, testy A/B), ich zalety i słabości. Potrafi wyjaśnić pojęcie relewantnej wyszukiwarki. Zna techniki poprawiania jakości wyszukiwarki oraz cykl budowy i utrzymania wyszukiwarki. Rozumie komercyjne i społeczne znaczenie wyszukiwarek internetowych.
- Potrafi stworzyć wyszukiwarkę przy pomocy gotowych silników wyszukiwarek. Umie poprawić jakość wyszukiwarki dla konkretnych przypadków użycia. Jest w stanie zwiększyć precyzję lub pokrycie wyników wyszukiwania. Umie wyznaczać popularne metryki rankingu przy wykorzystaniu gotowych narzędzi.
- Zna obecny stan badań w wyszukiwarkach tekstowych. Rozumie pojęcie wyszukiwania semantycznego. Zna obecne mechanizmy interakcji wyszukiwarki z użytkownikiem (podpowiadanie frazy, ghosting, rekomendacje, korekta frazy).
- Potrafi stworzyć wyszukiwarkę bazującą na podobieństwie semantycznym frazy do dokumentu.
- Rozumie pojęcie ekstrakcji informacji i zadania, które należą do tej dziedziny. Zna praktyczne zastosowania ekstrakcji informacji.
- Potrafi stworzyć system ekstrakcji fraz z określonego korpusu tekstów używając połączenia metod lingwistycznych i statystycznych.
- Zna pojęcie zadania sequence labeling oraz jego zastosowania: named entity recognition (NER), part-of-speech tagging (POS) i inne. Rozumie różne sposoby ewaluacji zadania sequence labeling. Potrafi wyjaśnić podstawy działania statystycznych modeli sequence labeling: hidden markov models (HMM) oraz conditional random fields (CRF).
- Umie stworzyć i ewaluować statystyczny model sequence labeling i zastosować go w zadaniu NER.
- Zna neuronowe podejścia do zadania sequence labeling bazujące na sieciach rekurencyjnych.
- Potrafi zaimplementować architekturę rekurencyjnego neuronowego modelu NER przy użyciu biblioteki sieci neuronowych. Potrafi wykonać trening tak zdefiniowanego modelu w celu uzyskania jak najlepszej jakości modelu.
- Zna neuronowe podejścia do zadania sequence labeling bazujące na sieciach neuronowych typu transformer.

- Potrafi użyć pretrenowany model typu transformer w celu jego dostrojenia do zadania NER. Potrafi porównać różne modele NER (statystyczne, rekurencyjne sieci neuronowe, sieci neuronowe typu transformer).
- Rozumie zadanie sumaryzacji tekstu i sposoby jego ewaluacji. Zna aktualny stan badań w zakresie sumaryzacji tekstu, w szczególności modele transformer i transformer text-to-text.
- Potrafi wykorzystać neuronowy model typu transformer do zadania sumaryzacji tekstu.
- Zna zadanie i obecny stan badań odnośnie neuronowej ekstrakcji informacji z nieustrukturyzowanych dokumentów należących do jednej dziedziny.
- Potrafi wykorzystać neuronowy model oparty o sieci typu transformer do ekstrakcji informacji z nieustrukturyzowanych dokumentów należących do jednej dziedziny.

Treści programowe dla zajęć:

- Wprowadzenie do wyszukiwarek tekstowych, przetwarzanie tekstu na potrzeby wyszukiwarek tekstowych, algorytmy rankingowania wyszukiwarek tekstowych (TF-IDF, Okapi BM25) optymalizacja działania wyszukiwarek (inverted index).
- Przetwarzanie tekstu i stworzenie prostego systemu rankingującego dokumenty dla określonego zapytania przy wykorzystaniu prostych narzędzi i bibliotek (innych niż silniki wyszukiwarki).
- Wyszukiwarki tekstowe – część zaawansowana. Dostępne komercyjnie silniki wyszukiwarek (Solr, Elasticsearch), sposoby ewaluacji wyszukiwarek, cykl rozwoju i utrzymania wyszukiwarki i dopasowanie pod potrzeby biznesowe klienta, znaczenie społeczne wyszukiwarek internetowych.
- Stworzenie wyszukiwarki przy użyciu narzędzia ElasticSearch. Optymalizacja skuteczności wyszukiwania pod daną frazę. Liczenie metryki rankingu nDCG przy pomocy gotowych narzędzi.
- Najnowsze badania i trendy w wyszukiwarkach tekstowych, wyszukiwanie semantyczne, mechanizmy interakcji między wyszukiwarką i użytkownikiem.
- Implementacja prostej wyszukiwarki semantycznej.
- Ekstrakcja informacji – wprowadzenie, nakreślenie problemów i zadań, zastosowania.
- Implementacja systemu ekstrakcji fraz dla specjalistycznego korpusu tekstowego przy użyciu metod lingwistycznych i statystycznych.
- Zadanie sequence labelling i jego zastosowania (NER, POS). Ewaluacja zastosowań sequence labeling, statystyczne modele sequence labeling (HMM i CRF).
- Stworzenie statystycznego modelu NER wykorzystującego HMM lub CRF.
- Podstawy sieci neuronowych, neuronowy rekurencyjny model sequence labeling.
- Zapoznanie z sieciami neuronowymi w bibliotece pytorch, implementacja neuronowego rekurencyjnego modelu NER i trenowanie modelu.
- Neuronowe modele sequence labeling wykorzystujące sieci typu transformer.
- Dostrajanie pretrenowanego modelu typu transformer do zadania NER. Porównanie jakości różnych modeli NER stworzonych na zajęciach (statystycznego, neuronowego bazującego na sieciach rekurencyjnych, neuronowego bazującego na sieci typu transformer).
- Zadanie sumaryzacji tekstu i jego ewaluacja, aktualny stan badań w zakresie sumaryzacji tekstu ze szczególnym uwzględnieniem modeli typu transformer i transformer text-to-text.
- Wykorzystanie pretrenowanego modelu typu transformer do zadania sumaryzacji tekstu.
- Zadanie ekstrakcji informacji z nieustrukturyzowanych dokumentów domenowych – definicja zadania, modele neuronowe typu transformer, obecny stan wiedzy naukowej.
- Wykorzystanie modelu neuronowego typu transformer do zadania ekstrakcji informacji z nieustrukturyzowanych dokumentów należących do jednej dziedziny.

Nazwa zajęć**: *Gry kombinatoryczne***

Po zakończeniu zajęć i potwierdzeniu osiągnięcia efektów uczenia się student:

- Potrafi przedstawić proste gry kombinatoryczne w postaci ekstensywnej (drzewa gry). Potrafi zastosować na drzewie gry analizę wstecz.
- Potrafi rozstrzygnąć, kto ma strategię wygrywającą lub nieprzegrywającą w prostych grach, korzystając z metody kradzieży strategii i metody ruchów odpowiadających. Potrafi oceniać poprawność strategii.
- Umie ocenić złożoność obliczeniową prostych algorytmów związanych z poszukiwaniem dobrej strategii w grach kombinatorycznych.
- Umie przedstawiać tok swojego rozumowania w sposób zrozumiały dla słuchaczy.

Treści programowe dla zajęć:

- Definicja gry kombinatorycznej. Przykłady gier kombinatorycznych, w tym HEX, szachy, kółko i krzyżyk. Drzewo gry, wartość gry. Analiza wstecz drzewa gry. Twierdzenie o strategii nieprzegrywającej. Ocena złożoności obliczeniowej naiwnych metod analizy drzewa gry.
- Techniki pomagające w ocenie gry: metoda kradzieży strategii, metoda ruchów odpowiadających. Analiza poprawności strategii.
- Narzędzia teoriografowe w grach. Gra Shannona, gry ramseyowskie, wielowymiarowe gry w kółko i krzyżyk.
- Gry NIM.
- Gry kombinatoryczne a złożoność obliczeniowa i hipoteza P<>NP. Przykłady gier NP-zupełnych, PCSPACE-zupełnych, EXPTIME-zupełnych.
- Algorytmy alfa-beta analizy drzewa gry. Deep Blue i inne programy szachowe.
- Przeszukiwanie Monte Carlo drzewa gry (MCST). AlphaGo.

Nazwa zajęć**:** ***Inteligencja Obliczeniowa***
Po zakończeniu zajęć i potwierdzeniu osiągnięcia efektów uczenia się student:

- Zna matematyczne podstawy inteligencji obliczeniowej
- Zna standardowe metody inteligencji obliczeniowej
- Umie zaprogramować algorytmy inteligencji obliczeniowej
- Potrafi przygotować dane na potrzeby algorytmów inteligencji obliczeniowej

Treści programowe dla zajęć:

- Wprowadzenie do inteligencji obliczeniowej.
- Współczesne kierunki badań i zastosowań
- inteligencji obliczeniowej.
- Logika i zbiory rozmyte
- Algorytmy genetyczne i ewolucyjne
- Sztuczne sieci neuronowe
- Projekt

Nazwa zajęć**:** ***Inżynieria uczenia maszynowego***
Po zakończeniu zajęć i potwierdzeniu osiągnięcia efektów uczenia się student:

- Zna specyfikę rozwoju modeli uczenia maszynowego.
- Zna i rozumie podstawowe pojęcia i koncepcje związane z procesem ciągłej integracji.
- Zna popularne biblioteki uczenia maszynowego i różnice między nimi.
- Potrafi przeprowadzić trening i ewaluację prostego modelu uczenia maszynowego przy użyciu wybranej biblioteki.
- Potrafi przygotować dane trenujące model uczenia maszynowego i podzielić je na podzbiory. Rozumie na czym polega ewaluacja krzyżowa.
- Rozumie podstawę działania kontenerów w systemie operacyjnym Linux i zna ich zastosowania.
- Potrafi korzystać z narzędzia konteneryzacji Docker.
- Rozumie, na czym polega proces ciągłej integracji, potrafi podać przykład takiego procesu.
- Potrafi utworzyć i skonfigurować zadanie w systemie ciągłej integracji Jenkins.
- Potrafi zintegrować system ciągłej integracji z systemem kontroli wersji.
- Potrafi stworzyć proces ciągłej integracji modelu uczenia maszynowego w systemie ciągłej integracji Jenkins.
- Potrafi stworzyć wykres wizualizujący wyniki działania modelu uczenia maszynowego.
- Potrafi zintegrować proces wizualizacji wyników ewaluacji modelu uczenia maszynowego z procesem ciągłej integracji w środowisku Jenkins.
- Zna popularne narzędzia do kontroli danych, parametrów i wyników eksperymentów uczenia maszynowego.
- Potrafi zastosować wybrane narzędzie do kontroli danych, parametrów i wyników eksperymentów uczenia maszynowego.
- Potrafi zintegrować wybrane narzędzie do kontroli danych, parametrów i wyników eksperymentów uczenia maszynowego z narzędziem do wizualizacji wyników. Potrafi zinterpretować wyniki przedstawione w tym narzędziu.
- Potrafi zintegrować wszystkie elementy kompletnego systemu ciągłej integracji uczenia maszynowego i używać tego systemu do rozwoju modeli uczenia maszynowego.

Treści programowe dla zajęć:

- Wprowadzenie podstawowych koncepcji ciągłej integracji, konteneryzacji i uczenia maszynowego. Przedstawienie specyfiki procesów rozwoju, testowania i integracji modeli uczenia maszynowego.
- Przegląd najpopularniejszych bibliotek używanych w uczeniu maszynowym.
- Sposoby podziału danych trenujących i ewaluacji modeli uczenia maszynowego.
- Przygotowanie danych i bazowego rozwiązania dla wybranego problemu uczenia maszynowego.
- Konteneryzacja przy użyciu narzędzia Docker. Stworzenie obrazu Docker zawierającego środowisko potrzebne do przeprowadzania prostego eksperymentu uczenia maszynowego dla wybranego wcześniej problemu.
- Wprowadzenie do ciągłej integracji. Przedstawienie istniejących środowisk ciągłej integracji.
- Zapoznanie z możliwościami i zastosowaniami ciągłej integracji na przykładzie systemu Jenkins.
- Stworzenie pierwszego zadania w systemie Jenkins uruchamiające prosty eksperyment uczenia maszynowego.
- Połączenie kilku zadań Jenkins ze sobą w całość automatyzującą wielostopniowy proces uczenia maszynowego.
- Wizualizacja wyników eksperymentów za pomocą biblioteki do tworzenia wykresów, np. Matplotlib, stworzenie zadania Jenkins realizującego wizualizację
- Przegląd narzędzi do kontroli danych, parametrów i wyników eksperymentów: DVC, Sacred, MLFlow.
- Kontrola parametrów i wyników eksperymentów uczenia maszynowego przy pomocy wybranego narzędzia.
- Wizualizacja wyników eksperymentów uczenia maszynowego przy pomocy wybranego narzędzia.
- Integracja kompletnego procesu trenowania i ewaluacji modelu uczenia maszynowego przy użyciu poznanych wcześniej narzędzi
- Podsumowanie wyników pracy podczas kursu.

Nazwa zajęć**:** ***Inżynieria wiedzy***
Po zakończeniu zajęć i potwierdzeniu osiągnięcia efektów uczenia się student:

- Zna pojęcia związane z zarządzaniem wiedzą.
- Zna wybrane historyczne narzędzia i metody przeznaczone do reprezentacji i organizacji wiedzy.
- Potrafi wskazać obszary zastosowania wybranych metod reprezentacji i organizacji wiedzy.
- Umie wykorzystać język logiki w reprezentacji wiedzy.
- Zna obszary zastosowania języków informacyjnych oraz różne sposoby organizacji wiedzy, w tym różne typy relacji pomiędzy pojęciami.
- Potrafi wskazać obszary zastosowania wybranych języków informacyjnych z uwzględnieniem kontekstu i potrzeb budowanego systemu informatycznego opartego o wiedzę.
- Zna sposoby definiowania i formalizacji ontologii. Zna wybrane metody i narzędzie wykorzystywane przy budowie i przetwarzaniu ontologii.
- Zna sposoby definiowania grafów wiedzy (ang. knowledge graph).
- Potrafi wskazać obszary zastosowania ontologii, ich struktury oraz narzędzia wykorzystywane do ich budowy.
- Potrafi wskazać obszary zastosowania grafów wiedzy oraz rozstrzygać o postaci zasobów w tej postaci.
- Potrafi budować ontologie w wybranym formacie oraz przy pomocy odpowiednich metod i narzędzi.
- Zna pojęcia związane z Linked data oraz zna metody i narzędzie wykorzystywane do budowy zasobów tego typu. Zna obszary zastosowania Linked data we współczesnej gospodarce
- Potrafi wskazać obszary wykorzystania Linked data i budować zasoby tego typu.
- Zna cykl życia produktów opartych na zasobach wiedzowych oraz zna zasady uzyskania zasobów tego typu wysokiej jakości.
- Zna strukturę leksykalnych baz danych typu wordnet oraz ontologii typu wordnet. Zna podstawowe zasady budowy zasobów tego typu.
- Potrafi wskazać różnice przy budowie leksykalnych baz danych typu wordnet oraz ontologii typu wordnet

- Potrafi wskazać obszary zastosowania i narzędzia wykorzystywane przy budowie leksykalnych baz danych typu wordnet oraz ontologii typu wordnet.

Treści programowe dla zajęć:

- Wprowadzenie do zarządzania wiedzą (dane, informacja, wiedza, składnia, semantyka, taksonomie, ontologie).
- Przegląd wybranych historycznych narzędzi i metod przeznaczonych do reprezentacji i organizacji wiedzy (sieci semantyczne, ramy, grafy konceptualne) i ich wykorzystanie w systemach eksperckich. Implementacja przykładowej bazy wiedzy w języku logiki.
- Języki informacyjne w organizowaniu informacji (systemy klasyfikacyjne, słowniki, tezaurusy, tagowanie, metadane w opisywaniu treści) i przykładowe zasoby tego typu (Geonames, Getty AAT). Różne typy relacji (synonimia, polihierarchiczność, polirelacyjność, podobieństwo, powiązania). Analiza wybranych obszarów zastosowań.
- Ontologie, różne sposoby ich definiowania i zapisu. Wykorzystanie w organizacji informacji i zarządzaniu wiedzą. Przykładowe ontologie (Cidoc CRM, Sumo, CYC, ontologie dziedzinowe). Języki zapisu i przetwarzania ontologii (RDF, OWL, SPARQL). Budowa grafów wiedzy (knowledge graph) i obszary ich wykorzystania we współczesnej gospodarce opartej na wiedzy. Implementacja przykładowej ontologii.
- Linked data jako przykład zastosowania inżynierii wiedzy i zarządzania wiedzą.
- Cykl życia produktów opartych na zasobach wiedzowych. Uzyskanie zasobów wiedzowych wysokiej jakości.
- Leksykalne bazy danych typu wordnet i ich przykłady (WordNet, plWordnet). Ontologie typu wordnet jako przykład polihierarchicznego i polirelacyjnego systemu zarządzania wiedzą na przykładzie ontologii PMAH. Analiza wybranych obszarów zastosowań i narzędzi wykorzystywanych przy przetwarzaniu zasobów tego typu.

Nazwa zajęć**: Inżynieria wsteczna złośliwego oprogramowania (Malware Reverse Engineering)**

Po zakończeniu zajęć i potwierdzeniu osiągnięcia efektów uczenia się student:

- Potrafi wytłumaczyć, czym jest złośliwe oprogramowanie, przedstawić podstawowe techniki wykorzystywane do jego analizy.
- Potrafi stworzyć własne laboratorium do pracy ze złośliwym oprogramowaniem.
- Rozumie, czym są cechy statyczne plików, jak je sprawdzić i zinterpretować.
- Potrafi przeprowadzić analizę dynamiczną w bezpiecznym środowisku.
- Potrafi korzystać z debuggera.
- Zna popularne metody obfuskacji kodu.
- Wie, w jakim celu wykonuje się analizę złośliwego oprogramowania. Ma świadomość zagrożenia, jakie stanowi złośliwe oprogramowanie dla funkcjonowania społeczeństwa i gospodarki.
- Potrafi korzystać z deasemblera.
- Zna podstawy języka asembler.
- Potrafi przedstawić zasadę działania różnych typów złośliwego oprogramowania.
- Potrafi wykonać analizę ruchu sieciowego.
- Potrafi wykonać analizę złośliwych skryptów.
- Wie, z jakich technik utrudniających analizę korzystają twórcy złośliwego oprogramowania oraz jak je omijać.
- Potrafi wykonać analizę złośliwych dokumentów pakietu Microsoft Office oraz PDF.
- Potrafi przeprowadzić analizę zrzutu pamięci.

Treści programowe dla zajęć:

- Wprowadzenie do analizy złośliwego oprogramowania: Czym jest złośliwe oprogramowanie? Konfiguracja laboratorium do przeprowadzania analizy. Metody i techniki analizy.
- Analiza statyczna: Narzędzia i techniki. Open source intelligence.
- Analiza dynamiczna: Narzędzia i techniki monitorujące działanie złośliwego oprogramowania. Interakcja ze złośliwym oprogramowaniem.
- Podstawy analizy kodu: Korzystanie z debuggera. Popularne metody obfuskacji kodu.
- Cele analizy: Cykl zapewniania bezpieczeństwa w przedsiębiorstwach. Rola analityków złośliwego oprogramowania. Indicators of Compromise (IOCs)
- Analiza kodu języka niskiego poziomu: Korzystanie z deasemblera. Podstawy języka asemblera.
- Popularne techniki wykorzystywane przez złośliwe oprogramowanie: Rootkits. Keyloggers. Downloaders. HTTP C2 channels.

- Przechwytywanie ruchu sieciowego: iNetSim, iptables, Wireshark.
- Interakcja ze złośliwymi stronami internetowymi: tor, wget, curl, CapTipper, NetworkMiner. Deobfuskacja skryptów.
- Self-defending malware: wykrywanie oraz omijanie zabezpieczeń utrudniających analizę.
- Analiza złośliwych dokumentów: Pliki pakietu Microsoft Office. Pliki PDF.
- Analiza pamięci. Czym jest i jakie korzyści daje analiza pamięci? Podstawy korzystania z pakietu Volatility.

Nazwa zajęć**: Komputerowe wspomaganie tłumaczenia**
Po zakończeniu zajęć i potwierdzeniu osiągnięcia efektów uczenia się student:

- Zna podstawowe techniki komputerowego wspomagania tłumaczenia przy użyciu pamięci tłumaczeń.
- Zna zaawansowane techniki użycia pamięci tłumaczeń oraz kierunki badań w tej dziedzinie.
- Potrafi zaimplementować działający ekstraktor terminologii używając różnych technik rozwiązania problemu.
- Potrafi zbudować słownik dziedzinowy używając samodzielnie zaprojektowanych algorytmów.
- Potrafi używać wyrażeń regularnych do zadań związanych ze wspomaganiem pracy tłumacza.
- Zna kryteria użyteczności systemów tłumaczenia automatycznego oraz potrafi dokonać ewaluacji tych systemów.
- Potrafi używać technik web scrapingu do pozyskiwania zasobów lingwistycznych.
- Posiada wiedzę i umiejętności w dziedzinie automatycznego urównoleglania tekstów.
- Potrafi wykonać ewaluację przy użyciu oprogramowania typu key logger.
- Zna techniki korekty pisowni, potrafi zaimplementować własny korektor pisowni.
- Zna algorytmy korekty gramatycznej tekstu, potrafi zaimplementować własny korektor gramatyczny na podstawie znanych rozwiązań.
- Potrafi skonstruować własny system komputerowego wspomagania tłumaczenia.

Treści programowe dla zajęć:

- Zapoznanie z podstawowymi pojęciami wspomagania tłumaczenia.
- Testowanie działania pamięci tłumaczeń w popularnych programach do wspomagania tłumaczenia
- Zaawansowane użycie pamięci tłumaczeń - ICE matching, fuzzy matching
- Implementacja modułu pamięci tłumaczeń w oparciu o bibliotekę Lucene.
- Techniki automatycznego zarządzania terminologią, w tym ekstrakcja terminologii.
- Ewaluacja narzędzi do ekstrakcji fraz i terminologii.
- Klasyfikacja dziedzinowa terminologii – techniki automatyczne i półautomatyczne.
- Tworzenie słowników dziedzinowych.
- Preprocessing i postprocessing tłumaczonych tekstów – automatyczne wstawianie elementów formatujących, konwersje dat i liczb.
- Zastosowanie wyrażeń regularnych do postprocessingu.
- Tłumaczenie automatyczne jako technika wspomagania tłumaczenia.
- Ewaluacja jakości oraz możliwości wykorzystania tłumaczenia automatycznego do wspomagania tłumaczenia.
- Web scraping - pozyskiwanie danych na potrzeby wspomagania tłumaczenia.
- Praktyczne ćwiczenia z web scrapingu.
- Urównoleglanie jako technika tworzenia pamięci tłumaczeń.
- Eksperymenty z wykorzystaniem dostępnych narzędzi do urównoleglania.
- Techniki badania wydajności procesu tłumaczenia - key logging, eye tracking.
- Uruchomienie keyloggera (Translog) i analiza wyników.
- Algorytmy automatycznej korekty pisowni.
- Ewaluacja/implementacja korektora pisowni.
- Automatyczna korekta gramatyczna tekstu.
- Ewaluacja wybranych narzędzi do korekty gramatycznej (Grammarly, MS Word)
- Projekt własnego mechanizmu wspomagania tłumaczenia.
- Implementacja i ewaluacja wybranej techniki wspomagania tłumaczenia.

Nazwa zajęć**: Kryptografia Post-Kwantowa**
Po zakończeniu zajęć i potwierdzeniu osiągnięcia efektów uczenia się student:

- Zna współczesną terminologię kryptologiczną.

- Zna model matematyczny komputera kwantowego oraz rozumie zasadę działania podstawowych algorytmów kwantowych.
- Umie analizować bezpieczeństwo protokołów kryptologicznych odpornych na ataki z wykorzystaniem komputera kwantowego.
- Potrafi ocenić zagrożenia dla bezpieczeństwa systemu informatycznego wynikające z budowy komputera kwantowego.
- Potrafi efektywnie implementować systemy kryptologiczne.
- Potrafi wykorzystać w implementacji istniejące biblioteki kryptograficzne.
- Wykorzystuje twierdzenia matematyczne w analizie systemów kryptograficznych.
- Rozumie zagrożenia wynikające z niewłaściwego wykorzystania technik kryptologicznych.
- Zna i rozumie współczesne rekomendacje systemów kryptologicznych odpornych na ataki z wykorzystaniem komputera kwantowego.
- Rozwija swoje kompetencje matematyczne w zakresie informatyki.

Treści programowe dla zajęć:

- Ciała skończone i ich implementacja.
- Krzywe eliptyczne nad ciałami skończonymi.
- Izogenie krzywych eliptycznych. Grafy izogenii.
- Protokół wymiany klucza SIDH.
- Supersingular Isogeny Key Encapsulation (SIKE).
- Aspekty implementacyjne SIKE.
- Przegląd rekomendacji NIST dotyczących systemów post-kwantowych.
- Model komputera kwantowego oraz wprowadzenie do obliczeń i algorytmów kwantowych.
- Qiskit - open source quantum development.
- Algorytm Grovera.
- Algorytm Simona.
- Kwantowe ataki na symetryczne systemy szyfrowe.
- Algorytm oszacowania fazy.
- Algorytm Shora.
- Kwantowe ataki na asymetryczne systemy szyfrowe.

Nazwa zajęć**: Kryptologia**

Po zakończeniu zajęć i potwierdzeniu osiągnięcia efektów uczenia się student:

- Zna współczesną terminologię kryptologiczną
- Potrafi wskazać wady i zalety danego rozwiązania kryptologicznego.
- Potrafi analizować bezpieczeństwo protokołów kryptologicznych.
- Potrafi obliczyć złożoność obliczeniową algorytmów wykorzystywanych do konstrukcji systemów kryptologicznych.
- Potrafi efektywnie implementować podstawowe systemy kryptologiczne.
- Potrafi wykorzystać w implementacji istniejące biblioteki kryptograficzne.
- Wykorzystuje twierdzenia matematyczne w analizie systemów kryptograficznych.
- Rozumie zagrożenia wynikające z niewłaściwego wykorzystania technik kryptologicznych.
- Zna i rozumie współczesne rekomendacje systemów kryptologicznych.
- Zna współcześnie stosowane podstawowe protokoły i rozwiązania kryptograficzne.
- Zna specyfikację standardów kryptologicznych.

Treści programowe dla zajęć:

- Wprowadzenie do kryptologii. Podstawowe protokoły kryptograficzne.
- Złożoność obliczeniowa. Funkcje jednokierunkowe.
- Symetryczne systemy szyfrowania.
- Szyfrowanie doskonałe. Szyfry blokowe.
- Advanced Encryption Standard (AES).
- Kryptoanaliza szyfrów blokowych.
- Szyfrowanie uwierzytelnione.
- Jednokierunkowe funkcje skrótu i ich zastosowanie.
- Ataki na jednokierunkowe funkcje skrótu.
- Generatory ciągów pseudolosowych. Szyfry strumieniowe.
- Kryptoanaliza szyfrów strumieniowych
  Asymetryczne systemy szyfrowania. Algorytm RSA. Szyfrowanie RSA-OAEP
- Algorytm ElGamala. Założenie Diffiego-Hellmana.

- Ataki na asymetryczne systemy szyfrowania
- Protokoły uzgadniania kluczy. Protokół Diffiego-Hellmana.
- Bibiloteka Openssl.
- Krzywe eliptyczne nad ciałem skończonym.
- Problem logarytmu dyskretnego na krzywej eliptycznej. Protokół ElGamala na krzywej eliptycznej.
- Protokół ECDH. Problem obliczeniowy DH i problem decyzyjny DH.
- Aspekty implementacyjne systemów opartych na krzywych eliptycznych.
- Rekomendacje NIST dotyczące krzywych eliptycznych.
- Schematy podpisów cyfrowych. Podpis cyfrowy RSA.
- Standard podpisu cyfrowego. Podpis ECDSA
- Protokoły związane z podpisami cyfrowymi.
- Ślepe podpisy. Kanał podprogowy.
- Protokoły uwierzytelniania. Protokół challenge and response.
- Dowody z wiedzą zerową. Protokół Schnorra.
- Certyfikaty. Infrastruktura klucza publicznego
- Protokół SSL.
- Protokół Kerberos.
- Kleptografia.

Nazwa zajęć**: Matematyczne podstawy sztucznej inteligencji i cyberbezpieczeństwa**
Po zakończeniu zajęć i potwierdzeniu osiągnięcia efektów uczenia się student:

- Zna pojęcia wektora i macierzy. Potrafi wykonywać operacje algebraiczne takie jak dodawanie i mnożenie macierzy i wektorów, mnożenie wektora przez macierz. Potrafi opisać macierz o wybranych własnościach. Zna pojęcie przestrzeni liniowej wektorów oraz liniowej niezależności i bazy wektorów.
- Rozwiązuje układy równań liniowych wielu zmiennych. Znajduje wyznaczniki z nimi stowarzyszone i potrafi określić liczbę rozwiązań w zależności od wyznacznika. Zna sposób obliczania wyznacznika oraz jego własności i potrafi znaleźć odwrotność zadanej macierzy.
- Wyznacza wielomian charakterystyczny macierzy, wartości własne. Oblicza wektory własne oraz bazę takich wektorów.
- Umie obliczyć normę, iloczyn skalarny dwóch wektorów. Potrafi obliczyć odległość dwóch punktów w metryce euklidesowej. Zna pojęcie wektorów ortogonalnych i potrafi dokonać rzutowania jednego wektora względem drugiego.
- Potrafi dokonać rozkładu macierzy (Choleskiego, Singular Value Decomposition).
- Potrafi obliczyć pochodną funkcji i wyznaczac szereg Taylora. Dla funkcji wielu zmiennych umie obliczyć pochodne cząstkowe i gradient funkcji.
- Potrafi wyznaczyć splot funkcji (konwolucję) i jego dyskretny odpowiednik.
- Umie wyznaczyć dystrybuantę i gęstość rozkładu zmiennej losowej. Zna pojęcie rozkładu warunkowego i brzegowego. Potrafi rozróżnić wybrane rozkłady prawdopodobieństwa. Zna twierdzenie Bayesa i umie zastosować je we wnioskowaniu. Potrafi obliczyć wartość oczekiwaną i wariancję zmiennej losowej.
- Potrafi zbudować model statystyczny. Zna metody estymacji punktowej i przedziałowej. Potrafi szacować parametry za pomocą metody najmniejszych kwadratów i metody największej wiarogodności.
- Potrafi testować wybrane hipotezy statystyczne.
- Zna pojęcie funkcji straty, miary jakości modelu i pojęcie regularyzacji. Umie przeprowadzić wnioskowanie bayesowskie. Zna idee algorytmów Monte Carlo Markov Chains (MCMC) oraz podstawowe próbniki.
- Potrafi zaimplementować algorytm największego spadku i zna jego podstawowe warianty: algorytm największego spadku z momentum, stochastyczny algorytm największego spadku. Zna algorytm wstecznej propagacji błędu.
- Potrafi scharakteryzować problem optymalizacji wypukłej. Zna metodą mnożników Lagrange'a.
- Zna metodę programowania liniowego (wraz z twierdzeniem o dualności) oraz kwadratowego. Potrafi wykonać algorytm sympleks oraz algorytm interior point.
- Zna pojęcie grafu i jego podstawowe własności. Potrafi opisać praktyczne przykłady grafów i wyznaczyć ich cechy charakterystyczne.

- Potrafi zamodelować propagację ataku w sieci komputerowej z uwzględnieniem jej specyficznej topologii. Umie wykorzystać techniki uczenia maszynowego do maksymalizowania wybranych parametrów bezpieczeństwa sieci.
- Potrafi wykorzystać poznany aparat pojęciowy z zakresu algebry do sformułowania i rozwiązania problemu redukcji wymiaru za pomocą metody składowych głównych. Umie ująć swoje wnioski płynące z obliczeń w postaci raportu i potrafi zaprezentować swój model w sposób popularny.
- Potrafi wykorzystać zaawansowane techniki statystyczne i analityczne do rozwiązania wybranego problemu uczenia maszynowego za pomocą sieci neuronowych. Umie przygotować dokumentację techniczną swojego modelu, umie przeprowadzić dyskusję wybranych parametrów i architektury. Raportuje swoje obserwacje i właściwie komunikuje je grupie.

Treści programowe dla zajęć:

- Wprowadzenie do algebry liniowej. Zdefiniowanie pojęcia wektora, macierzy transformacji, liniowej niezależności wektorów oraz bazy. Opisanie podstawowych typów macierzy: macierz diagonalna, górnotrójkątna. Wprowadzenie do operacji na macierzach – dodawanie, mnożenie, mnożenie macierzy przez wektor i skalar.
- Wyznaczanie macierzy z opisu transformacji liniowej. Praktyczne ćwiczenia w rachunkach na macierzach, dyskusja optymalnych metod wykonywania rachunków na macierzach wraz z praktyczną implementacją.
- Rozwiązywanie układów równań liniowych wraz z dyskusją liczby rozwiązań. Sprowadzanie macierzy do postaci całkowicie zredukowanej i wyznaczanie przestrzeni rozwiązań układu liniowego. Definicja wyznacznika oraz jego podstawowe własności. Znajdowanie odwrotności macierzy metodą eliminacji Gaussa.
- Dyskusja efektywnych metod rozwiązywania układów równań liniowych. Znajdowanie rozwiązań za pomocą algorytmu eliminacji Gaussa. Przykłady dla małej liczby zmiennych i implementacja metody w przypadku problemu o bardzo dużej liczbie zmiennych.
- Zdefiniowanie wielomianu charakterystycznego i minimalnego macierzy oraz wartości własnych. Opis algorytmu wyznaczania wektorów własnych zadanej wartości własnej. Zdefiniowanie bazy wektorów własnych zadanej przestrzeni.
- Wyznaczanie wielomianów charakterystycznych i minimalnych dla wybranych macierzy i operatorów liniowych. Implementacja algorytmu wyznaczania bazy wektorów własnych.
- Wprowadzenie pojęcia iloczynu skalarnego wektorów oraz normy wektora. Przedstawienie pojęcia metryki euklidesowej. Definicja rzutu ortogonalnego oraz projekcji.
- Obliczanie wektora ortogonalnego do zadanego wektora. Wyznaczanie rzutu ortogonalnego na zadaną podprzestrzeń liniową. Implementacja rzutowania wraz z dyskusją przykładów.
- Opis faktoryzacji macierzy metodą Cholesky'ego wraz z opisem algorytmu.
- Opis metody diagonalizacji macierzy symetrycznych.
- Wprowadzenie do rozkładu macierzy metodą SVD i porównanie z innymi metodami.
- Implementacja wybranych rozkładów (SVD, Cholesky, diagonalizacja) z wykorzystaniem opisu algorytmów z wykładu. Porównanie i dyskusja szybkości wybranych metod oraz wskazanie niestabilności numerycznej wybranych algorytmów.
- Definicja pochodnej funkcji i pochodnej cząstkowej. Wprowadzenie pojęcia szeregu Taylora dla funkcji jednej i wielu zmiennych. Definicja gradientu funkcji. Definicja splotu.
- Obliczanie pochodnych funkcji elementarnych oraz reguły obliczania pochodnych dla iloczynu, sumy i ilorazu funkcji oraz funkcji złożonej. Wyznaczanie gradientu funkcji wektorowej. Wyznaczanie splotów funkcji.
- Definicja zmiennej losowej, dystrybuanty oraz gęstości rozkładu. Zdefiniowanie rozkładu wielu zmiennych oraz rozkładu warunkowego i brzegowego. Prezentacja wybranych rozkładów prawdopodobieństwa. Omówienie twierdzenia Bayesa i jego znaczenia we wnioskowaniu. Definicja wartości oczekiwanej i wariancji zmiennej losowej, przykłady wyznaczania. Definicja kowariancji.
- Wyznaczanie rozkładów empirycznych. Porównanie rozkładu empirycznego z rozkładem teoretycznym. Wyliczanie wartości oczekiwanych i wariancji dla zmiennej losowej oraz dla próby. Porównanie uzyskanych wyników. Wyznaczanie macierzy kowariancji.
- Definicja modelu statystycznego. Przedstawienie podstaw estymacji punktowej i przedziałowej. Wprowadzenie do metod estymacji: metoda najmniejszych kwadratów i metoda największej wiarogodności. Pokazanie praktycznych przykładów zastosowań.

- Wyznaczanie estymatorów punktowych i przedziałowych z danych. Wyznaczenie parametrów w modelu regresji liniowej za pomocą metody najmniejszych kwadratów, implementacja rozwiązania. Implementacja bootstrapowych przedziałów ufności.
- Przedstawienie problemu testowania hipotez. Definicja błędu pierwszego i drugiego rodzaju. Definicja poziomu istotności testu oraz p-wartości. Omówienie przykładowych procedur testowych: test t dla jednej i dwóch prób, test niezależności oraz test zgodności. Przedstawienie idei testów permutacyjnych.
- Praktyczne testowanie hipotez dla rzeczywistych zbiorów danych. Implementacja permutacyjnego testu t.
- Definicja podstawowych funkcji straty stosowanych w uczeniu maszynowym. Omówienie związku funkcji straty z miarą jakości modelu. Wprowadzenie idei regularyzacji i jego znaczenia w optymalizacji. Wprowadzenie do statystyki bayesowskiej. Definicja rozkładu a priori i a posteriori. Wprowadzenie do algorytmów MCMC, próbnik Metropolisa-Hastingsa oraz próbnik Gibbsa.
- Implementacja różnych funkcji straty w uczeniu parametrów regresji liniowej. Implementacja próbnika Metropolisa-Hastingsa.
- Omówienie algorytmu największego spadku oraz pojęcia momentum. Omówienie stochastycznego algorytmu największego spadku. Omówienie algorytmu wstecznej propagacji błędu.
- Implementacja algorytmów największego spadku do estymacji parametrów regresji liniowej. Wykorzystanie algorytmu wstecznej propagacji błędu do uczenia parametrów sieci neuronowych.
- Zdefiniowanie problemu optymalizacji wypukłej. Omówienie metody mnożników Lagrange'a. Przedstawienie przykładów na rzeczywistych problemach.
- Wykorzystanie metody mnożników Lagrange'a do rozwiązania praktycznych problemów optymalizacji z ograniczeniami.
- Omówienie metody programowania liniowego oraz programowania kwadratowego. Podanie twierdzenia o dualności i omówienie jego konsekwencji. Przedstawienie algorytmów sympleks oraz interior point.
- Wykorzystanie algorytmów sympleks i interio point do rozwiązania praktycznych problemów optymalizacji. Implementacja algorytmu simpleks.
- Omówienie pojęcia grafu, drzewa, sieci i ich podstawowych własności.
- Wskazanie podstawowych przykładów grafów, opis topologii sieci (LAN, WAN) w terminach dużych grafów. Ćwiczenie wyznaczania podstawowych własności grafów.
- Modelowanie cyberbezpieczeństwa sieci za pomocą pojęcia grafów. Opis propagacji ataku na sieć komputerów. Modelowanie bezpieczeństwa sieci z wykorzystaniem sieci neuronowej i algorytmu uczącego opartego na uczeniu maszynowym.
- Praktyczna analiza sieci o małej liczbie wierzchołków pod kątem jej bezpieczeństwa, z uwzględnieniem wyboru topologii i metod ochrony komputerów. Dyskusja algorytmu uczącego dobierającego wagowo środki bezpieczeństwa.
- Implementacja metody składowych głównych z wykorzystaniem poznanych metod algebraicznych.
- Implementacja sieci neuronowej z wykorzystaniem poznanych metod optymalizacyjnych.

Nazwa zajęć**: Modelowanie języka**
Po zakończeniu zajęć i potwierdzeniu osiągnięcia efektów uczenia się student:

- Zna podstawowe techniki przetwarzania języka naturalnego (tokenizacja, lowercasing, itp.) i potrafi je zastosować. Potrafi „czyścić" dane tekstowe. Rozumie sposób reprezentacji tekstu w komputerze i problemy związane z wysoką wymiarowością reprezentacji liczbowej tekstu.
- Zna definicję modelu językowego, sposobów jego ewaluacji i zastosowania.
- Potrafi wytłumaczyć sposób działania statystycznego modelu językowego. Rozumie pojęcie n-gramu. Rozumie sposób trenowania statystycznego modelu językowego.
- Potrafi stworzyć statystyczny model językowy bez użycia gotowych bibliotek. Potrafi stworzyć statystyczny model językowy przy pomocy gotowych bibliotek. Umie ewaluować modele językowe.
- Rozumie podstawy działania sieci neuronowych. Potrafi stworzyć prostą sieć neuronową przy wykorzystaniu biblioteki języka programowania.

- Zna modele typu „word2vec”. Potrafi wskazać zastosowania tych modeli, ich zalety i wady. Zna pojęcie reprezentacji numerycznej tekstu („embedding”) i potrafi wyjaśnić różnice reprezentacji rzadkiej i gęstej.
- Potrafi wykorzystywać gotowe modele typu „word2vec”.
- Rozumie sposób działania neuronowego modelu językowego opartego o prostą sieć „feed-forward”.
- Potrafi zaimplementować i wytrenować prosty model językowy oparty o sieć „feed-forward”.
- Rozumie sposób działania rekurencyjnego neuronowego modelu językowego. Zna mechanizm atencji.
- Potrafi zaimplementować i wytrenować rekurencyjny neuronowy model językowy.
- Rozumie sposób działania neuronowego modelu językowego opartego o sieć typu „transformer”.
- Potrafi zaimplementować i wytrenować model językowy oparty o sieć typu „transformer” przy użyciu gotowych bibliotek.
- Zna różne algorytmy trenowania modelu językowego typu transformer („masked language model”, „next sentence prediction”). Zna pojęcia „pretraining”, „fine-tuning”, „transfer learning”.
- Potrafi wykorzystać wytrenowany model językowy typu „transformer” dla problemu klasyfikacji.
- Zna alternatywne sposoby ewaluacji neuronowych modeli językowych na podstawie benchmarków „GLUE” i „SuperGLUE”. Potrafi wskazać różnice pomiędzy modelami językowymi typu „transformer”: „BERT”, „RoBERTa”, rodzina modeli „GPT”.
- Potrafi generować tekst przy użyciu modelu językowego typu transformer.

Treści programowe dla zajęć:

- Wprowadzenie do przetwarzania języka naturalnego. Przedstawienie podstawowych metod przetwarzania języka naturalnego. Zaznajomienie z problemami wynikającymi z pracy z tekstem.
- Zastosowanie metod przetwarzania tekstu w praktyce na podstawie implementacji klasyfikatora tekstu opartego o TF-IDF i regresję logistyczną.
- Modele językowe i ich ewaluacja
- Statystyczne modele językowe – sposób działania, trenowania, zalety i wady.
- Trenowanie statystycznego modelu językowego bez użycia gotowych bibliotek. Ewaluacja wytrenowanego modelu językowego
- Trenowanie statystycznego modelu językowego z wykorzystaniem biblioteki „KenLM”.
- Wprowadzenie do sieci neuronowych.
- Implementacja prostej sieci neuronowej przy wykorzystaniu biblioteki „pytorch”
- Wprowadzenie modeli typu „word2vec” – algorytm trenowania, zalety, wady.
- Wykorzystanie modeli typu „word2vec” w praktyce.
- Modele językowe oparte o sieć „feed-forward”.
- Implementacja modelu językowego opartego o sieć „feed-forward”.
- Modele językowe oparte o rekurencyjne sieci neuronowe, sieci typu „GRU” i „LSTM”. Mechanizm atencji.
- Implementacja i trenowanie modelu językowego opartego o sieć rekurencyjną przy użyciu biblioteki „pytorch”. Implementacja i trenowanie modelu językowego opartego o sieć rekurencyjną z mechanizmem atencji przy użyciu biblioteki „pytorch”.
- Modele językowe oparte o sieci neuronowe, sieci typu transformer – wprowadzenie.
- Implementacja i trenowanie modelu językowego opartego o sieć typu „transformer” przy użyciu biblioteki „pytorch”.
- Modele językowe oparte o sieci neuronowe, sieci typu transformer – metody zaawansowane, zastosowania.
- Klasyfikacja tekstu za pomocą sieci typu „transformer” przy wykorzystaniu biblioteki fairseq.
- Modele językowe oparte o sieci neuronowe typu transformer- ewaluacja, alternatywne sposoby ewaluacji modeli. Omówienie różnych modeli typu „transformer” („BERT”, „RoBERTa”, rodzina „GPT”).
- Generowanie tekstu przy użyciu modelu językowego GPT2.

Nazwa zajęć**: Narzędzia matematyczne sztucznej inteligencji i cyberbezpieczeństwa**
Po zakończeniu zajęć i potwierdzeniu osiągnięcia efektów uczenia się student:

- Potrafi zbudować model matematyczny prostych zjawisk rzeczywistych przy pomocy narzędzi teorii grafów lub skończonych łańcuchów Markowa.

- Potrafi dostrzec związki między pojęciami algebry liniowej (np. wektorami i wartościami własnymi macierzy) a własnościami grafowymi.
- Umie przewidzieć wynik symulacji procesów Markowa, korzystając z twierdzeń ergodycznych.
- Umie przedstawiać tok swojego rozumowania w sposób zrozumiały dla słuchaczy.

Treści programowe dla zajęć:

- Interpretacja zjawisk rzeczywistych w języku teorii grafów. Macierz przyległości grafu (prostego), wartości i wektory własne tej macierzy. Własności spektrum macierzy symetrycznych.
- Odczytywanie własności grafowych na podstawie spektrum macierzy przyległości, np. informacji o spójności, regularności, dwudzielności, cyklach i spacerach zadanej długości.
- Laplasjan grafu. Metody spektralne szacowania trudnych obliczeniowo parametrów grafowych: liczby niezależności, liczby chromatycznej, największego cięcia w grafie. Wyznaczanie liczby drzew rozpiętych metodą algebraiczną.
- Definicja (skończonego) łańcucha Markowa. Macierz przejścia, skierowany graf łańcucha Markowa. Algebraiczne i probabilistyczne metody wyznaczania rozkładu łańcucha Markowa po zadanej liczbie kroków.
- Symulacja wielu kroków łańcucha Markowa. Rozkłady stacjonarne, twierdzenie ergodyczne.
- Odwracalne łańcuchy Markowa. Błądzenie na grafach, czas pokrycia. Generowanie struktur losowych metodą Monte Carlo na łańcuchach Markowa (MCMC).
- Zastosowania łańcuchów Markowa i MCMC do łamania szyfrów podstawieniowych, generowania tekstów, generowania muzyki, tworzenia rankingu Page'a.

Nazwa zajęć**: Podstawy bezpieczeństwa komputerowego**

Po zakończeniu zajęć i potwierdzeniu osiągnięcia efektów uczenia się student:

- Rozumie rolę zabezpieczeń fizycznych.
- Zna popularne produkty lub rozwiązania służące do zwiększenia bezpieczeństwa fizycznego.
- Zna popularne wektory ataku na infrastrukturę lokalną.
- Zna popularne wektory ataku na infrastrukturę zdalną.
- Zna popularne produkty lub rozwiązania służące do wykrywania podatności i zapobiegania ich powstawaniu.
- Potrafi zastosować popularne produkty lub rozwiązania służące do wykrywania podatności.
- Potrafi wykorzystać wykrytą podatność dla dalszej penetracji, destabilizacji systemu, przejęcia kontroli nad systemem lub pozyskania danych.
- Potrafi prowadzić przegląd kodu w celu wyeliminowania faktycznych i potencjalnych podatności.
- Potrafi pozyskiwać wiedzę na temat aktualnych zagrożeń i mechanizmów zapobiegania im.
- Rozumie potrzebę ustawicznego podnoszenia wiedzy i umiejętności z zakresu bezpieczeństwa komputerowego.
- Rozumie kulturowe i socjologiczne uwarunkowania bezpieczeństwa komputerowego.

Treści programowe dla zajęć:

- Przedstawienie układu treści kursu. Wprowadzenie podstawowej terminologii. Charakteryzacja “bezpieczeństwa” jako szerokiego wachlarza zagadnień technicznych i nietechnicznych.
- Linux i pisanie skryptów w powłoce systemu Linux. Doświadczalne zidentyfikowanie poziomu wiedzy i umiejętności uczestników w tym zakresie. Samodzielne uzupełnienie lub rozszerzenie wiedzy i umiejętności w oparciu o wskazane materiały.
- Określenie konfiguracji eksperymentów
- na podstawie deklaracji uczestników co do ich wiedzy, umiejętności i doświadczenia w zakresie konkretnych języków programowania, bibliotek, narzędzi, itd.,
- Identyfikacja i zapobieganie podatnościom. Najbardziej rozpoznawalne grupy problemów. Symptomy podatności i standardowe rozwiązania.
- Przeprowadzenie eksperymentów: Wykrycie i wykorzystanie podatności. Praca w izolowanym i kontrolowanym środowisku, w którym podatności zostały celowo umieszczone. Samodzielne uzupełnienie lub rozszerzenie wiedzy i umiejętności w zakresie narzędzi lub technologii występujących w omawianym środowisku.
- Przegląd narzędzi do wykrywania podatności. Omówienie popularnych rozwiązań dla danego zastosowania.
- Przeprowadzenie eksperymentów związanych z wykrywaniem podatności.
- Systemy kontroli dostępu w zabezpieczaniu infrastruktury: Wyzwania. Przegląd produktów i rozwiązań.

- Przeprowadzenie eksperymentów związanych z systemami kontroli dostępu w zabezpieczaniu infrastruktury.
- Systemy monitoringu infrastruktury: Wyzwania. Przegląd produktów i rozwiązań.
- Przeprowadzenie eksperymentów związanych z monitoringiem infrastruktury.
- Systemy wykrywania wtargnięć i ich raportowanie: Przegląd rozwiązań.
- Przeprowadzenie eksperymentów związanych z wykrywaniem wtargnięć.
- Zabezpieczenia maszyn lokalnych. Ustanowienie warstwowości zabezpieczeń. Specyfika czynności administracyjnych.
- Przeprowadzenie eksperymentów związanych z zabezpieczeniem maszyn lokalnych.
- Ochrona dostępu zdalnego. Implementacja lokalnych środków zapobiegawczych. Utwardzanie.
- Przeprowadzenie eksperymentów związanych z ochroną dostępu zdalnego.
- Budowa sieci i protokoły sieciowe. Wykorzystanie niezgodnie z przeznaczeniem.
- Przeprowadzenie eksperymentów związanych z ochroną protokołów sieciowych.
- Konfiguracja serwerów. Dobre praktyki czynności administracyjnych.
- Przeprowadzenie eksperymentów związanych z konfiguracją serwerów.
- Urządzenia sieciowe i media transmisyjne. Perspektywa bezpieczeństwa.
- Przeprowadzenie eksperymentów związanych z urządzeniami sieciowymi i mediami transmisyjnymi.
- Ochrona sieci prywatnej. Perimeter security.
- Przeprowadzenie eksperymentów związanych z ochroną sieci prywatnej.
- Ochrona transferu przez sieć publiczną. Metody i narzędzia.
- Przeprowadzenie eksperymentów związanych z ochroną transferu przez sieć publiczną.
- Kulturowe spojrzenie na bezpieczeństwo. Ludzie, organizacje, styl życia i pracy.
- Przeprowadzenie eksperymentów związanych z kulturowym spojrzeniem na bezpieczeństwo.
- Przygotowanie do egzaminów końcowych.
- Podsumowanie kursu.

Nazwa zajęć**: Polityka cyberbezpieczeństwa**
Po zakończeniu zajęć i potwierdzeniu osiągnięcia efektów uczenia się student:

- Zna podstawowe pojęcia oraz zjawiska z zakresu bezpieczeństwa i cyberbezpieczeństwa. Potrafi wskazać cechy cyberprzestrzeni.
- Potrafi wskazać przyczyny, przejawy oraz następstwa rewolucji informatycznej. Rozumie ich kontekst społeczny oraz technologiczny, a także powiązania ze sferą bezpieczeństwa.
- Zna ,,miękkie'' wyzwania i zagrożenia dla polityki cyberbezpieczeństwa. Potrafi wytłumaczyć formy i przykłady ich stosowania.
- Rozumie istotę ,,twardych'' wyzwań i zagrożeń dla polityki bezpieczeństwa. Potrafi scharakteryzować ich przykłady, w tym wskazać znaczenie czy następstwa.
- Zna formy i metody zwalczania cyberzagrożeń. Potrafi ocenić ich skuteczność wskazując mocne i słabe strony. Umie prognozować nowe potencjalne zagrożenia.

Treści programowe dla zajęć:

- Wprowadzenie do polityki bezpieczeństwa i cyberbezpieczeństwa.
- Rewolucja informatyczna i główne jej tendencje.
- ,,Miękkie'' zagrożenia i wyzwania dla polityki cyberbezpieczeństwa (np. cyfrowa przepaść, cyberszpiegostwo, hacking itp.).
- ,,Twarde'' zagrożenia i wyzwania dla polityki cyberbezpieczeństwa (np. cyberwojny, cyberterroryzm itp.).
- Zwalczanie cyberzagrożeń oraz prognoza ich przyszłej ewolucji

Nazwa zajęć**: Praktyczne zastosowania chmury obliczeniowej**
Po zakończeniu zajęć i potwierdzeniu osiągnięcia efektów uczenia się student:

- Zna podstawowy obliczeń w chmurze
- Rozumie hierarchię modeli chmury obliczeniowej
- Rozumie model IaaS chmury obliczeniowej
- Rozumie model PaaS chmury obliczeniowej
- Rozumie model FaaS/Serverless chmury obliczeniowej
- Rozumie model SaaS chmury obliczeniowej,
- Zna typy chmury obliczeniowej
- Zna dostawców chmury w modelu PaaS

- Potrafi wykorzystać interfejs CLI do konfiguracji zasobów chmury
- Potrafi przygotować i wdrożyć prostą aplikację w modelu PaaS
- Potrafi zautomatyzować proces wdrażania systemu w modelu PaaS
- Zna dostawców chmury w modelu IaaS
- Potrafi skonfigurować sieć prywatną w chmurze
- Potrafi skonfigurować zasoby dyskowe w chmurze
- Potrafi obsługiwać narzędzie cloud-init
- Potrafi wykorzystać chmurę do obliczeń na kartach graficznych (GPU)
- Rozumie znaczenie kompleksowej publicznej chmury obliczeniowej
- Potrafi wykorzystać interfejs webowy do konfiguracji zasobów chmury
- Potrafi skonfigurować system równoważenia obciążenia
- Potrafi skonfigurować automatyczne skalowanie infrastruktury w chmurze
- Zna usługi dostępne w modelu SaaS
- Potrafi wykorzystać zarządzaną w modelu SaaS bazę danych w chmurze
- Zna dostępne usługi w modelu FaaS
- Potrafi zaimplementować webserwis w modelu Serverless
- Rozumie potrzebę ujednolicenia i automatyzacji procesu przydzielania zasobów (IaaC)
- Potrafi wykorzystać chmurę obliczeniową do budowy złożonego systemu informatycznego
- Potrafi wykorzystać chmurę do obliczeń wymagających dużej ilości zasobów
- Potrafi przeprowadzić analizę kosztów infrastruktury chmurowej w różnych modelach

Treści programowe dla zajęć:

- Wprowadzenie do obliczeń w chmurze, hierarchia modeli chmury obliczeniowej: IaaS, PaaS, FaaS, SaaS, dostawcy i typy chmury.
- Przedstawienie modelu Platform as a Service (PaaS), umówienie dostawców (np. Heroku), wykorzystanie interfejsu CLI do konfiguracji zasobów, przygotowanie i wdrożenie prostej aplikacji w modelu PaaS, automatyzacja procesu wdrażania poprzez integrację z Git.
- Przedstawienie modelu Infrastructure as a Service (IaaS), omówienie dostawców (np. Hetzner, Linode), wykorzystanie CLI do konfiguracji zasobów, konfiguracja sieci prywatnej, zasobów dyskowych, obsługa narzędzia cloud-init, wykorzystanie chmury do obliczeń na kartach graficznych (GPU).
- Omówienie kompleksowej publicznej chmury obliczeniowej (np. AWS, Azure), konfiguracja z wykorzystaniem przeglądarki: sieci prywatnej, zasobów dyskowych, systemu równoważenia obciążenia (load balancer), automatycznego skalowania infrastruktury.
- Przedstawienie modelu Software as a Service (SaaS), umówienie dostępnych usług, omówienie modelu na podstawie zarządzanej bazy danych w chmurze (np. AWS DynamoDB).
- Przedstawienie modelu Serverless/Function as a Service (FaaS), omówienie dostępnych usług (np. AWS Lambda, Azure Functions), implementacja REST API z modelu Serverless.
- Projekt obejmujący przygotowanie złożonego systemu informatycznego wdrożonego w całości w chmurze i korzystającego z pełni jej możliwości. Omówienie problemu ujednolicania i automatyzacji procesu przydzielania zasobów (Infrastructure as a Code, IaaC).

Nazwa zajęć**: Zabezpieczenia protokołów sieciowych**

Po zakończeniu zajęć i potwierdzeniu osiągnięcia efektów uczenia się student:

- Rozumie pojęcia związane z transmisją danych w sieciach komputerowych opartych o model TCP/IP.
- Potrafi używać wiersz poleceń systemu Cisco IOS.
- Potrafi użyć protokoły Telnet i SSH do zdalnego zarządzania sprzętem sieciowym oraz zna ich wady i zalety.
- Rozumie problemy związane z użyciem dużej liczby komputerów w jednej domenie rozgłoszeniowej i potrafi je rozwiązać stosując VLANy.
- Potrafi zabezpieczyć urządzenia przed atakiem zalewania adresami MAC.
- Zna problemy związane z użyciem Dynamic Trunking Protocol i potrafi im zapobiegać.
- Rozumie niebezpieczeństwo pętli w sieci komputerowej i umie konfigurować Spanning Tree Protocol.
- Potrafi wdrożyć redundancję łącza.
- Zna pojęcie routingu w sieciach komputerowych.
- Potrafi użyć protokół OSPF i zabezpieczyć go.
- Potrafi użyć protokół EIGRP i zabezpieczyć go.

- Potrafi wdrożyć redundancję bramy sieciowej przy użyciu protokołu HSRP.
- Potrafi przeciwdziałać atakom związanym z protokołem DHCP.
- Zna metody filtrowania niepożądanego ruchu sieciowego przy użyciu list dostępu.
- Potrafi uruchomić szyfrowany tunel pomiędzy urządzeniami obsługującymi ruch sieciowy.

Treści programowe dla zajęć:

- Omówienie pojęć związanych z transmisją danych w sieciach komputerowych.
- Zapoznanie ze sprzętem sieciowym. Podstawy obsługi systemu operacyjnego Cisco IOS.
- Konfiguracja protokołów zdalnego zarządzania: Telnet i SSH.
- Zagrożenia związane z dużą domeną rozgłoszeniową. Konfiguracja wirtualnej sieci lokalnej.
- Zabezpieczanie portów przy użyciu Port Security.
- Omówienie Dynamic Trunking Protocol i zagrożeń z nim związanych.
- Spanning Tree Protocol jako rozwiązanie problemu przypadkowych pętli.
- Zabezpieczanie sieci lokalnej przed awarią pojedynczego łącza.
- Podstawy routingu.
- Routing dynamiczny na przykładzie OSPF. Zabezpieczanie protokołu OSPF.
- Routing dynamiczny na przykładzie EIGRP. Zabezpieczanie protokołu EIGRP.
- Redundancja bramy sieciowej.
- Przeciwdziałanie atakom związanym z protokołem DHCP.
- Filtrowanie niepożądanego ruchu sieciowego przy użyciu list dostępu.
- Szyfrowanie ruchu w tunelu sieciowym.

Nazwa zajęć**: Zaawansowane algorytmy rozproszone**

Po zakończeniu zajęć i potwierdzeniu osiągnięcia efektów uczenia się student:

- Rozumie w jaki sposób modeluje się obliczenia w sieciach różnego typu
- Potrafi twórczo rozwinąć znane algorytmy i/lub znaleźć dla nich nowe zastosowania algorytmiczne
- Zna i rozumie zaawansowane techniki projektowania algorytmów rozproszonych, złożone z wielu faz
- Zna metody ułatwiające tworzenie algorytmów asynchronicznych
- Potrafi się posługiwać narzędziem do automatycznej analizy algorytmów rozproszonych/ współbieżnych
- Potrafi się posługiwać symulatorem algorytmów rozproszonych w celu eksperymentów, analizy i testowania własnych algorytmów

Treści programowe dla zajęć:

- Modelowanie obliczeń rozproszonych w sieciach (model z przesyłaniem komunikatów); wersja synchroniczna i asynchroniczna; pojęcia: konfiguracja, zdarzenie, egzekucja; złożoność komunikatowa i czasowa algorytmów; implementowanie algorytmów rozproszonych w prostym symulatorze; motywacja problemów grafowych w sieciach
- Problem wyboru lidera (LE) w cyklu, wersje o złożoności komunikatowej O(n^2) i O(n logn); różne uogólnienia tych problemów; implementacja w symulatorze
- Obliczanie 3-kolorowania wierzchołków drzewa metodą Cole&Vishkin oraz różne uogólnienia i zastosowania tego algorytmu; implementacja w symulatorze; dolne oszacowanie na 2 i 3-kolorowanie wierzchołkowe cyklu
- Obliczanie skojarzeń w grafie metodą skojarzenia frakcyjnego i deterministycznego zaokrąglania (algorytm M. Fischer)
- Narzędzia do budowania algorytmów asynchronicznych: synchronizatory alfa, beta i gamma; implementacja w symulatorze
- Algorytmy odporne na błędy sieci: problem Consensus; implementacja w symulatorze
- Narzędzie do automatycznej weryfikacji algorytmów rozproszonych: spin i język promela; zastosowanie tego narzędzia do analizy własnych algorytmów

Nazwa zajęć**: Zaawansowane algorytmy kombinatoryczne**

Po zakończeniu zajęć i potwierdzeniu osiągnięcia efektów uczenia się student:

- Rozumie praktyczne znaczenie algorytmów kombinatorycznych.
- Zna podstawowe obiekty kombinatoryczne i sposoby ich reprezentacji w komputerze.
- Potrafi konstruować algorytmy generowania podstawowych obiektów kombinatorycznych w określonym porządku.

- Potrafi konstruować algorytmy pozycyjne i antypozycyjne dla prostych obiektów kombinatorycznych.
- Potrafi kodować i rozkodowywać drzewa, używając kodu Prüfera.
- Potrafi generować losowe drzewa, grafy nieskierowane i skierowane.
- Zna i rozumie pojęcie sieci przepływowych, potrafi wykorzystać je do modelowania zjawisk przepływowych.
- Potrafi rozwiązać problem maksymalnego przepływu i zna jego zastosowania.
- Rozumie pojęcie NP-trudności, zna przykładowe problemy NP-trudne.
- Rozumie pojęcie algorytmu aproksymacyjnego i potrafi konstruować algorytmy aproksymacyjne dla problemów optymalizacji kombinatorycznej.
- Potrafi krytycznie ocenić skonstruowany algorytm, analizując jego poprawność i złożoność czasową.
- Zna podstawowe metaheurystyki.
- Potrafi konstruować algorytmy metaheurystyczne dla problemów optymalizacji kombinatorycznej.
- Potrafi przeprowadzić analizę eksperymentalną wydajności zaimplementowanych algorytmów.
- Potrafi konstruować algorytmy rozwiązujące problemy trudne obliczeniowo, wykorzystując technikę programowania dynamicznego.
- Potrafi konstruować algorytmy rozwiązujące problemy trudne obliczeniowo, wykorzystując technikę podziału i ograniczeń.
- Zna problem wyszukiwania wzorca i potrafi rozwiązać go, wykorzystując proste algorytmy.
- Potrafi rozwiązać problem wyszukiwania wzorca, wykorzystując automat skończony.
- Potrafi rozwiązać problem wyszukiwania wzorca w czasie liniowym.

Treści programowe dla zajęć:

- Proste obiekty kombinatoryczne: ciągi, zbiory, permutacje. Porządek leksykograficzny, porządek minimalnych zmian. Algorytmy pozycyjne i antypozycyjne.
- Generowanie podzbiorów o ustalonej liczbie elementów. Porządek antyleksykograficzny.
- Kod Prüfera. Generowanie losowych drzew i grafów.
- Sieci przepływowe. Problem maksymalnego przepływu. Metoda Forda-Fulkersona.
- Zastosowania problemu maksymalnego przepływu. Skojarzenia w grafach dwudzielnych.
- Problemy NP-trudne. Algorytmy aproksymacyjne.
- Metaheurystyki: przeszukiwanie lokalne, iteracyjne przeszukiwanie lokalne.
- Metaheurystyki: przeszukiwanie zmiennego sąsiedztwa.
- Metaheurystyki: algorytmy genetyczne.
- Implementacja i porównanie wybranych metaheurystyk dla zadanego problemu optymalizacyjnego.
- Algorytmy dokładne dla problemów trudnych obliczeniowo: programowanie dynamiczne.
- Algorytmy dokładne dla problemów trudnych obliczeniowo: metoda podziału i ograniczeń.
- Algorytmy wyszukiwania wzorca: podejście naiwne, algorytm Rabina-Karpa.
- Algorytmy wyszukiwania wzorca: wykorzystanie automatów skończonych.
- Algorytmy wyszukiwania wzorca: algorytm Knutha-Morrisa-Pratta.

Nazwa zajęć**: Wykrywanie Incydentów**

Po zakończeniu zajęć i potwierdzeniu osiągnięcia efektów uczenia się student:

- Potrafi określić cechy incydentu bezpieczeństwa.
- Zna metodyki zarządzania incydentami i przedstawić ich etapy.
- Potrafi przygotować raport opisujący incydent bezpieczeństwa.
- Potrafi wykorzystać wywiad zagrożeń do wykrywania incydentów.
- Potrafi odwzorować incydent wykorzystując modele zagrożeń.
- Zna metody i technologie wykorzystywane do wykrywania ataków cybernetycznych.
- Potrafi scharakteryzować technologię SIEM i jej zastosowanie w wykrywaniu incydentów.
- Potrafi zidentyfikować ataki na zewnętrzne usługi sieciowe.
- Potrafi rozpoznać i dokonać analizy ataków typu phishing.
- Potrafi przeprowadzić podstawową analizę złośliwych dokumentów elektronicznych.
- Rozumie techniki rekonesansu i techniki ich wykrywania.
- Rozumie charakterystykę ruchu Command & Control i możliwości jego identyfikacji.
- Potrafi zidentyfikować próby eskalacji uprawnień.
- Potrafi przedstawić metody utrwalania dostępu i sposoby na ich rozpoznanie.

- Potrafi przedstawić przykłady technik ruchu poprzecznego i metody ich detekcji.
- Zna techniki eksfiltracji danych i metody ich wykrycia.
- Zna charakterystykę ataków w środowisku chmurowym i możliwości ich identyfikacji.

Treści programowe dla zajęć:

- Wprowadzenie do wykrywania incydentów bezpieczeństwa: Charakterystyka incydentu bezpieczeństwa. Porównanie metodyk zarządzania incydentami.
- Modelowanie i wywiad zagrożeń: Wykrywanie incydentów w oparciu o wywiad zagrożeń oraz wykorzystanie modeli „Cyber Kill Chain" i Mitre „Att&ck" do przedstawienia cyklu ataku cybernetycznego.
- Metody i systemy wykrywania ataków: Zapoznanie z podstawowymi metodami wykrywania ataków. Porównanie możliwości systemów IDS, EDR, WAF oraz UBA.
- Technologia SIEM: Zapoznanie się z możliwościami wykorzystania rozwiązań SIEM do identyfikacji ataków cybernetycznych.
- Ataki na perymetr sieci:Przegląd wybranych technik ataków na zewnętrzne usługi sieciowe i aplikacje webowe oraz ich detekcji.
- Ataki typu phishing i złośliwe dokumenty: Analiza wybranych wariantów ataków wykorzystujących pocztę elektroniczną i złośliwe dokumenty elektroniczne.
- Rekonesans: Wykrywanie aktywności cyberprzestępców związanych z rozpoznaniem otoczenia skompromitowanych systemów informatycznych.
- Ruch Command & Control: Poznanie wybranych metod komunikacji z serwerami C2 i opracowanie sposobu ich identyfikacji.
- Eskalacja uprawnień: Przedstawienie metod podnoszenia uprawnień oraz ich detekcji.
- Techniki utrwalania dostępu: Prezentacja sposobów na zapewnienie trwałego dostępu do skompromitowanych systemów i ich identyfikację.
- Techniki ruchu poprzecznego: Omówienie kluczowych technik poruszania się po skompromitowanej sieci oraz generowanych przez nie śladów.
- Eksfiltracja danych: Poznanie technik eksfiltracji danych oraz metod ich detekcji.
- Ataki w chmurze: Metody wykrywania incydentów w środowisku chmurowym.
- Podsumowanie kursu

Nazwa zajęć**: Wykrywanie ataków sieciowych**

Po zakończeniu zajęć i potwierdzeniu osiągnięcia efektów uczenia się student:

- Zna w pogłębionym stopniu problematykę sieci komputerowych.
- Potrafi przygotować złożony warsztat pracy z wykorzystaniem maszyn wirtualnych i dostępnego oprogramowania.
- Potrafi rozróżniać różne typy pasywnych zabezpieczeń sieciowych i zna podstawy konfiguracji sieci.
- Zna problematykę zabezpieczeń aktywnych.
- Rozumie podstawowe zasady działania protokołów HTTP, SPDY oraz QUIC. Rozumie problemy związane z wydajnym wykorzystaniem sieci komputerowych do udostępniania treści.
- Rozumie idee i sposoby zabezpieczeń serwerów http. Potrafi analizować logi bezpieczeństwa wybranego serwera.
- Potrafi przeprowadzić rozpoznanie sieci z wykorzystaniem zarówno narzędzi kontaktowych, jak i bezkontaktowych.
- Zna zaawansowane aspekty konfiguracji systemów IPS.
- Zna i potrafi stosować wybrane oprogramowanie do przeprowadzania testów penetracyjnych.
- Zna wybrane, zaawansowane metody pozyskiwania wiedzy o atakach sieciowych
- Zna podział i powody ataków na systemy komputerowe.
- Zna narzędzia wyszukiwania błędów w oprogramowaniu.
- Zna narzędzia i potrafi tworzyć reguły wybranych narzędzi ochrony antywirusowej i analizy plików.
- Zna narzędzia i potrafi tworzyć reguły wybranych narzędzi ochrony antyspamowej.
- Potrafi pisać reguły systemów wykrywania i przeciwdziałania atakom.
- Zna i rozumie wybrane parametry sprzętu i systemu operacyjnego, istotne w procesie wydajnego wykrywania zagrożeń.
- Zna wybrane narzędzia służące do phishingu i ochrony przed phishingiem, rozumie ich działanie i ograniczenia.
- Zna i rozumie problem podszywania się w sieci.

- Zna i rozumie miary skuteczności systemów oraz granice ich wydajności.

Treści programowe dla zajęć:

- Krótkie wprowadzenie do sieci komputerowych, protokół TCP/IPv4 i TCP/IPv6 (TCP/UDP/ICMP). Przypomnienie i uszczegółowienie wiedzy zdobytej na studiach inżynierskich.
- Przygotowanie środowiska pracy, instalacja oprogramowania na maszynach wirtualnych
- Zapory sieciowe, serwery proxy (rodzaje, funkcjonalność, zasady działania i możliwości), lokalizacja sensorów (problemy), ukrywanie zabezpieczeń
- Systemy pasywne i aktywne (IDS, IPS), przykładowe systemy (Snort – Cisco, Suricata i inne), programy „antywirusowe" (wprowadzenie), wymagania dla systemów IPS
- Protokół http – problemy z protokołem, nowe pomysły (SPDY, QUIC), wpływ na bezpieczeństwo
- Zabezpieczanie serwerów HTTP (reguły ModSecurity, czytanie logów)
- Metody skanowania sieci (narzędzia bezkontaktowe i kontaktowe)
- Konfiguracja Snort/Suricata, preprocesory, interpretacja wyników programów
- Narzędzia do przeprowadzania testów penetracyjnych, etapy ataku, narzędzia, wykorzystanie Metasploita, przykładowe ataki
- Sposoby pozyskiwania danych o atakach (honypots, podejrzane zachowania, anomalie), sygnatury ataków
- Podstawowe powody włamań i naruszeń bezpieczeństwa
- Wykrywanie problemów w oprogramowaniu (podstawy np. Valgrinda), zabezpieczenia sprzętowe systemów operacyjnych
- Narzędzia analizy przesyłanych danych (np. program ClamAV i połączenie go z IPS, pisanie reguł do ClamAV), Safe Browsing.
- Narzędzia wykrywania i analizy SPAMu, techniki
- Pisanie reguł systemów IPS, dodatkowe źródła reguł
- Techniczne aspekty wykrywania ataków (ustawienia systemu operacyjnego, kart sieciowych, przechwytywania pakietów), Algorytmy wyszukiwania wzorców (w łańcuchach).
- Ataki phishingowe (narzędzia do organizacji ataków i sposoby wykrywania)
- Spoofing, projekty typu HoneyPot, skanowanie portów i ochrona
- Miary skuteczności systemów wykrywających ataki, próby wykrywania ataków metodami sztucznej inteligencji, ograniczenia systemów wykrywania

Nazwa zajęć**: Wizualizacja danych**

Po zakończeniu zajęć i potwierdzeniu osiągnięcia efektów uczenia się student:

- Zna podstawowe typy wykresów wykorzystywane w analizie danych. Potrafi przygotować wykresy korzystając z języka R.
- Potrafi wykorzystać bibliotekę ggplot2 do przygotowania wykresów statycznych.
- Potrafi wykorzystać poznane wykresy do sformułowania i wizualizacji problemu analizy rzeczywistego zbioru danych za pomocą biblioteki ggplot2. Umie wyciągnąć wnioski płynące z wizualizacji i przedstawić je w postaci raportu w sposób popularny.
- Potrafi wykorzystać różne biblioteki do tworzenia wykresów interaktywnych.
- Potrafi prezentować dane na mapach.
- Potrafi wykorzystać poznane biblioteki do formułowania i wizualizacji problemu analizy rzeczywistego zbioru danych. Umie przedstawić swoje wnioski płynące z wizualizacji w postaci raportu i zaprezentować je w sposób popularny.
- Zna podstawowe typy dashboardów. Potrafi je przygotować z wykorzystaniem biblioteki shiny.
- Potrafi wykorzystać poznane biblioteki do tworzenia dashboardów. Umie przedstawić swoje wnioski płynące z wizualizacji w postaci raportu i zaprezentować je w sposób popularny.

Treści programowe dla zajęć:

- Podstawowe typy wykresów wykorzystywane w analizie danych.
- Biblioteka ggplot2.
- Przygotowanie projektu wykorzystującego możliwości biblioteki ggplot2. Przygotowanie prezentacji omawiającej wyniki projektu wykorzystującego bibliotekę ggplot2. Prezentacja projektów wykorzystujących bibliotekę ggplot2.
- Wykresy interaktywne. Biblioteki: plotly, NVD3, MorrisJS.
- Przedstawianie danych na mapach. Biblioteki: leaflet oraz maps.
- Przygotowanie projektu prezentującego możliwości interaktywnej wizualizacji danych oraz przedstawiania danych na mapach. Przygotowanie prezentacji omawiającej wyniki projektu

wykorzystującego wykresy interaktywne oraz mapy. Prezentacja projektów wykorzystujących wykresy interaktywne oraz mapy.
- Dashboardy. Biblioteka shiny.
- Przygotowanie projektu prezentującego wyniki raportów za pomocą dashboardów. Przygotowanie prezentacji omawiającej projekty wykorzystujące dashboardy. Prezentacja projektów wykorzystujących dashboardy.

Nazwa zajęć**: Widzenie komputerowe**
Po zakończeniu zajęć i potwierdzeniu osiągnięcia efektów uczenia się student:
- Zna tematykę badawczą dziedziny widzenia komputerowego. Potrafi przygotować środowisko programistyczne do przetwarzania obrazów i wideo oraz wykonywać w nim podstawowe operacje.
- Potrafi stosować algorytmy binaryzacji na obrazach.
- Potrafi stosować wzmacnianie i filtrowanie obrazów.
- Potrafi stosować zaawansowane algorytmy przetwarzania obrazów i fotografii obliczeniowej.
- Potrafi wykonywać transformacje geometryczne i operować cechami obrazów.
- Potrafi dokonywać segmentacji i rozpoznawania obrazów.
- Potrafi analizować materiały wideo w celu śledzenia obiektów.
- Potrafi rozpoznawać twarze na obrazach.
- Potrafi wykrywać i rozpoznawać tekst na obrazach.
- Zna i potrafi stosować metody uczenia głębokiego w zagadnieniach widzenia komputerowego.

Treści programowe dla zajęć:
- Wprowadzenie do widzenia komputerowego: przygotowanie środowiska programistycznego OpenCV i przegląd modułów; podstawowe operacje na obrazach; interfejs HighGUI
- Operacje binarne na obrazach: progowanie, operacje morfologiczne, erozja i dylacja, otwarcie i zamknięcie; analiza połączonych komponentów i konturów; wykrywanie obszarów
- Wzmacnianie i filtrowanie obrazów: przestrzenie kolorów i ich transformacje, filtrowanie, wygładzanie, metody gradientowe
- Zaawansowane przetwarzanie obrazów i fotografia obliczeniowa: transformata Hougha, obrazy HDR; gładkie klonowanie; usuwanie niechcianych obiektów
- Transformacje geometryczne i cechy obrazów: transformacje afiniczne, homografia, cechy obrazów, łączenie cech, wyrównywanie obrazów, tworzenie panoramy; szukanie znanych obiektów na obrazie
- Segmentacja i rozpoznawanie obrazów: segmentacja obrazów, rozpoznawanie obiektów, klasyfikacja obrazów
- Analiza wideo: przepływ optyczny, śledzenie obiektów, filtr Kalmana, algorytmy Meanshift i Camshift
- Rozpoznawanie twarzy metodami Eigen Faces i Fisher Faces, histogramy lokalnych wzorców binarnych, PCA i LDA
- Wykrywanie i rozpoznawanie tekstu
- Metody głębokiego uczenia dla klasyfikacji obrazów, rozpoznawania obiektów, rozpoznawania twarzy, szacowania pozy człowieka

Nazwa zajęć**: Warsztaty tłumaczenia automatycznego**
Po zakończeniu zajęć i potwierdzeniu osiągnięcia efektów uczenia się student:
- Rozumie pojęcie „jednostka nazwana” i potrafi wskazać jej wystąpienia w tekście.
- Potrafi przygotować korpus danych niezbędny do stworzenia modelu rozpoznawania jednostek nazwanych.
- Potrafi wytrenować model rozpoznawania jednostek nazwanych.
- Umie stworzyć system rozpoznający jednostki nazwane.
- Umie ocenić skuteczność systemu rozpoznawania jednostek nazwanych.
- Rozumie pojęcie korpusu równoległego i potrafi zastosować w praktyce metody jego pozyskania.
- Potrafi wydzielić zbiory danych konieczne do wytrenowania i ewaluacji modelu neuronowego tłumaczenia automatycznego.
- Rozumie pojęcie jednostek podwyrazowych i potrafi zastosować w praktyce algorytm Byte Pair Encoding w celu przetworzenia zbiorów danych.
- Potrafi wytrenować model neuronowego tłumaczenia automatycznego oraz zastosować go do tłumaczenia tekstu.

- Potrafi zastosować rozpoznawanie jednostek nazwanych w celu poprawienia jakości tłumaczenia automatycznego.
- Umie ocenić jakość modelu tłumaczenia automatycznego.

Treści programowe dla zajęć:

- Pojęcie i charakterystyka jednostek nazwanych. Różnica pomiędzy rozpoznawaniem jednostek nazwanych a ekstrakcją informacji. Wizualizacja jednostek nazwanych w tekście za pomocą biblioteki spaCy.
- Opracowanie cech dobrego korpusu danych dla zadania rozpoznawania jednostek nazwanych oraz metod jego pozyskiwania. Pozyskanie korpusu danych i przygotowanie go do przetworzenia za pomocą biblioteki flair.
- Implementacja modułu trenującego model rozpoznawania jednostek nazwanych. Wykorzystanie biblioteki flair.
- Implementacja systemu rozpoznającego jednostki nazwane w języku programowania Python 3.
- Charakterystyka metod ewaluacji systemu rozpoznawania jednostek nazwanych. Ewaluacja stworzonego systemu.
- Pojęcie korpusu równoległego. Pozyskanie korpusów równoległych ze zbiorów ogólnodostępnych. Zastosowanie narzędzi ekstrakcji tekstu z dokumentów. Zastosowanie narzędzia hunalign w celu stworzenia korpusu równoległego.
- Wstępne przetwarzanie korpusu. Wydzielenie zbiorów danych uczących i testowych z korpusu przy zachowaniu odpowiednich proporcji.
- Pojęcie jednostek podwyrazowych i algorytm Byte Pair Encoding. Porównanie narzędzi Subword-NMT, fastBPE i SentencePiece. Wytrenowanie modelu SentencePiece i zastosowanie go w celu przetworzenia zbiorów danych.
- Zaznajomienie z narzędziem fairseq. Binaryzacja danych oraz wytrenowanie modelu tłumaczenia automatycznego. Zastosowanie wytrenowanego modelu do tłumaczenia tekstu.
- Pojęcie ograniczeń leksykalnych. Zastosowanie rozpoznawanych jednostek nazwanych jako ograniczeń leksykalnych podczas tłumaczenia automatycznego w celu polepszenia jakości tłumaczenia.
- Ewaluacja ludzka oraz ewaluacja automatyczna z wykorzystaniem metryki BLEU. Ograniczenia metryki BLEU. Wykorzystanie narzędzia GEval do automatycznej ewaluacji stworzonego systemu tłumaczenia.

Nazwa zajęć**: Uczenie maszynowe**

Po zakończeniu zajęć i potwierdzeniu osiągnięcia efektów uczenia się student:

- Rozumie rolę i znaczenie uczenia maszynowego we współczesnej informatyce, potrafi wskazać przykłady zastosowań uczenia maszynowego.
- Potrafi wyróżnić podstawowe typy zadań uczenia maszynowego i wskazać ich przykłady.
- Umie korzystać z podstawowych narzędzi bibliotek NumPy i PyTorch oraz elementów języka Python przydatnych do implementowania rozwiązań z dziedziny uczenia maszynowego.
- Umie przetwarzać dane przechowywane w tekstowych formatach tabelarycznych (CSV/TSV).
- Umie wizualizować dane, korzystając z bibliotek Matplotlib i Seaborn.
- Rozumie zagadnienie regresji liniowej jednej i wielu zmiennych.
- Rozumie metodę gradientu prostego.
- Umie zaimplementować algorytm gradientu prostego do znalezienia rozwiązania problemu regresji liniowej.
- Rozumie zagadnienie regresji logistycznej.
- Umie zaimplementować algorytm gradientu prostego do znalezienia rozwiązania problemu regresji logistycznej.
- Rozumie znaczenie ewaluacji algorytmów uczenia maszynowego i zna jej podstawowe metody.
- Rozumie rolę zbiorów danych: uczącego, walidacyjnego i testowego, i potrafi z nich korzystać.
- Zna podstawowe miary jakości stosowane przy ewaluacji algorytmów uczenia maszynowego.
- Potrafi korzystać z modułów pakietu Scikit-Learn do implementacji rozwiązań uczenia maszynowego.
- Potrafi dokonać ewaluacji zaimplementowanego rozwiązania.
- Rozumie zjawiska nadmiernego i niedostatecznego dopasowania.
- Zna metody regularyzacji.
- Umie zapobiegać nadmiernemu i niedostatecznemu dopasowaniu w implementowanych przez siebie rozwiązaniach.

- Umie poprawnie reprezentować dane różnych typów i korzystać z nich do rozwiązywania problemów metodami uczenia maszynowego.
- Rozumie znaczenie optymalizacji i zna jej podstawowe metody.
- Umie stosować metody optymalizacji uczenia maszynowego.
- Rozumie ideę uczenia nienadzorowanego i zna najważniejsze algorytmy uczenia nienadzorowanego.
- Potrafi zaimplementować przykładowy algorytm uczenia nienadzorowanego.
- Rozumie zasadę działania naiwnego klasyfikatora bayesowskiego.
- Rozumie zasadę działania algorytmu k najbliższych sąsiadów.
- Rozumie zasadę działania drzew decyzyjnych.
- Rozumie zasadę działania sztucznych sieci neuronowych, w tym wielowarstwowych.
- Potrafi wykorzystywać metodę propagacji wstecznej do uczenia wielowarstwowych sieci neuronowych.
- Potrafi implementować sieci neuronowe z wykorzystaniem biblioteki PyTorch.
- Potrafi implementować sieci neuronowe z wykorzystaniem biblioteki Keras.
- Rozumie zasadę działania i potrafi wskazać zastosowania rekurencyjnych sieci neuronowych.
- Rozumie zasadę działania i potrafi wskazać zastosowania splotowych sieci neuronowych.
- Rozumie zasadę działania i potrafi wskazać zastosowania modeli typu encoder-decoder, w szczególności do tworzenia modeli języka i tłumaczenia maszynowego.
- Rozumie mechanizm atencji i zasadę działania architektury typu transformer, potrafi wskazać ich zastosowania.
- Rozumie ideę uczenia przez wzmacnianie i zna podstawowe paradygmaty uczenia przez wzmacnianie.
- Potrafi zaprojektować, zaimplementować i zewaluować system wykorzystujący uczenie maszynowe.

Treści programowe dla zajęć:

- Wprowadzenie do uczenia maszynowego. Czym jest uczenie maszynowe? Rola i miejsce uczenia maszynowego we współczesnej informatyce. Przegląd zastosowań i metod uczenia maszynowego. Podstawowe pojęcia związane z uczeniem maszynowym.
- Podstawowe narzędzia uczenia maszynowego. Elementy języka Python przydatne przy implementowaniu algorytmów uczenia maszynowego. Biblioteki NumPy i PyTorch.
- Narzędzia przetwarzania i wizualizacji danych w języku Python. Format CSV/TSV. Biblioteki Matplotlib i Seaborn.
- Regresja liniowa jednej zmiennej. Funkcja kosztu. Metoda gradientu prostego. Regresja liniowa wielu zmiennych.
- Implementacja regresji liniowej jednej zmiennej w języku Python.
- Regresja logistyczna. Metoda gradientu prostego dla regresji logistycznej.
- Implementacja regresji logistycznej w języku Python.
- Ewaluacja algorytmów uczenia maszynowego. Podział na zbiory: uczący, testowy i walidacyjny. Walidacja krzyżowa. Miary jakości.
- Pakiet Scikit-Learn. Implementacja regresji liniowej i regresji logistycznej z wykorzystaniem gotowych modułów. Implementacja wybranych metod ewaluacji.
- Nadmierne i niedostateczne dopasowanie. Obciążenie i wariancja. Ilustracja problemu nadmiernego dopasowania na przykładzie regresji wielomianowej. Metody regularyzacji.
- Nadmierne i niedostateczne dopasowanie w praktyce. Implementacja metod zapobiegających nadmiernemu dopasowaniu.
- Sposoby reprezentacji danych. Implementacja algorytmów regresji dla danych różnych typów, w tym dla danych nieliczbowych, oraz dla danych niepełnych.
- Stochastic Gradient Descent. Przegląd metod optymalizacji.
- Porównanie różnych metod optymalizacji na przykładach.
- Uczenie nienadzorowane. Algorytm k średnich. Algorytm analizy głównych składowych.
- Implementacja metod uczenia nienadzorowanego na przykładzie algorytmu k średnich.
- Przegląd metod uczenia nadzorowanego. Naiwny klasyfikator bayesowski. Algorytm k najbliższych sąsiadów. Drzewa decyzyjne.
- Wprowadzenie do sztucznych sieci neuronowych. Prosty perceptron. Funkcje aktywacji. Głębokie uczenie maszynowe. Wielowarstwowe sieci neuronowe.
- Propagacja wsteczna. Uczenie wielowarstwowych sieci neuronowych.
- Implementacja sieci neuronowych z wykorzystaniem biblioteki PyTorch.

- Implementacja sieci neuronowych z wykorzystaniem biblioteki Keras.
- Rekurencyjne sieci neuronowe – idea, przegląd najpopularniejszych architektur, przegląd zastosowań.
- Splotowe sieci neuronowe – idea, przegląd najpopularniejszych architektur, przegląd zastosowań.
- Modele typu encoder-decoder. Neuronowe tłumaczenie maszynowe. Autoencoder. Word embeddings.
- Mechanizm atencji. Modele typu Transformer.
- Uczenie przez wzmacnianie.
- Indywidualny projekt programistyczny – implementacja wybranych metod uczenia maszynowego.

Nazwa zajęć**: Uczenie głębokie w przetwarzaniu tekstu**
Po zakończeniu zajęć i potwierdzeniu osiągnięcia efektów uczenia się student:

- Zna neuronowe modele języka: „word2vec", „GloVe" oraz „fastText" z uwzględnieniem ich różnic. Rozumie pojęcia „word embedding" oraz „sentence embedding". Potrafi wytrenować neuronowy model języka: „word2vec", „Glove" lub „fastText".
- Potrafi rozwiązać problem „analogii słów".
- Zna oraz potrafi wskazać sposoby rozwiązania problemu „rzadkich słów" – BPE (Byte Pair Encoding), SentencePiece.
- Zna neuronowe modele języka oparte na architekturze Transformer. Potrafi wskazać różnicę pomiędzy modelami opartymi na architekturze enkoder oraz dekoder.
- Zna oraz potrafi wskazać cech charakterystyczne neuronowych modeli języka: BERT, RoBERTa, GPT-2, GPT-3, Polish RoBERTa.
- Zna oraz potrafi wskazać cechy charakterystyczne neuronowych modeli języka: XLM (RoBERTa), Multilingual BERT, T5, mT5.
- Zna problem długości sekwencji w tekście oraz potrafi wskazać sposoby na rozwiązanie tego problemu. Zna neuronowe modele języka: Longformer oraz Linformer.
- Zna różnicę pomiędzy modelami dziedzinowymi i ogólnymi (jednojęzyczne oraz wielojęzyczne). Potrafi wskazać wady i zalety modeli.
- Potrafi wytrenować oraz dostroić neuronowy model języka. Zna pojęcia „pretraining" oraz „fine-tuning".
- Potrafi wykorzystać neuronowy model języka w problemie klasyfikacji oraz podobieństwie semantycznym zdań.

Treści programowe dla zajęć:

- Pojęcie i charakterystyka neuronowych modeli języka: „word2vec", „GloVe" oraz „fastText". Znaczenie pojęć „word embedding" oraz „sentence embedding". Wytrenowanie neuronowego modelu języka „word2vec", „Glove" lub „fastText" z wykorzystanie gotowych narzędzi.
- Rozwiązanie problemu „analogii słów" z wykorzystaniem biblioteki gensim.
- Rozwiązanie problemu „rzadkich słów" z wykorzystaniem BPE (Byte Pair Encoding) oraz SentencePiece.
- Charakterystyka architektury Transformer, modeli opartych na architekturze enkoder oraz dekoder.
- Charakterystyka neuronowych modeli języka: BERT, RoBERTa, GPT-2, GPT-3, Polish RoBERTa.
- Charakterystyka neuronowych modeli języka: XLM (RoBERTa), Multilingual BERT, T5, mT5.
- Rozwiązywanie problemu długości sekwencji, wykorzystanie neuronowych modeli języka: Longformer oraz Linformer.
- Charakterystyka dziedzinowych, ogólnych (jednojęzycznych oraz wielojęzycznych) modeli.
- Trenowanie oraz dostrojenie neuronowych modeli języka z wykorzystaniem biblioteki transformers.
- Trenowanie neuronowych modeli języka w zadaniach klasyfikacji oraz podobieństwa semantycznego zdań.

Nazwa zajęć**: Testy Penetracyjne**
Po zakończeniu zajęć i potwierdzeniu osiągnięcia efektów uczenia się student:

- Rozumie etyczne oraz prawne obostrzenia dotyczące testów penetracyjnych.
- Potrafi zainicjować test penetracyjny oraz przygotować niezbędne środowisko pracy.
- Potrafi przygotować prawidłowy raport z testu penetracyjnego.

- Potrafi przeprowadzić rekonesans pasywny na temat sieci/domeny/firmy będącej celem testu penetracyjnego.
- Potrafi przeprowadzić rekonesans aktywny na temat sieci/systemu będącego celem testu penetracyjnego.
- Potrafi zaprogramować proste narzędzia automatyzujące pracę podczas testu penetracyjnego.
- Potrafi przeprowadzić enumerację systemu Linux.
- Potrafi przeprowadzić enumerację systemu Windows.
- Zna zagadnienia związane z podnoszeniem uprawnień.
- Potrafi korzystać ze skanera podatności oraz interpretować wyniki skanów.
- Potrafi wyszukiwać podatności na podstawie zdobytych informacji.
- Zna aspekty pracy z exploitami oraz potrafi z nich skorzystać.
- Potrafi przeprowadzić zdalny atak na systemy uwierzytelniające różnych usług.
- Potrafi przeprowadzić atak offline na różnego rodzaju skróty haseł.
- Zna metodykę prowadzenia testu penetracyjnego aplikacji webowej.
- Potrafi mapować aplikację webową oraz wykrywać ukryte zasoby.
- Potrafi namierzyć i wykorzystać podstawowe błędy obsługi danych od użytkownika.
- Potrafi namierzyć i wykorzystać podatności związane z obsługą plików.
- Potrafi namierzyć i wykorzystać inne podatności w aplikacjach webowych.
- Zna techniki postexploitacyjne w systemach Linux i Windows.
- Potrafi wykorzystać oprogramowanie metasploit framework podczas testu penetracyjnego.
- Zna zasadę działania ataków typu client-side.
- Zna metodykę ataków socjotechnicznych oraz potrafi przeprowadzić symulację ataku phishingowego.
- Zna podstawowe techniki omijania mechanizmów bezpieczeństwa.
- Potrafi przełamać zabezpieczenia sieci-wifi opartej o WPA2.

Treści programowe dla zajęć:

- Wprowadzenie do testów penetracyjnych: Metodyki prowadzenia testów penetracyjnych. Zapoznanie z etycznymi oraz prawnymi aspektami testów penetracyjnych. Wskazanie kierunków rozwoju, źródeł i sposobów nabywania umiejętności oraz omówienie przydatnej literatury. Przygotowanie środowiska laboratoryjnego.
- Raportowanie testu penetracyjnego: Przygotowanie szablonu raportu z testu penetracyjnego. Omówienie istotnych elementów raportu oraz kryteriów świadczących o jego jakości. Otwarcie raportu z opisem przeprowadzonych działań w trakcie trwania kursu, który będzie głównym elementem zaliczenia kursu.
- Rekonesans pasywny: Omówienie metod przeprowadzania białego wywiadu, informacji jakie mogą być przydatne do dalszych działań oraz przedstawienie publicznie dostępnych źródeł.
- Rekonesans aktywny: Wprowadzenie do aktywnej enumeracji sieci wraz z przedstawieniem niezbędnika narzędziowego każdego pentestera.
- Programowanie: Automatyzacja prac wykonywanych przez pentestera z wykorzystaniem języków programowania i poleceń powłoki (Python, Bash, Powershell).
- Enumeracja w systemach Linux: Przeprowadzenie rekonesansu w systemie Linux. Wykorzystanie błędów konfiguracji systemu i działającym na nich usług do podniesienia uprawnień oraz pozyskania informacji użytecznych podczas testu penetracyjnego.
- Enumeracja w systemach Windows: Przeprowadzenie rekonesansu w systemie Windows. Wykorzystanie błędów konfiguracji systemu i działającym na nich usług do podniesienia uprawnień oraz pozyskania informacji użytecznych podczas testu penetracyjnego.
- Ocena podatności: Analiza zdobytych informacji, metody wyszukiwania podatności. Wykorzystanie skanerów podatności. Praca z exploitami.
- Ataki na hasła: Przeprowadzenie ataków na zdalne mechanizmy uwierzytelniające. Przeprowadzenie ataków offline na różnego rodzaju skróty haseł.
- Testy penetracyjne aplikacji webowych: Metodyka prowadzenia testów penetracyjnych aplikacji webowych. Modelowanie zagrożeń. Przydatne narzędzia. Zdobywanie informacji o celu. Mapowanie aplikacji. Wykrywanie ukrytych zasobów.
- Podstawowe błędy obsługi danych od użytkownika: Wykrywanie i wykorzystywanie podatności typu SQL Injection, Cross-Site Scripting, Code Injection.
- Podatności związane z obsługą plików: Wykrywanie podatności typu Local/Remote File Inclusion. Wykrywanie podatności związanych z uploadem plików oraz obsługą różnych rozszerzeń (XML, SVG).

- Pozostałe podatności w aplikacjach webowych: Ataki na mechanizmy zarządzania sesją, wykorzystanie podatności Cross-Site Request Forgery, analiza podatności w logice biznesowej aplikacji, wykorzystanie podatności typu Server Side Request Forgery. Wykorzystanie podatności związanych z deserializacją danych.
- Techniki postexploitacyjne w systemach Linux i Windows: Alternatywne sposoby transferu plików. Podnoszenie uprawnień w systemie. Lateral movements. Przekierowania portów i tunelowanie ruchu.
- Metasploit framework: Praca z narzędziem msfconsole. Wykorzystanie narzędzia meterpreter. Omówienie dodatkowych narzędzi z pakietu msf.
- Ataki typu client-side: Opracowanie ataku typu client-side. Wprowadzenie do ataków socjotechnicznych i przeprowadzenie ataku phishingowego.
- Techniki omijania: Omówienie zasad funkcjonowania mechanizmów bezpieczeństwa typu anty wirus, web application firewall czy filtr antyspamowy. Przeprowadzenie działań mających na celu ich ominięcie.
- Sieci Wi-Fi: Omówienie sposobów ataków na sieci wi-fi oraz ich klientów. Przeprowadzenie ataku na standard WPA2.
- Podsumowanie kursu: Prezentacja stworzonych raportów oraz ich omówienie.

Nazwa zajęć**: Sztuczna empatia**

Po zakończeniu zajęć i potwierdzeniu osiągnięcia efektów uczenia się student:

- Zna podstawowe pojęcia związane z empatią. Potrafi określić poziom swojej empatii. Zauważa celowość stosowania empatii zarówno w życiu codziennym jak i w systemach informatycznych.
- Zna podstawy etyki tworzenia sztucznej inteligencji. Rozumie aspekty społeczne rozwijania sztucznej inteligencji. Potrafi wytyczyć cele badań i rozwoju psychologicznego sztucznej inteligencji.
- Potrafi odróżnić naśladownictwo od nauki w systemach informatycznych. Zna podstawowe metody implementacji empatii w robotyce.
- Potrafi wskazać cele i zalety implementacji empatii w systemach dialogowych. Zna metody implementacji zachowań empatycznych w systemach dialogowych.
- Potrafi określić wymagania użytkownika dotyczące zarządzania emocjami w grach komputerowych.
- Zna aktualne i istniejące zastosowania systemów wykorzystujących sztuczną empatię. Potrafi określić cele i zalety stosowania sztucznej empatii w marketingu.
- Zna zastosowania sztucznej empatii w medycynie.
- Zna metody rozpoznawania emocji. Potrafi podzielić je w zależności od sygnału wejściowego. Potrafi zaplanować zastosowanie odpowiedniego mechanizmu rozpoznawania emocji.
- Zna mechanizmy zachowań stadnych i potrafi określić ich zastosowanie w robotach społecznych, np. w roju.
- Potrafi zaplanować rozszerzenie prostego istniejącego systemu dialogowego o mechanizmy umożliwiające modelowanie emocji.
- Potrafi stworzyć słownik emocji, zaplanować zachowania agenta oraz dokonać mapowania emocji na zachowania agenta.
- Potrafi zaimplementować zachowania empatyczne.
- Potrafi zaimplementować graficzną reprezentację emocji.
- Potrafi testować system informatyczny.
- Potrafi zaprezentować i scharakteryzować wykonany system informatyczny.

Treści programowe dla zajęć:

- Podstawy empatii: Podstawowe pojęcia związane z empatią. Empatia kognitywna. Neuronauka. Skrócony test ilorazu empatii.
- Podstawy empatii: Podstawowe pojęcia związane z empatią. Neurologiczne podstawy zjawiska empatii. Rodzaje empatii.
- Empatia a prawdziwa Sztuczna Inteligencja (SI): Empatia a doskonała SI. Test Turinga. Hierarchia emocji. Zasady z Asimolar. Perspektywy na przyszłość.
- Empatia a prawdziwa SI: Empatia a doskonała SI. Nauka poprzez empatię. Odtwarzanie modelu rozwoju dziecka.
- Empatia w robotyce: Naśladownictwo i mimikra. Systemy wizyjne i uczenie maszynowe.
- Empatia w robotyce: Systemy wizyjne i uczenie maszynowe. Sieci neuronowe. Proste roboty naśladujące i uczące się: Affetto i WAMEOBA

- Empatia w systemach dialogowych: Chatboty i wirtualni asystenci. Analiza języka mówionego i pisanego.
- Empatia w systemach dialogowych: Analiza języka mówionego i pisanego. Metody implementacji emocji w systemach dialogowych. Emocjonalni wirtualni rozmówcy.
- Empatia w grach: Zagadnienie immersji. Realistyczni NPCs. Avatary. Analiza wymagań użytkownika.
- Empatia w grach: Zagadnienie immersji. Analiza wymagań gracza.
- Empatia w zastosowaniach praktycznych: Zastosowania sztucznej empatii w motoryzacji, marketingu i naukach społecznych.
- Empatia w zastosowaniach praktycznych: Rozpoznawanie emocji w marketingu i jego wykorzystanie w celu zwiększenia sprzedaży. Spersonalizowane reklamy. Nastrój.
- Empatia w medycynie: Uszkodzenia mózgu a empatia. Zastosowania w niwelowaniu skutków Alzheimera. Gra FearNot!
- Empatia w medycynie: Uszkodzenia mózgu i ich wpływ na empatię. Empatia emocjonalna i empatia kognitywna.
- Rozpoznawanie emocji: Metody i sposoby rozpoznawania emocji. Affectiva.
- Rozpoznawanie emocji: Metody i sposoby rozpoznawania emocji.
- Behawioryzm: Zachowania stadne i społeczne. Dynamika roju. Test wiedzy z zakresu zajęć 1-9.
- Behawioryzm: Zachowania stadne i społeczne implementowane w robotyce. Rój.
- Tworzenie empatycznego bota I: Zapoznanie się z bazowym systemem, na którym rozwijany będzie projekt empatycznego bota. Praca w zespołach.
- Tworzenie empatycznego bota I: Przygotowanie bazowego system oraz zaplanowanie postaci oraz metod implementacji modułów umożliwiających rozpoznanie emocji I zachowania empatyczne. Praca w zespołach.
- Tworzenie empatycznego bota II: Tworzenie słownika emocji, zbioru zachowań, mapowanie emocji na odpowiednie zachowania. Praca w zespołach.
- Tworzenie empatycznego bota II: Planowanie szczegółowych rozwiązań oraz komponentów koniecznych do implementacji modułu rozpoznania emocji I zachowań empatycznych. Praca w zespołach.
- Tworzenie empatycznego bota III: Implementacja zachowań. Praca w zespołach.
- Tworzenie empatycznego bota III: Implementacja zaprojektowanych rozwiązań. Praca w zespołach.
- Tworzenie empatycznego bota IV: Implementacja graficznej reprezentacji emocji. Praca w zespołach.
- Tworzenie empatycznego bota IV: Implementacja wybranej reprezentacji graficznej emocji. Praca w zespołach.
- Tworzenie empatycznego bota V: Testowanie. Praca w zespołach.
- Tworzenie empatycznego bota V: Testowanie wdrożonych rozwiązań I całego systemu. Praca w zespołach.
- Podsumowanie zajęć: Przedstawienie botów. Prezentacje multimedialne.
- Podsumowanie zajęć: Stworzenie prezentacji multimedialnej nt. powstałego bota. Przygotowanie bota do prezentacji.

Nazwa zajęć**: Systemy rozmyte**

Po zakończeniu zajęć i potwierdzeniu osiągnięcia efektów uczenia się student:

- zna pojęcie zbioru rozmytego, umie z jego pomocą modelować pojęcia nieprecyzyjne
- umie operować na zbiorach rozmytych, zna praktyczny aspekt tych operacji
- rozumie istotę i przeznaczenie sterowania rozmytego w porównaniu z konwencjonalnymi metodami sterowania, zna zasady działania sterownika rozmytego, potrafi zaprojektować i zaimplementować prosty sterownik rozmyty
- zna dostępne biblioteki oprogramowania w standardzie FCL i potrafi je zastosować do implementacji własnych sterowników

Treści programowe dla zajęć:

- Pojęcie zbioru rozmytego, jego modelowanie i przykłady
- Pojęcie zmiennej lingwistycznej i jej modelowanie
- Sterowanie rozmyte wraz z przykładami
- Język FCL i jego implementacje

- Algorytmy rozmyte w systemach podejmowania decyzji
- Projekt

Nazwa zajęć**: Systemy dialogowe**
Po zakończeniu zajęć i potwierdzeniu osiągnięcia efektów uczenia się student:
- Zna podstawowe rodzaje systemów dialogowych i potrafi wskazać występujące pomiędzy nimi różnice.
- Potrafi zaimplementować prostego chatbota.
- Potrafi przeprowadzić analizę wymagań na potrzeby budowy systemu dialogowego.
- Potrafi pozyskiwać dane do budowy systemu dialogowego.
- Potrafi scharakteryzować strukturę semantyczną dialogu.
- Potrafi opisać architekturę systemu dialogowego ukierunkowanego na wykonanie zadania.
- Potrafi napisać gramatykę dla parsera semantycznego.
- Potrafi zastosować techniki uczenia maszynowego do prowadzenia analizy semantycznej wypowiedzi.
- Potrafi konstruować reguły zarządzania dialogiem.
- Potrafi zastosować techniki uczenia maszynowego do zarządzania dialogiem.
- Potrafi posługiwać się technikami generowania tekstu.
- Zna techniki ujednoznaczniania wypowiedzi w systemach ukierunkowanych na wykonywanie wielu zadań równocześnie.
- Potrafi ocenić jakość systemu dialogowego z wykorzystaniem powszechnie przyjętych metod i metryk.
- Potrafi opisać architekturę systemu dialogowego end-to-end.
- Potrafi zaimplementować system dialogowy w architekturze end-to-end.

Treści programowe dla zajęć:
- Rodzaje systemów dialogowych.
- Systemy dialogowe ukierunkowane na wykonywanie zadań oraz chatboty.
- Analiza wymagań, jakie powinien spełniać system dialogowy.
- Metody pozyskiwania danych do budowy systemów dialogowych.
- Eksperymenty typu „Czarnoksiężnik z Oz”.
- Charakterystyka struktury semantycznej dialogu: akty mowy, ramy i sloty. Poprawki, wtręty i żądania resetu. Annotowane korpusy dialogów.
- Architektura systemu dialogowego ukierunkowanego na wykonanie zadania.
- Parsing semantyczny z wykorzystaniem gramatyk.
- Parsing semantyczny z wykorzystaniem technik uczenia maszynowego.
- Zarządzanie dialogiem z wykorzystaniem reguł.
- Zarządzanie dialogiem z wykorzystaniem technik uczenia maszynowego.
- Generowanie odpowiedzi.
- Ujednoznacznianie wypowiedzi w systemach ukierunkowanych na wykonywanie wielu zadań równocześnie.
- Inteligentne asystenty osobiste.
- Ewaluacja systemów dialogowych – metody i metryki.
- Systemy dialogowe end-to-end. Zbiory danych, agenty, środowiska wykonania i symulatory.

Nazwa zajęć**: Symulowanie wizualne**
Po zakończeniu zajęć i potwierdzeniu osiągnięcia efektów uczenia się student:
- Rozumie zasady przedstawienia rozwoju biologicznego za pomocy reguł zamiennych. Zna definicje L-Systemow. Potrafi stworzyć reprezentację dwuwymiarową prostych struktur danych.
- Potrafi z prostych struktur danych stworzyć reprezentację trójwymiarową.
- Potrafi wytłumaczyć komórkowe automaty Ulama. Zna reguły Game of Life.
- Potrafi implementować modele segregacji Schellinga. Rozumie pojęcie tipping points w modelach komórkowych. Potrafi wytłumaczy model Gravettera, model standing ovation i ruchy Boida.
- Potrafi zaimplementować proces wzrostu populacji bakterii za pomocą równań różniczkowych.
- Potrafi zaimplementować proces dyfuzji za pomocy równań różniczkowych.
- Potrafi zastosować metodę nullclines i przeprowadzić analizę za pomocy macierzy Jacobiego.
- Potrafi zaimplementować model Turinga do generowania pasm i kropek 2D.

- Rozumie sposób kinetycznego modelu Monte Carlo i potrafi nim zaimplementować pasma Turinga.
- Rozumie architekturę i działanie sieci konwolucyjnych. Potrafi stworzyć CNN za pomocy biblioteki Keras.
- Rozumie pojecia saliency maps i transfer learning. Potrafi zaimplementować sieć neuronową wyuczoną na danych syntetycznych.
- Umie stosować narzędzia w dziedzinie data segmentation. Potrafi zaimplementować i wyuczyć sieć Mask R-CNN za pomocy biblioteki Keras.
- Rozumie sposób działania i konstrukcję sieci typu autoencoder. Potrafi uruchomić i zastosować sieć pointnet.
- Rozumie sposób działania I konstrukcję sieci typu GAN. Potrafi zastosować Cycle-GAN do augmentacji obrazów.

Treści programowe dla zajęć:

- Wprowadzenie do L-Systemów. Modele kontekstowe, parametryczne i stochastyczne.
- Wprowadzenie do modelowania trójwymiarowego za pomocy L-Systemów. Przetwarzanie przykładu generowania różnych filotaksji.
- Modele komórkowe i ich zastosowania.
- Modelowanie społeczne: modele Schellinga, Gravettera, model standing ovation.
- Wprowadzenie do modelowania za pomocy równań różniczkowych. Przetwarzanie przykładu modelowania wzrostu populacji bakterii.
- Wyprowadzenie modelu dyfuzji. Omówienie pierwszego i drugiego prawa Ficka.
- Przeprowadzenie analizy stabilności.
- Wprowadzenie do modeli dyfuzja-reakcja.
- Modele Monte Carlo i ich porównanie do modeli Turinga
- Wprowadzenie do uczenia maszynowego za pomocą splotowych sieci neuronowych: motywacja i architektura.
- Splotowe sieci neuronowe, cd.: Saliency maps i transfer learning.
- Segmentacja obrazów i wykrywanie obiektów.
- Mask R-CNN.
- Sieci typu autoencoder na przykładzie sieci pointnet.
- Wprowadzenie do sieci typu Generative Adversarial Networks. Przetwarzanie przykładu Cycle-GAN.

Nazwa zajęć**: Warsztat kompetencji miękkich dla cyberspecjalisty**

Po zakończeniu zajęć i potwierdzeniu osiągnięcia efektów uczenia się student:

- Potrafi analizować własny styl funkcjonowania, a w efekcie zwiększa efektywność osobistą.
- Umie rozpoznawać swój potencjał: mocne i słabe strony.
- Zna i stosuje w praktyce narzędzia poprawienia efektywności zespołu, z którym pracuje.
- Rozpoznaje style myślenia oraz funkcjonowania.
- Skutecznie poprawia swoje relacje społeczne.
- Bardziej efektywnie współpracuje w zespole. Docenia znaczenie pracy zespołowej.
- Skuteczniej komunikuje się z otoczeniem – współpracownikami, przełożonymi, bliskimi w życiu prywatnym
- Rozumie i potrafi wytłumaczyć wybrane teorie i narzędzia z obszaru psychologii i socjologii.
- Zna metodologię i znaczenie inwentarzy osobowości oraz testów psychologicznych.
- Rozumie znaczenie kompetencji miękkich w życiu zawodowym i osobistym.
- Potrafi przeprowadzić autoprezentację i profesjonalne wystąpienie publiczne.
- Udziela profesjonalnej informacji zwrotnej. Rozumie znaczenie informacji zwrotnej w motywowaniu ludzi.
- Stosuje podstawowe narzędzia z obszaru coachingu i mentoringu.
- Potrafi wyciągać wnioski z porażek w celu podejmowania trafnych decyzji w przyszłości.

Treści programowe dla zajęć:

- Kompetencje społeczne. Kompetencje miękkie i twarde. Kompetencje przyszłości (2030). Leadership.
- Warsztat team-buildingu.
- Warsztat kompetencji interpersonalnych.
- Warsztat efektywności osobistej.
- Sztuka feedbacku. Model FUKO.

- Coaching. Implementacja najważniejszych narzędzi.
- Mentoring w pracy zawodowej.
- Autoprezentacja i wystąpienia publiczne.
- Sztuka błądzenia – jak wyciągać wnioski z porażek i niepowodzeń w życiu zawodowym (prywatnym).
- Model pogłębionych relacji międzyludzkich.

Nazwa zajęć**: Przygotowanie do projektu badawczo-rozwojowego**
Po zakończeniu zajęć i potwierdzeniu osiągnięcia efektów uczenia się student:

- Rozumie społeczne aspekty pracy zespołowej w projekcie badawczo-rozwojowym.
- Potrafi określić cechy innowacyjnego projektu informatycznego.
- Potrafi wizualizować system informatyczny za pomocą makiety dynamicznej.
- Potrafi pozyskiwać inwestorów dla projektu badawczo-rozwojowego.
- Umie przygotować prezentację koncepcji projektu badawczo-rozwojowego.
- Potrafi publicznie prezentować koncepcję projektu badawczo-rozwojowego.
- Potrafi opracować business-plan projektu badawczo-rozwojowego.
- Potrafi uczestniczyć w projekcie prowadzonym we współpracy z inwestorem i użytkownikami.
- Potrafi dostarczać częściowe rezultaty prac wykonywanych w ramach projektu-badawczego.
- Potrafi opracować specyfikację zakresu systemu informatycznego.
- Potrafi przeprowadzić proces testowania w metodyce zwinnej.
- Umie zorganizować proces testowania integracyjnego i systemowego.
- Potrafi zapewnić użyteczność systemu informatycznego, będącego wynikiem prac badawczo-rozwojowych.
- Potrafi uruchomić procesy weryfikujące jakość systemu informatycznego.
- Potrafi planować zadania w projekcie badawczo-rozwojowym.
- Zna specyfikę zarządzania projektem badawczo-rozwojowym.
- Potrafi przedstawić cele i działanie systemu informatycznego jego interesariuszom.
- Potrafi przygotować i przeprowadzić demonstrację systemu informatycznego.

Treści programowe dla zajęć:

- Społeczne aspekty pracy zespołowej w projekcie badawczo-rozwojowym: Charakterystyka pamięci ludzkiej. Motywacja do pracy. Aspekty pracy zespołowej. Cechy lidera.
- Społeczne aspekty pracy zespołowej w projekcie informatycznym: Symulacja pracy zespołowej w projekcie informatycznym pod stresem czasowym.
- Charakterystyka innowacyjnego systemu informatycznego: Cechy innowacyjnego systemu informatycznego. Wartość dodana produktu. Zasady osiągnięcia sukcesu na rynku nowych technologii.
- Charakterystyka innowacyjnego systemu informatycznego: Opracowanie koncepcji projektu spełniającego cechy innowacyjności. Wizualizacja systemu informatycznego za pomocą makiety dynamicznej. Analiza wartości dodanej projektowanego systemu.
- Pozyskiwanie inwestorów: Typy i przykłady inwestorów. Sposoby prezentacji projektu badawczo-rozwojowego: „elevator pitch”, prezentacja biznesowa. Tworzenie business planu.
- Pozyskiwanie inwestorów: Przygotowanie publicznej prezentacji koncepcji projektu badawczo-rozwojowego.
- Pozyskiwanie inwestorów: Publiczna prezentacja koncepcji projektu badawczo-rozwojowego biznesowego w obecności potencjalnych inwestorów i użytkowników.
- Pozyskiwanie inwestorów: Opracowanie business-planu projektu-badawczo-rozwojowego.
- Wybrane elementy inżynierii oprogramowania: Podstawowe koncepcje metodyk zwinnych. Metodyka SCRUM. Elementy metodyki Kanban. Ciągła integracja. Metody prototypowania. Współczesne systemy kontroli wersji.
- Wybrane elementy inżynierii oprogramowania: Zakładanie projektu w systemie Jira. Stworzenie dziennika projektu. Określenie metody prototypowania. Otwarcie projektu w systemie kontroli wersji. Założenie projektu w serwerze ciągłej integracji. Integracja systemu kontroli wersji z systemem ciągłej integracji.
- Formalna specyfikacja zakresu systemu informatycznego: Charakterystyka użytkowników. Lista aktor – cel. Lista in – out. Wymagania funkcjonalne i niefunkcjonalne. Formalny opis przypadków użycia.

- Formalna specyfikacja zakresu systemu informatycznego: Opracowanie charakterystyki użytkowników, listy aktor – cel, listy in – out, wymagań funkcjonalnych i niefunkcjonalnych oraz przypadków użycia dla realizowanych projektów.
- Testowanie w programowaniu zwinnym: Zarządzanie testowaniem w metodyce SCRUM. Poziomy testowania w metodyce SCRUM. Testowanie jednostkowe. Testowanie metodą „Test First". Typy wstawek testowych: stub, spy, mock, fake, dummy.
- Testowanie w programowaniu zwinnym – zastosowanie metod testowania zwinnego w projekcie badawczo-rozwojowym: Opracowanie testów jednostkowych. Zaimplementowanie metody TestFirst. Opracowanie wstawek testowych w projekcie badawczo-rozwojowym.
- Testowanie integracyjne i systemowe: Opracowywanie przypadków testowych dla testowania integracyjnego. Strategie testowania integracyjnego. Środowisko testowania systemowego. Testowanie manualne: testowania eksploracyjne, testowanie sesyjne, testowanie akceptacyjne. Testowanie automatyczne: oparte na nagrywaniu, oparte na słowach kluczowych, oparte na zachowaniu.
- Testowanie integracyjne i systemowe: Opracowanie przypadków testowych projektowanego systemu. Przygotowanie planu testowania. Przeprowadzenie testów manualnych. Opracowanie i przeprowadzenie testów automatycznych odpowiednich dla projektowanego systemu informatycznego.
- Wybrane zagadnienia użyteczności systemu informatycznego: Wprowadzenie pojęcia użyteczności systemu informatycznego. Omówienie aspektów użyteczności: kontekst, wprowadzanie danych, wyprowadzanie danych, responsywność, łączność z siecią, zasoby.
- Wybrane zagadnienia użyteczności systemu informatycznego: Modyfikacja dziennika projektu pod kątem użyteczności. Opracowanie szablonu raportu użyteczności dla projektowanego systemu.
- Przygotowanie testów użyteczności. Przeprowadzenie testów użyteczności.
- Ocena jakości systemu informatycznego: Określenie pojęcia jakości systemu informatycznego. Jakość systemu a jakość kodu. Metryki jakości oprogramowania. Metryki zadowolenia klienta.
- Ocena jakości systemu informatycznego: Opracowanie metryk jakości dla projektowanego systemu. Ewaluacja jakości projektowanego systemu według zadanych metryk jakości.
- Planowanie projektu: Zasady tworzenia harmonogramu projektu. Siatka podziału zadań. Zależności między zadaniami. Alokacja zasobów. Wykres Gantta. Narzędzie MS-Project.
- Planowanie projektu: Opracowanie harmonogramu wstecznego dla projektowanego systemu za pomocą programu MS-Project. Obliczenie wartości finansowej projektu.
- Zarządzanie projektem informatycznym: Specyfika zarządzania zespołem informatycznym. Sterowanie presją. Czynniki decydujące o wydajności pracy.
- Przygotowanie prezentacji systemu informatycznego dla interesariuszy i inwestorów.
- Przygotowanie publicznej demonstracji systemu
- Publiczna demonstracja systemu będącego rezultatem projektu badawczo-rozwojowego opracowanego podczas semestru dla inwestorów i użytkowników.

Nazwa zajęć**: Projekt badawczo-rozwojowy 1**
Po zakończeniu zajęć i potwierdzeniu osiągnięcia efektów uczenia się student:

- Potrafi zastosować znajomość aspektów społecznych pracy zespołowej w projekcie badawczo-rozwojowym.
- Potrafi wdrożyć cechy innowacyjnego projektu informatycznego w projekcie badawczo-rozwojowym.
- Potrafi wizualizować system informatyczny za pomocą makiety dynamicznej.
- Potrafi pozyskiwać inwestorów dla projektu badawczo-rozwojowego.
- Potrafi uczestniczyć w projekcie prowadzonym we współpracy z inwestorem i użytkownikami.
- Potrafi przygotować środowisko pracy w projekcie badawczo-rozwojowym.
- Potrafi dostarczać częściowe rezultaty prac wykonywanych w ramach projektu-badawczego.
- Potrafi zaimplementować użyteczny system informatyczny, będący wynikiem prac badawczo-rozwojowych.
- Potrafi zapewnić wysoką jakość systemu informatycznego będącego wynikiem prac badawczo-rozwojowych.
- Potrafi dokumentować architekturę i działanie systemu informatycznego będącego wynikiem prac badawczo-rozwojowych.

Treści programowe dla zajęć:

- Stworzenie zespołu projektowego uwzględniające społeczne aspekty pracy zespołowej. Wykonanie ćwiczeń mających na celu weryfikację współpracy w zespole. Podział ról w zespole projektowym.
- Opracowanie koncepcji projektu badawczo-rozwojowego. Konsultowanie koncepcji projektu z przedstawicielami podmiotów gospodarczych, instytucji administracyjnych lub jednostek badawczych. Opracowanie wizji systemu informatycznego realizującego koncepcję projektu. Konsultowanie wizji systemu informatycznego z jego interesariuszami.
- Przygotowanie środowiska prac projektowych: system kontroli wersji, narzędzia do planowania i kontroli wykonywanych zadań, system ciągłej integracji. Opracowanie dziennika projektu. Zaplanowanie zadań do wykonania w pierwszym sprincie.
- Wykonanie zadań zaplanowanych w pierwszym sprincie. Konsultowanie wyników pierwszego sprintu z interesariuszami systemu informatycznego. Retrospektywa pierwszego sprintu. Zaplanowanie zadań do wykonania w drugim sprincie.
- Wykonanie zadań zaplanowanych w drugim sprincie. Konsultowanie wyników drugiego sprintu z interesariuszami systemu informatycznego. Retrospektywa drugiego sprintu. Zaplanowanie zadań do wykonania w trzecim sprincie.
- Wykonanie zadań zaplanowanych w trzecim sprincie. Konsultowanie wyników pierwszego trzeciego z interesariuszami systemu informatycznego. Retrospektywa trzeciego sprintu. Zaplanowanie zadań do wykonania w celu przedstawienia prototypu systemu (czwarty sprint).
- Opracowanie demonstracyjnej wersji działającego prototypu systemu. Opracowanie dokumentacji prototypu systemu. Przekazanie demonstracyjnej wersji prototypu do testowania akceptacyjnego. Przekazanie dokumentacji prototypu jego interesariuszom.

Nazwa zajęć**: Projekt badawczo-rozwojowy 2**

- Po zakończeniu zajęć i potwierdzeniu osiągnięcia efektów uczenia się student:
- Potrafi zastosować znajomość aspektów społecznych pracy zespołowej w projekcie badawczo-rozwojowym.
- Potrafi dostarczać częściowe rezultaty prac wykonywanych w ramach projektu-badawczego.
- Potrafi zaimplementować użyteczny system informatyczny, będący wynikiem prac badawczo-rozwojowych.
- Potrafi zapewnić wysoką jakość systemu informatycznego będącego wynikiem prac badawczo-rozwojowych.
- Potrafi dokumentować architekturę i działanie systemu informatycznego będącego wynikiem prac badawczo-rozwojowych.
- Potrafi przeprowadzić proces testowania systemowego.
- Potrafi przeprowadzić proces testowania użyteczności.

Treści programowe dla zajęć:

- Analiza aspektów pracy zespołowej w ramach przedmiotu Projekt Badawczo-Rozwojowy 1. Analiza realizacji obowiązków poszczególnych członków zespołu implementującego system informatyczny w ramach przedmiotu Projekt Badawczo-Rozwojowy 1. Organizacja zespołu projektowego uwzględniające wyniki ww. analiz. Podział ról w zespole projektowym.
- Retrospektywa sprintów wykonanych w ramach przedmiotu Projekt Badawczo-Rozwojowy 1. Zaplanowanie zadań do wykonania w piątym sprincie.
- Wykonanie zadań zaplanowanych w piątym sprincie. Konsultowanie wyników piątego sprintu z interesariuszami systemu informatycznego. Retrospektywa piątego sprintu. Zaplanowanie zadań do wykonania w szóstym sprincie.
- Wykonanie zadań zaplanowanych w szóstym sprincie. Konsultowanie wyników szóstego sprintu z interesariuszami systemu informatycznego. Retrospektywa szóstego sprintu. Zaplanowanie zadań do wykonania w siódmym sprincie.
- Retrospektywa siódmego sprintu. Przeprowadzenie procesu testowania systemowego: sporządzenie planu testów, opracowanie przypadków testowych, przeprowadzenie testów, sporządzenie raportu z testów. Zaplanowanie zadań do wykonania w ósmym sprincie na podstawie raportu z testów.
- Retrospektywa ósmego sprintu. Przeprowadzenie procesu testowania użyteczności: sporządzenie planu testów, zdefiniowanie typów użytkowników, opracowanie person, opracowanie zadań do wykonania podczas testowania, zaplanowanie scenariuszy, przeprowadzenie testów użyteczności, zdefiniowanie obszarów problemowych, sporządzenie raportu testów.

- Opracowanie finalnej wersji systemu informatycznego, będącego wynikiem projektu badawczo-rozwojowego na podstawie raportu z testów użyteczności. Opracowanie dokumentacji systemu. Przekazanie końcowej wersji systemu informatycznego do testowania akceptacyjnego. Przekazanie dokumentacji systemu jego interesariuszom. Modyfikacja systemu na podstawie raportu z testów akceptacyjnych.

Nazwa zajęć**: Seminarium magisterskie I**
Po zakończeniu zajęć i potwierdzeniu osiągnięcia efektów uczenia się student:
- Potrafi wykorzystać pogłębioną wiedzę z zakresu wybranych obszarów informatyki do rozwiązywania problemów stawianych podczas zajęć.
- Zna podstawy metodologii prowadzenia badań w obszarze danego kierunku.
- Potrafi identyfikować problemy badawcze oraz pogłębia wiedzę z zakresu wybranych obszarów informatyki związanych z problematyką seminarium.
- Potrafi dobierać właściwe narzędzia badawcze, a także projektować warsztat badawczy.
- Potrafi projektować rozwiązania lub modyfikacje istniejących rozwiązań.
- Potrafi przygotować i zaprezentować krótkie opracowanie wybranego problemu w sposób zrozumiały dla innych uczestników. Potrafi redagować spójną i logiczną wypowiedź z wykorzystaniem poprawnej i terminologii.
- Potrafi poszerzać swoją wiedzę poprzez samodzielne poszukiwania w istniejących opracowaniach naukowych.
- Potrafi samodzielnie rozwiązać problem teoretyczny lub praktyczny.

Treści programowe dla zajęć:
- Prezentacja przez prowadzącego seminarium wyników badań prowadzonych aktualnie w grupach badawczych.
- Prezentacja przez studentów opracowań wybranych problemów i projektów badawczych zgodnych z potencjalną tematyką pracy magisterskiej.
- Określenie celu pracy dyplomowej oraz zakresu planowanych do wykonania w ramach niej badań.
- Przygotowanie warsztatu badawczego oraz konspektu pracy magisterskiej.

Nazwa zajęć**: Seminarium magisterskie II**
Po zakończeniu zajęć i potwierdzeniu osiągnięcia efektów uczenia się student:
- Potrafi wykorzystać pogłębioną wiedzę z zakresu wybranych obszarów informatyki do rozwiązywania problemów stawianych podczas zajęć.
- Zna podstawy metodologii prowadzenia badań w obszarze danego kierunku.
- Potrafi identyfikować problemy badawcze oraz pogłębia wiedzę z zakresu wybranych obszarów informatyki związanych z problematyką seminarium.
- Potrafi dobierać właściwe narzędzia badawcze, a także projektować warsztat badawczy.
- Potrafi projektować rozwiązania lub modyfikacje istniejących rozwiązań.
- Potrafi przygotować i zaprezentować krótkie opracowanie wybranego problemu w sposób zrozumiały dla innych uczestników. Potrafi redagować spójną i logiczną wypowiedź z wykorzystaniem poprawnej i terminologii.
- Potrafi poszerzać swoją wiedzę poprzez samodzielne poszukiwania w istniejących opracowaniach naukowych.
- Potrafi samodzielnie rozwiązać problem teoretyczny lub praktyczny.

Treści programowe dla zajęć:
- Prezentacja przez studentów opracowań wybranych problemów i projektów badawczych zgodnych z zadaniami realizowanymi w pracy magisterskiej.
- Raportowanie i prezentowanie wykonania badań teoretycznych związanych z tematyką pracy magisterskiej.
- Raportowanie i prezentowanie wykonania prac rozwojowych związanych z tematyką pracy magisterskiej.
- Dostarczenie ustalonego fragmentu pracy magisterskiej.

Nazwa zajęć**: Seminarium magisterskie III**
Po zakończeniu zajęć i potwierdzeniu osiągnięcia efektów uczenia się student:
- Potrafi wykorzystać pogłębioną wiedzę z zakresu wybranych obszarów informatyki do rozwiązywania problemów stawianych podczas zajęć.
- Zna podstawy metodologii prowadzenia badań w obszarze danego kierunku.

- Potrafi identyfikować problemy badawcze oraz pogłębia wiedzę z zakresu wybranych obszarów informatyki związanych z problematyką seminarium.
- Potrafi dobierać właściwe narzędzia badawcze, a także projektować warsztat badawczy.
- Potrafi projektować rozwiązania lub modyfikacje istniejących rozwiązań.
- Potrafi przygotować i zaprezentować krótkie opracowanie wybranego problemu w sposób zrozumiały dla innych uczestników. Potrafi redagować spójną i logiczną wypowiedź z wykorzystaniem poprawnej i terminologii.
- Potrafi poszerzać swoją wiedzę poprzez samodzielne poszukiwania w istniejących opracowaniach naukowych.
- Potrafi samodzielnie rozwiązać problem teoretyczny lub praktyczny.

Treści programowe dla zajęć:

- Prezentacja przez studentów opracowań wybranych problemów i projektów badawczych zgodnych z zadaniami realizowanymi w pracy magisterskiej.
- Raportowanie i prezentowanie wykonania badań teoretycznych związanych z tematyką pracy magisterskiej.
- Raportowanie i prezentowanie wykonania prac rozwojowych związanych z tematyką pracy magisterskiej.
- Dostarczenie kolejnych rozdziałów pracy magisterskiej.
- Dostarczenie końcowej wersji pracy magisterskiej.