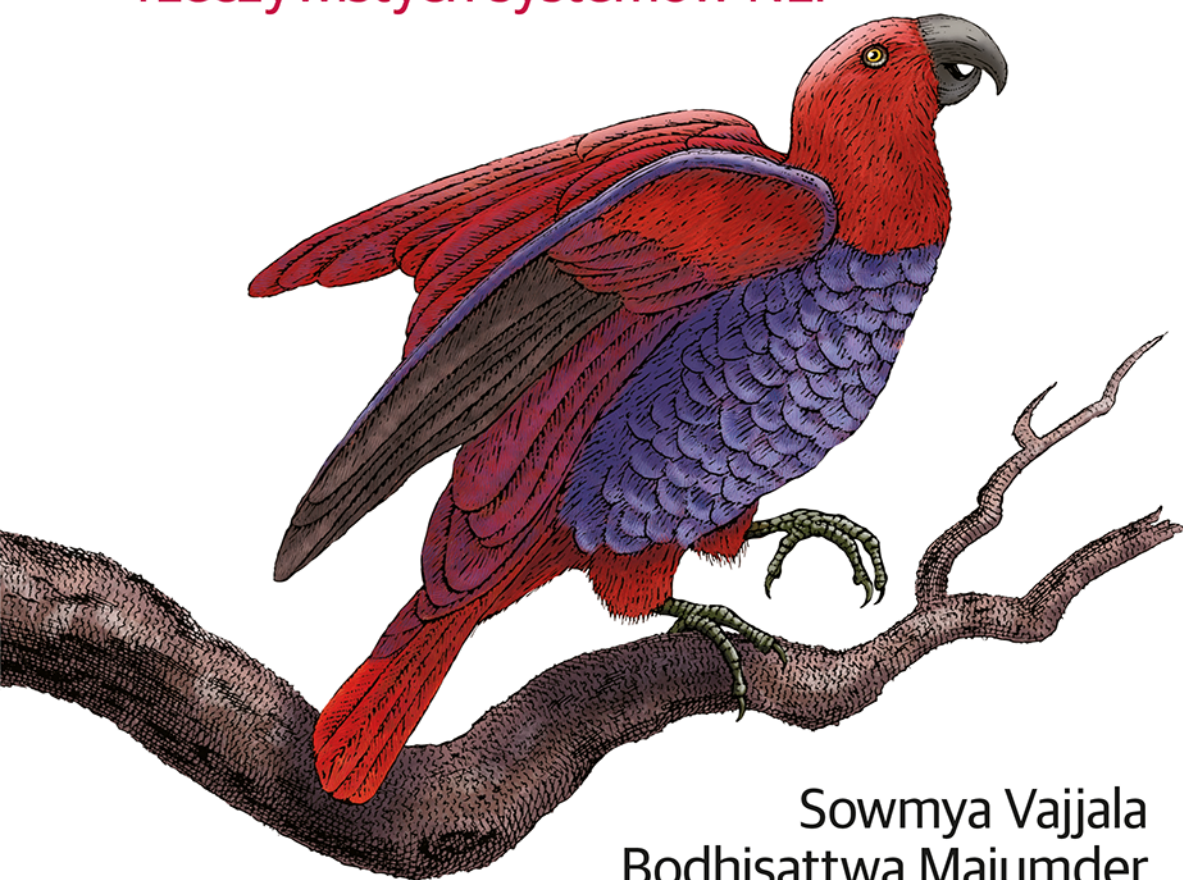


O'REILLY®

Przetwarzanie języka naturalnego w praktyce

Przewodnik po budowie
rzeczywistych systemów NLP



Sowmya Vajjala
Bodhisattwa Majumder
Anuj Gupta
Harshit Surana

Helion 

Tytuł oryginału: Practical Natural Language Processing:
A Comprehensive Guide to Building Real-World NLP Systems

Tłumaczenie: Grzegorz Werner

ISBN: 978-83-8322-726-9

© 2023 Helion S.A.

Authorized Polish translation of the English edition of *Practical Natural Language Processing*
ISBN 9781492054054 © 2020 Anuj Gupta, Bodhisattwa Prasad Majumder, Sowmya Vajjala,
and Harshit Surana.

This translation is published and sold by permission of O'Reilly Media, Inc.,
which owns or controls all rights to publish and sell the same.

All rights reserved. No part of this book may be reproduced or transmitted in any
form or by any means, electronic or mechanical, including photocopying, recording
or by any information storage retrieval system, without permission from the Publisher.

Wszelkie prawa zastrzeżone. Nieautoryzowane rozpowszechnianie całości
lub fragmentu niniejszej publikacji w jakiegokolwiek postaci jest zabronione.
Wykonywanie kopii metodą kserograficzną, fotograficzną, a także kopiowanie
książki na nośniku filmowym, magnetycznym lub innym powoduje naruszenie
praw autorskich niniejszej publikacji.

Wszystkie znaki występujące w tekście są zastrzeżonymi znakami firmowymi
bądź towarowymi ich właścicieli.

Autor oraz wydawca dołożyli wszelkich starań, by zawarte w tej książce informacje
były kompletne i rzetelne. Nie biorą jednak żadnej odpowiedzialności ani za ich
wykorzystanie, ani za związane z tym ewentualne naruszenie praw patentowych
lub autorskich. Autor oraz wydawca nie ponoszą również żadnej odpowiedzialności
za ewentualne szkody wynikłe z wykorzystania informacji zawartych w książce.

Drogi Czytelniku!

Jeżeli chcesz ocenić tę książkę, zajrzyj pod adres

<https://helion.pl/user/opinie/przjen>

Możesz tam wpisać swoje uwagi, spostrzeżenia, recenzję.

Helion S.A.

ul. Kościuszki 1c, 44-100 Gliwice

tel. 32 230 98 63

e-mail: helion@helion.pl

WWW: <https://helion.pl> (księgarnia internetowa, katalog książek)

Printed in Poland.

- [Kup książkę](#)
- [Poleć książkę](#)
- [Oceń książkę](#)

- [Księgarnia internetowa](#)
- [Lubię to! » Nasza społeczność](#)

Spis treści

Słowo wstępne 15

Przedmowa 17

CZĘŚĆ I. Podstawy 27

1. NLP — elementarz 29

NLP w rzeczywistym świecie 30

Zadania NLP 32

Czym jest język? 33

Podstawowe elementy języka 34

Dlaczego NLP jest trudnym wyzwaniem? 37

Uczenie maszynowe, uczenie głębokie i NLP — przegląd 38

Podejścia do NLP 40

NLP oparte na heurystyce 40

Uczenie maszynowe w NLP 42

Uczenie głębokie w NLP 46

Dlaczego uczenie głębokie nie jest jeszcze „srebrną kulą” NLP? 51

Przewodnik po NLP — agenty konwersacyjne 54

Podsumowanie 55

Bibliografia 56

2. Potok NLP 59

Pozyskiwanie danych 61

Ekstrakcja i oczyszczanie tekstu 64

Parsowanie i oczyszczanie HTML-a 64

Normalizacja Unikodu 66

Poprawianie pisowni 67

Poprawianie błędów specyficzne dla systemu 68

Przetwarzanie wstępne	70
Czynności wstępne	70
Częste czynności	73
Inne czynności	76
Przetwarzanie zaawansowane	77
Inżynieria cech	80
Klasyczny potok NLP/ML	80
Potok DL	82
Modelowanie	82
Zacznij od prostej heurystyki	82
Budowanie modelu	84
Budowanie ostatecznego modelu	84
Ewaluacja	87
Ewaluacja wewnętrzna	88
Ewaluacja zewnętrzna	90
Fazy następujące po modelowaniu	91
Wdrażanie	91
Monitorowanie	92
Aktualizowanie modelu	92
Praca z innymi językami	93
Studium przypadku	94
Podsumowanie	95
Bibliografia	95
3. Reprezentacja tekstu	99
Modele przestrzeni wektorowej	102
Proste metody wektoryzacji	102
Kodowanie one-hot	103
Worek słów	104
Worek n-gramów	106
TF-IDF	107
Reprezentacje rozproszone	109
Osadzenia słów	110
Ponad słowa	118
Reprezentacje rozproszone na poziomach wyższych niż słowa i znaki	120
Uniwersalne reprezentacje tekstu	121
Wizualizacja osadzeń	123
Ręcznie utworzone reprezentacje cech	126
Podsumowanie	128
Bibliografia	128

4. Klasyfikacja tekstu	133
Zastosowania	134
Potok budowania systemów klasyfikacji tekstu	137
Prosty klasyfikator bez potoku klasyfikacji tekstu	138
Używanie istniejących interfejsów API do klasyfikacji tekstu	139
Jeden potok, wiele klasyfikatorów	139
Naiwny klasyfikator bayesowski	140
Regresja logistyczna	144
Maszyna wektorów nośnych	145
Osadzenia neuronowe w klasyfikacji tekstu	147
Osadzenia słów	147
Osadzenia podsłów i fastText	148
Osadzenia dokumentów	150
Uczenie głębokie w klasyfikacji tekstu	152
Sieci CNN do klasyfikacji tekstu	155
Sieci LSTM do klasyfikacji tekstu	156
Klasyfikacja tekstu z wykorzystaniem dużych, wstępnie wytrenowanych modeli językowych	157
Interpretacja modeli klasyfikacji tekstu	158
Wyjaśnianie prognoz klasyfikatora za pomocą Lime'a	159
Uczenie się bez danych lub na mniejszej ilości danych i adaptowanie modeli do nowych dziedzin	160
Brak danych treningowych	160
Mało danych treningowych — nauka aktywna i adaptacja dziedzinowa	161
Studium przypadku — obsługa zgłoszeń problemów	163
Praktyczne rady	165
Podsumowanie	167
Bibliografia	167
5. Ekstrakcja informacji	170
Zastosowania IE	171
Zadania IE	172
Ogólny potok IE	174
Ekstrakcja fraz kluczowych	175
Implementowanie KPE	176
Praktyczne rady	177
Rozpoznawanie nazwanych encji	177
Budowanie systemu NER	179
NER z wykorzystaniem istniejącej biblioteki	183

NER z wykorzystaniem nauki aktywnej	183
Praktyczne rady	184
Ujednoznacznianie i łączenie nazwanych encji	185
NEL z wykorzystaniem Azure API	186
Ekstrakcja relacji	188
Podejścia do RE	188
RE z wykorzystaniem Watson API	190
Inne zaawansowane zadania IE	191
Ekstrakcja informacji temporalnych	192
Ekstrakcja zdarzeń	193
Uzupełnianie szablonów	194
Studium przypadku	196
Podsumowanie	198
Bibliografia	199
6. Czatboty	203
Zastosowania	204
Prosty bot FAQ	205
Taksonomia czatbotów	206
Dialog ukierunkowany na cel	208
Pogawędki	208
Potok budowania systemów dialogowych	208
Szczegóły systemu dialogowego	210
Czatbot PizzaStop	211
Szczegółowa analiza komponentów systemu dialogowego	220
Klasyfikacja aktu dialogowego	221
Identyfikacja slotów	222
Generowanie odpowiedzi	223
Systemy dialogowe z przykładami kodu	223
Inne potoki dialogowe	228
Podejście kompleksowe	228
Generowanie dialogu poprzez uczenie głębokie ze wzmocnieniem	228
Człowiek w pętli	229
Rasa NLU	231
Studium przypadku — polecenie przepisów	233
Korzystanie z istniejących platform	234
Czatboty generatywne o strukturze otwartej	236
Podsumowanie	237
Bibliografia	238

7. Tematy w skrócie	240
Wyszukiwanie i zwracanie informacji	242
Komponenty wyszukiwarki	244
Typowy potok wyszukiwarki korporacyjnej	246
Budowanie wyszukiwarki — przykład	247
Studium przypadku — wyszukiwarka dla księgarni	248
Modelowanie tematyczne	249
Trenowanie modelu tematycznego — przykład	253
Co dalej?	254
Streszczanie tekstu	255
Zastosowania streszczania	256
Konfigurowanie narzędzia streszczającego — przykład	257
Praktyczne rady	257
Systemy rekomendujące dane tekstowe	259
Tworzenie systemu rekomendacji książek — przykład	260
Praktyczne rady	261
Tłumaczenie maszynowe	262
Używanie interfejsu API do tłumaczenia maszynowego — przykład	262
Praktyczne rady	263
Systemy odpowiadania na pytania	264
Budowanie własnego systemu odpowiadania na pytania	266
Poszukiwanie głębszych odpowiedzi	266
Podsumowanie	267
Bibliografia	267

CZĘŚĆ III. Praktyka **271**

8. Media społecznościowe	273
Aplikacje	274
Unikatowe wyzwania	276
Przetwarzanie języka naturalnego w danych społecznościowych	281
Chmura wyrazowa	281
Jonizator SMTD	282
Popularne tematy	283
Odczucia użytkowników Twittera	285
Wstępne przetwarzanie danych SMTD	287
Reprezentacja tekstu w SMTD	290
Obsługa klienta w kanałach społecznościowych	293
Memy i fake newsy	295
Identyfikowanie memów	295
Fake newsy	296

Podsumowanie	298
Bibliografia	298
9. E-commerce i handel detaliczny	302
Katalog e-commerce	302
Analiza recenzji	303
Wyszukiwanie produktów	303
Rekomendacje produktów	304
Wyszukiwanie w e-commerce	304
Budowanie katalogu e-commerce	306
Ekstrakcja atrybutów	306
Kategoryzacja i taksonomia produktów	311
Wzbogacanie produktów	313
Deduplikacja i dopasowywanie produktów	315
Analiza recenzji	317
Analiza odczuć	318
Aspektowa analiza odczuć	319
Łączenie ocen ogólnych z aspektami	321
Rozumienie aspektów	322
Rekomendacje w e-commerce	324
Studium przypadku — produkty substytucyjne i komplementarne	325
Podsumowanie	328
Bibliografia	328
10. Opieka zdrowotna, finanse i prawo	330
Opieka zdrowotna	330
Dokumentacja zdrowotna i medyczna	331
Ustalanie priorytetów i rozliczanie pacjentów	331
Nadzór farmakologiczny	332
Systemy wspomaganie decyzji klinicznych	332
Asystenty zdrowotne	333
Elektroniczna dokumentacja medyczna	334
Monitorowanie zdrowia psychicznego	342
Ekstrakcja i analiza informacji medycznych	344
Finanse i prawo	346
Zastosowania NLP w finansach	348
NLP w krajobrazie prawnym	350
Podsumowanie	352
Bibliografia	354

11. Kompleksowy proces NLP	359
Powrót do potoku NLP — wdrażanie oprogramowania NLP	360
Przykładowy scenariusz	361
Budowanie i utrzymywanie dojrzałego systemu	363
Znajdowanie lepszych cech	363
Iteracyjne rozwijanie istniejących modeli	364
Odtwarzalność kodu i modelu	365
Rozwiązywanie problemów i interpretowalność	366
Monitorowanie	368
Minimalizowanie długu technicznego	369
Automatyzacja uczenia maszynowego	370
Proces data science	374
Proces KDD	374
Proces Microsoft Team Data Science	375
Droga do sukcesu AI w Twojej organizacji	377
Zespół	377
Właściwy problem i właściwe oczekiwania	378
Dane i czas	379
Dobry proces	380
Inne aspekty	381
Spojrzenie poza horyzont	382
Ostatnie słowa	385
Bibliografia	386
Skorowidz	391

NLP — elementarz

*Język to nie tylko słowa. To kultura, tradycja, jednoczenie społeczności,
cała historia, która określa, czym społeczność jest.
Wszystko to jest odzwierciedlone w języku.*

— Noam Chomsky

Wyobraź sobie hipotetyczną osobę, Jana Kowalskiego. Jest on dyrektorem ds. technologii w szybko rozwijającym się start-upie. Pewnego dnia Jan budzi się i odbywa następującą rozmowę ze swoim asystentem cyfrowym:

Jan: „Jaka jest dziś pogoda?”

Asystent cyfrowy: „Na zewnątrz jest 27 stopni i nie pada”.

Jan: „Jakie są moje plany na dziś?”

Asystent cyfrowy: „Masz zebranie strategiczne o 16.00 i konferencję o 17.30. Zważywszy na sytuację na drogach, powinieneś wyjechać do biura najpóźniej o 8.15”.

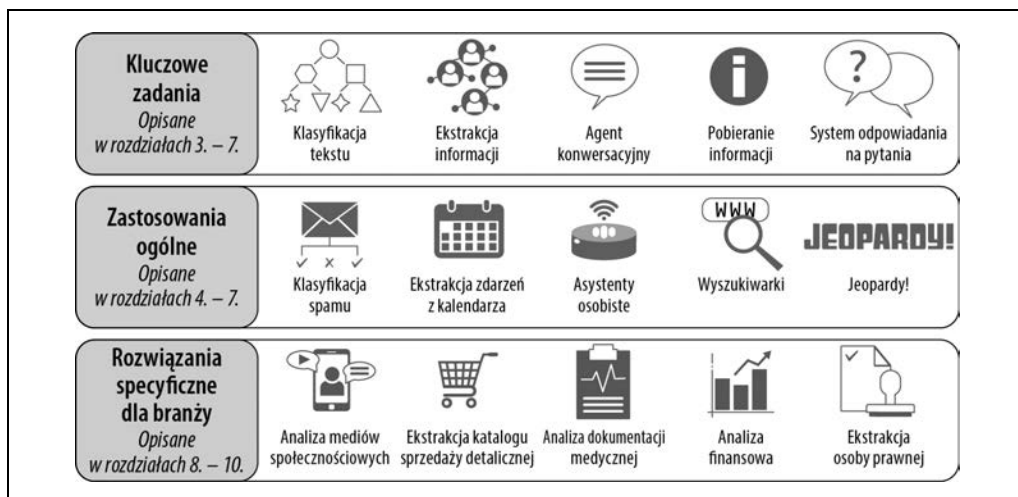
Kiedy Jan się ubiera, pyta swojego asystenta o dobór ubrań.

Jan: „Co powinienem dziś założyć?”

Asystent cyfrowy: „Biały wydaje się dobrym pomysłem”.

Być może prowadziłeś podobne rozmowy z inteligentnymi asystentami, takimi jak Amazon Alexa, Google Home lub Apple Siri. Mówimy do nich nie w języku programowania, ale w języku naturalnym — tym, w którym komunikujemy się na co dzień. Język naturalny jest podstawowym środkiem komunikacji między ludźmi od niepamiętnych czasów. Komputery jednak przetwarzają tylko dane binarne, tzn. zera i jedyńki. Choć możemy reprezentować dane językowe w postaci binarnej, jak sprawić, żeby maszyny rozumiały język? Tu na scenę wkracza przetwarzanie języka naturalnego (ang. *natural language processing* — NLP). Jest to dziedzina informatyki zajmująca się metodami analizowania, modelowania i rozumienia ludzkiego języka. Każda inteligentna aplikacja sterowana ludzkim językiem wykorzystuje jakąś postać NLP. W tej książce wyjaśnimy, czym jest NLP i jak używać go do budowania i skalowania inteligentnych aplikacji. Ze względu na otwartą naturę problemów NLP istnieją dziesiątki alternatywnych rozwiązań każdego problemu. Ta książka pomoże Ci poruszać się po labiryncie dostępnych opcji i wybrać tę, która najlepiej pasuje do danego problemu.

Celem tego rozdziału jest krótkie omówienie NLP, zanim zaczniemy zagłębiać się w praktyczną implementację rozwiązań opartych na NLP. Zaczniemy od przeglądu licznych aplikacji NLP w rzeczywistych scenariuszach, a następnie omówimy różne zadania, które stanowią podstawę budowania aplikacji NLP. Potem spróbujemy zrozumieć język z perspektywy NLP i wyjaśnić, dlaczego NLP jest trudne. Następnie powiemy kilka słów o heurystyce, uczeniu maszynowym i uczeniu głębokim, po czym przedstawimy kilka algorytmów powszechnie używanych w NLP. Wreszcie opiszemy elementy typowej aplikacji NLP. Rozdział zakończymy przeglądem pozostałych tematów z książki. Na rysunku 1.1 pokazano organizację rozdziałów według rozmaitych zadań i zastosowań NLP.



Rysunek 1.1. Zadania i zastosowania NLP

Zacznijmy od przyjrzenia się popularnym aplikacjom, których używamy na co dzień i których istotnym składnikiem jest jakaś forma NLP.

NLP w rzeczywistym świecie

NLP jest istotnym składnikiem licznych aplikacji, których używamy w codziennym życiu. W tym podrozdziale przedstawimy niektóre kluczowe aplikacje i przyjrzymy się typowym zadaniom realizowanym przez różnorodne aplikacje NLP. Wspomnimy o zastosowaniach pokazanych na rysunku 1.1, które zostaną opisane dokładniej dalej w tej książce.

Kluczowe zastosowania:

- Platformy e-mail, np. Gmail, Outlook, wykorzystują NLP do realizowania różnych funkcji, takich jak klasyfikacja spamu, umieszczanie wiadomości w skrzynce priorytetowej, ekstrakcja zdarzeń kalendarzowych, autouzupełnianie itd. Niektóre z tych funkcji omówimy szczegółowo w rozdziałach 4. i 5.
- Asystenty głosowe, takie jak Apple Siri, Google Assistant, Microsoft Cortana i Amazon Alexa, wykorzystują różne techniki NLP do interakcji z użytkownikiem, rozpoznawania poleceń

użytkownika i odpowiedniego reagowania na nie. Kluczowe aspekty takich systemów opiszemy w rozdziale 6., kiedy będziemy mówić o czatbotach.

- Współczesne wyszukiwarki, takie jak Google i Bing, które są podstawą dzisiejszego internetu, intensywnie wykorzystują NLP do różnych podzadań, takich jak rozpoznawanie zapytań, rozwijanie zapytań, odpowiadanie na pytania, wyszukiwanie informacji oraz klasyfikowanie i grupowanie wyników. Niektóre z tych podzadań omówimy w rozdziale 7.
- Usługi tłumaczenia maszynowego, takie jak Google Translate, Bing Microsoft Translator i Amazon Translate, są coraz powszechniej wykorzystywane w życiu codziennym i scenariuszach biznesowych. O tłumaczeniu maszynowym wspomnimy w rozdziale 7.

Inne zastosowania:

- Organizacje z różnych branż analizują swoje kanały społecznościowe, aby lepiej wsłuchiwać się w głos klienta. Omówimy to w rozdziale 8.
- NLP używa się do rozwiązywania wielu różnorodnych problemów na platformach e-commerce, takich jak Amazon, od ekstrakowania istotnych informacji z opisów produktów do rozumienia recenzji użytkowników. Zostanie to opisane szczegółowo w rozdziale 9.
- Postępy w NLP wykorzystuje się do rozwiązywania problemów w takich dziedzinach jak opieka zdrowotna, finanse i prawo. Będzie to tematem rozdziału 10.
- Firmy takie jak Arria [1] pracują nad technikami NLP do automatycznego generowania raportów w różnych dziedzinach, od prognozowania pogody do usług finansowych.
- NLP jest kluczowym elementem narzędzi do sprawdzania gramatyki i pisowni, takich jak usługa Grammarly oraz sprawdzanie pisowni w programach Microsoft Word i Google Docs.
- *Jeopardy!* to popularny teleturniej (w Polsce znany jako *Va banque*). Uczestnicy otrzymują wskazówki w formie haseł i udzielają odpowiedzi sformułowanych jako pytania. Firma IBM zbudowała system Watson AI, aby konkurować z najlepszymi graczami. Watson zdobył pierwszą nagrodę w wysokości miliona dolarów, więcej niż mistrz świata. Został zbudowany z wykorzystaniem technik NLP i jest przykładem bota NLP, który wygrał mistrzostwa świata.
- NLP używa się w różnych narzędziach i technologiach edukacyjnych oraz oceniających, np.: w systemach automatycznego oceniania egzaminów takich jak Graduate Record Examination (GRE), w systemach wykrywania plagiatów (np. Turnitin), w inteligentnych systemach nauczania oraz w aplikacjach do nauki języków takich jak Duolingo.
- NLP używa się do budowania dużych baz wiedzy, takich jak Google Knowledge Graph, które przydają się do wyszukiwania informacji i odpowiadania na pytania.

Lista ta nie jest bynajmniej wyczerpująca. NLP coraz częściej używa się do innych celów, a kiedy piszę te słowa, nieustannie pojawiają się nowe zastosowania NLP. My skupimy się przede wszystkim na ideach stojących za tymi aplikacjami. Aby pokazać Ci, czego dowiesz się z tej książki, i przedstawić niuanse związane z tworzeniem oprogramowania NLP, opiszemy niektóre kluczowe zadania NLP, które stanowią podstawę wielu aplikacji i zastosowań branżowych.

Zadania NLP

Istnieje zbiór podstawowych zadań, które pojawiają się często w różnych projektach NLP. Ze względu na ich powtarzalną i fundamentalną naturę zostały one szczegółowo zbadane. Ich dobre zrozumienie przygotuje Cię do budowania aplikacji NLP dla różnych branż. (Niektóre pokazaliśmy wcześniej na rysunku 1.1). Oto ich krótki opis:

Modelowanie językowe

Zadanie to polega na przewidywaniu, jakie będzie następne słowo w zdaniu, na podstawie poprzednich słów. Celem tego zadania jest określenie prawdopodobieństwa sekwencji słów występujących w danym języku. Modelowanie językowe przydaje się w rozwiązywaniu wielu problemów, takich jak rozpoznawanie mowy, optyczne rozpoznawanie znaków, rozpoznawanie pisma ręcznego, tłumaczenie maszynowe i sprawdzanie pisowni.

Klasyfikacja tekstu

To zadanie polega na przypisywaniu tekstu do określonej kategorii na podstawie jego treści. Klasyfikacja tekstu jest zdecydowanie najpopularniejszym zadaniem w NLP i używa się jej w różnych narzędziach, od identyfikowania spamu do analizowania odczuć.

Ekstrakcja informacji

Jak wskazuje nazwa, zadanie to polega na wyodrębnianiu z tekstu istotnych informacji, np. zdarzeń kalendarzowych z wiadomości e-mail albo nazwisk osób wspomnianych w poście w mediach społecznościowych.

Pobieranie informacji

To zadanie polega na znajdowaniu dokumentów związanych z kwerendą użytkownika. Dobrze znanymi zastosowaniami pobierania informacji są takie aplikacje jak wyszukiwarka Google.

Agent konwersacyjny

To zadanie polega na budowaniu systemów dialogowych, które mogą prowadzić rozmowę w ludzkim języku. Typowymi zastosowaniami tego zadania są Alexa, Siri itp.

Streszczanie tekstu

To zadanie polega na tworzeniu krótkich streszczeń dłuższych dokumentów przy zachowaniu zasadniczej treści i ogólnego znaczenia tekstu.

Odpowiadanie na pytania

To zadanie polega na budowaniu systemu, który może automatycznie odpowiadać na pytania zadawane w języku naturalnym.

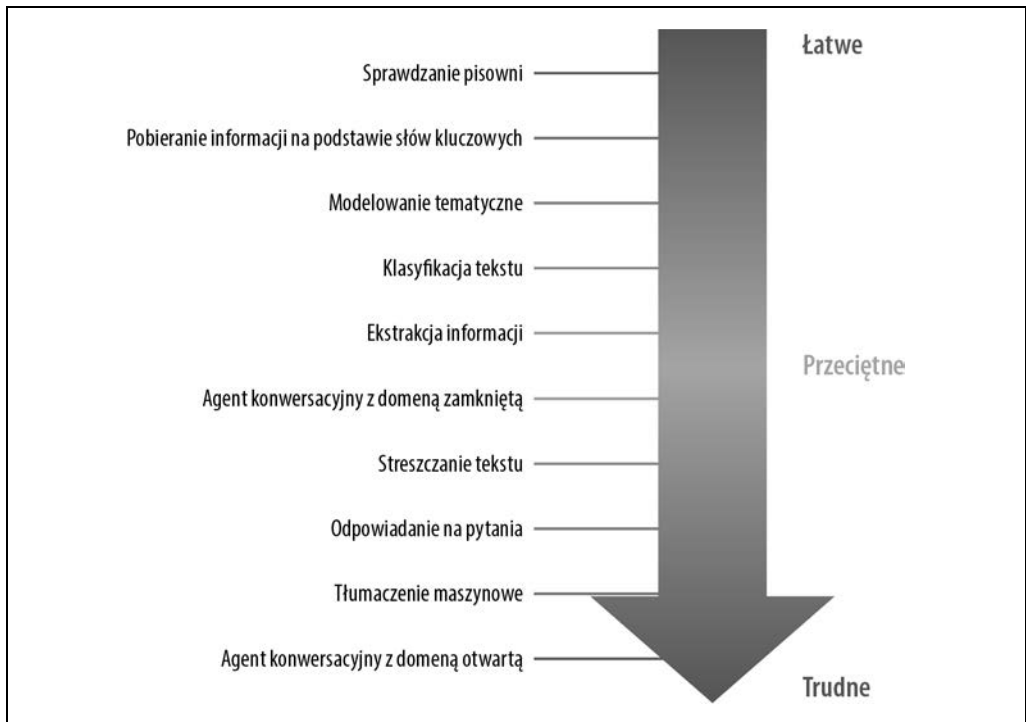
Tłumaczenie maszynowe

To zadanie polega na przekładaniu tekstu z jednego języka na drugi. Typowym zastosowaniem tego zadania są takie narzędzia jak Google Translate.

Modelowanie tematyczne

To zadanie polega na określaniu tematycznej struktury dużego zbioru dokumentów. Modelowanie tematyczne to popularne narzędzie „wydobywcze” używane w wielu dziedzinach, od literatury do bioinformatyki.

Na rysunku 1.2 przedstawiono te zadania według względnej trudności opracowywania kompleksowych rozwiązań.



Rysunek 1.2. Zadania NLP ułożone według względnego stopnia trudności

W pozostałych rozdziałach książki poznasz wyzwania, które wiążą się z tymi zadaniami, i nauczysz się tworzyć rozwiązania, które skutecznie działają w pewnych sytuacjach (nawet w przypadku trudnych zadań pokazanych na rysunku). Aby osiągnąć ten cel, trzeba zrozumieć naturę ludzkiego języka i problemy związane z automatycznym przetwarzaniem języka. Zagadnienia te zostaną krótko opisane w następnym dwóch podrozdziałach.

Czym jest język?

Język to ustrukturyzowany system komunikacji, który wykorzystuje skomplikowane kombinacje elementów składowych, takich jak znaki, słowa, zdania itd. Systematyczne badanie języka określa się mianem lingwistyki. Aby badać NLP, trzeba zrozumieć niektóre pojęcia lingwistyczne dotyczące struktury języka. W tym podrozdziale przedstawimy je i wyjaśnimy, jak wiążą się z niektórymi wspomnianymi wcześniej zadaniami NLP.

Możemy myśleć o języku jako o tworze złożonym z czterech podstawowych elementów: fonemów, morfemów i leksemów, składni oraz kontekstu. Aplikacje NLP wymagają znajomości różnych poziomów tych elementów, od podstawowych dźwięków języka (fonemów) do tekstów wyrażających pewien sens (kontekstu).

Na rysunku 1.3 pokazano podstawowe elementy języka, ich zakres i kilka wspomnianych wcześniej aplikacji NLP, które wymagają tej wiedzy. Niektóre użyte terminy, których nie wprowadzono wcześniej w tym rozdziale (np. analiza składniowa, osadzenia słów), zostaną wyjaśnione w kolejnych rozdziałach.



Rysunek 1.3. Podstawowe elementy języka i ich zastosowania

Podstawowe elementy języka

Opiszmy najpierw, czym są poszczególne elementy języka, aby przedstawić kontekst wyzwań związanych z NLP.

Fonemy

Fonemy to najmniejsze jednostki dźwiękowe języka. Same w sobie mogą nie mieć żadnego znaczenia, ale tworzą znaczenie, kiedy zostaną wypowiedziane w kombinacji z innymi fonemami. Na przykład standardowy język angielski ma 44 fonemy reprezentowane albo przez pojedyncze litery, albo przez kombinacje liter [2]. Na rysunku 1.4 pokazano te fonemy wraz z przykładowymi słowami. Fonemy są szczególnie istotne w aplikacjach, które wymagają rozumienia mowy, takich jak rozpoznawanie mowy, transkrypcja mowy i przekształcanie tekstu na mowę.

Morfemy i leksemy

Morfem to najmniejsza jednostka języka, która ma znaczenie. Morfemy powstają z połączenia fonemów. Nie wszystkie morfemy są słowami, ale wszystkie przedrostki i przyrostki są morfemami. Na przykład w słowie „multimedia” „multi-” nie jest słowem, ale przedrostkiem, który zmienia znaczenie słowa „media”. „Multi-” to morfem. Na rysunku 1.5 pokazano kilka słów i ich morfemy. W słowach takich jak „cats” i „unbreakable” poszczególne morfemy łączą się ze sobą bez żadnych zmian, natomiast w słowach takich jak „tumbling” i „unreliability” ulegają pewnym przekształceniom.

Fonemy spółgłoskowe z przykładowymi słowami		Fonemy samogłoskowe z przykładowymi słowami	
1. /b/ – bat	13. /s/ – sun	1. /a/ – ant	13. /oi/ – coin
2. /k/ – cat	14. /t/ – tap	2. /e/ – egg	14. /ar/ – farm
3. /d/ – dog	15. /v/ – van	3. /i/ – in	15. /or/ – for
4. /f/ – fan	16. /w/ – wig	4. /o/ – on	16. /ur/ – hurt
5. /g/ – go	17. /y/ – yes	5. /u/ – up	17. /air/ – fair
6. /h/ – hen	18. /z/ – zip	6. /ai/ – rain	18. /ear/ – dear
7. /j/ – jet	19. /sh/ – shop	7. /ee/ – feet	19. /ure/ – sure
8. /l/ – leg	20. /ch/ – chip	8. /igh/ – night	20. /ə/ – corner („schwa” – nieakcentowana samogłoska o brzmieniu zbliżonym do /u/)
9. /m/ – map	21. /th/ – thin	9. /oa/ – boat	
10. /n/ – net	22. /th/ – then	10. /oo/ – boot	
11. /p/ – pen	23. /ng/ – ring	11. /oo/ – look	
12. /r/ – rat	24. /zh/ – vision	12. /ow/ – cow	

Rysunek 1.4. Fonemy i przykładowe słowa

unbreakable <i>un + break + able</i>	cats <i>cat + s</i>
tumbling <i>tumble + ing</i>	unreliability <i>un + rely + able + ity</i>

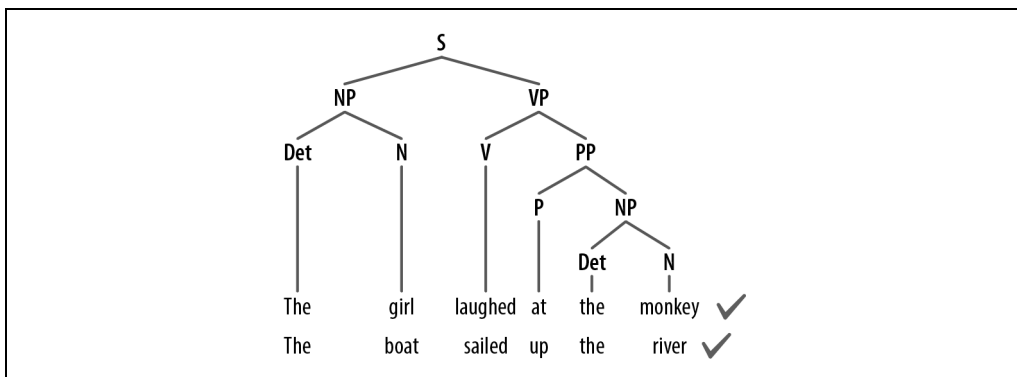
Rysunek 1.5. Przykładowe morfemy

Leksemy to strukturalne odmiany morfemów, które są powiązane znaczeniowo. Na przykład „run” i „running” należą do tej samej formy leksemowej. Analiza morfologiczna, która bada strukturę słów poprzez wyodrębnianie morfemów i leksemów, stanowi fundament wielu zadań NLP, takich jak tokenizacja, określanie tematów słów, nauka osadzeń słów i tagowanie części mowy, które zostaną przedstawione w następnym rozdziale.

Składnia

Składnia to zbiór reguł, które umożliwiają konstruowanie poprawnych gramatycznie zdań ze słów i z fraz danego języka. W lingwistyce istnieje wiele sposobów reprezentowania struktury składniowej. Często przedstawia się ją w postaci drzewa składniowego. Na rysunku 1.6 pokazano przykładowe drzewo składniowe dwóch angielskich zdań.

Drzewo odzwierciedla hierarchiczną strukturę języka ze słowami na najniższym poziomie, po których następują tagi części mowy, dalej frazy, a wreszcie zdanie na najwyższym poziomie. Zdania z rysunku 1.6 mają podobną strukturę składniową, a zatem podobne drzewo składniowe. W tej reprezentacji N



Rysunek 1.6. Struktura dwóch podobnych składniowo zdań

oznacza rzeczownik (ang. *noun*), V — czasownik (ang. *verb*), a P — przyimek (ang. *preposition*). Fraza nominalna (rzeczownikowa) jest oznaczona przez NP (ang. *noun phrase*), a fraza werbalna (czasownikowa) — przez VP (ang. *verb phrase*). Dwie frazy nominalne to „The girl” i „The boat”, a dwie frazy werbalne to „laughed at the monkey” i „sailed up the river”. Struktura składniowa podlega regułom gramatycznym języka (np. „Zdanie składa się z NP i VP”), a te z kolei wpływają na niektóre fundamentalne zadania przetwarzania języka, takie jak analiza składniowa, zwana również parsowaniem. Parsowanie to zadanie NLP polegające na automatycznym konstruowaniu takich drzew. Wykorzystuje się je m.in. do ekstrakcji encji i ekstrakcji relacji — zadań NLP, które omówimy dokładniej w rozdziale 5. Zauważ, że opisana wyżej struktura składniowa jest specyficzna dla języka angielskiego. Składnia jednego języka może się bardzo różnić od składni drugiego, dlatego sposoby przetwarzania tych języków bywają odmienne.

Kontekst

Kontekst określa, jak różne części języka łączą się ze sobą w celu przekazania jakiegoś znaczenia. Kontekst obejmuje odniesienia długoterminowe, wiedzę o świecie i zdrowy rozsądek obok dosłownego znaczenia słów i fraz. Sens zdania może się zmieniać w zależności od kontekstu, ponieważ słowa i frazy czasem mają wiele znaczeń. Ogólnie rzecz biorąc, kontekst obejmuje semantykę i pragmatykę. Semantyka to bezpośrednie znaczenie słów i fraz wyjętych z zewnętrznego kontekstu. Pragmatyka dodaje wiedzę o świecie i kontekst zewnętrzny, pozwalając nam domyślić się podtekstu. Złożone zastosowania NLP, takie jak wykrywanie sarkazmu, streszczanie i modelowanie tematyczne, to przykłady zadań, które intensywnie używają kontekstu.

Lingwistyka to nauka o języku, a zatem bardzo rozległa dziedzina, a my przedstawiliśmy tylko kilka podstawowych pojęć, aby zilustrować rolę wiedzy lingwistycznej w NLP. Różne zadania NLP wymagają różnego stopnia wiedzy o tych podstawowych elementach budulcowych języka. Zainteresowany czytelnik może sięgnąć po książki Emily Bender [3, 4] poświęcone lingwistyce w NLP. Teraz, kiedy wiemy już, z czego składa się język, zastanówmy się, dlaczego komputery mają problemy ze zrozumieniem języka i co sprawia, że NLP jest trudnym wyzwaniem.

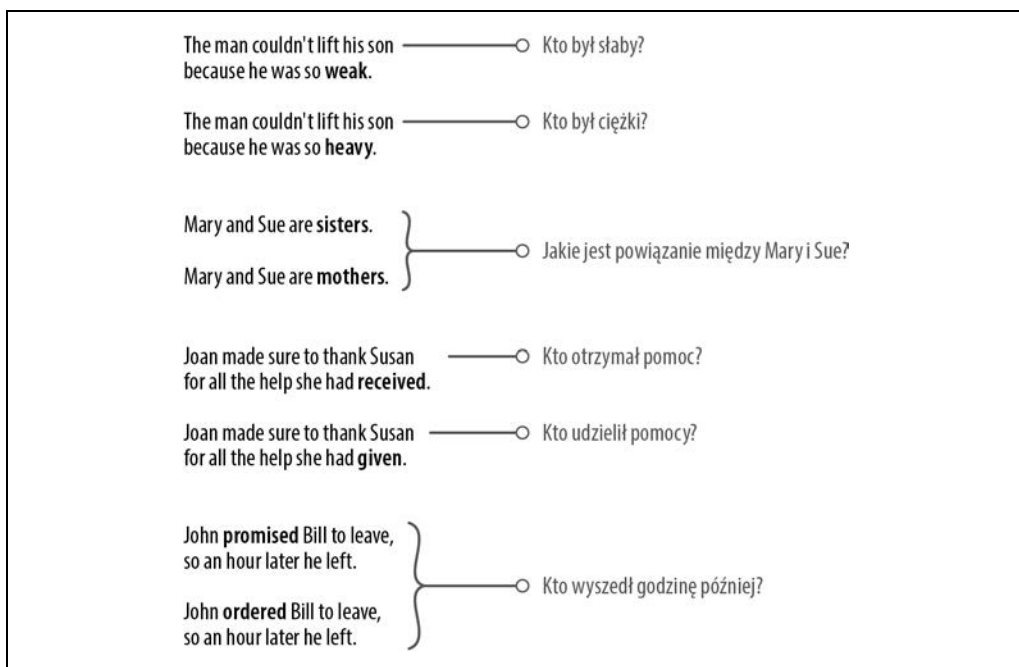
Dlaczego NLP jest trudnym wyzwaniem?

Co sprawia, że NLP jest trudnym wyzwaniem? Między innymi wieloznaczność i kreatywność ludzkiego języka. W tym punkcie zbadamy bliżej te cechy, zaczynając od wieloznaczności.

Wieloznaczność

Większość ludzkich języków jest z natury niejednoznaczna. Rozważ następujące angielskie zdanie: „I made her duck”. Ma ono wiele znaczeń. Pierwsze to: „Ugotowałem dla niej kaczkę”. Drugie to: „Zmusiłem ją do schylenia się w celu uniknięcia uderzenia przez lecący przedmiot”. (Istnieją też inne możliwe znaczenia; pozostawiamy je do wymyślenia czytelnikowi). Wieloznaczność wynika tu z użycia słowa „made”. To, które z tych dwóch znaczeń jest właściwe, zależy od kontekstu, w jakim wypowiedziano zdanie. Jeśli zdanie pojawia się w opowiadaniu o matce i dziecku, prawdopodobnie chodzi o pierwsze znaczenie. Ale jeśli pojawia się w książce o sporcie, prawdopodobnie chodzi o drugie znaczenie. Przykład, który widzieliśmy, to zdanie bezpośrednie.

Jeśli chodzi o język figuratywny — tzn. idiomy — wieloznaczność jest jeszcze większa. Na przykład: „He is as good as John Doe”. Jak dobry jest ten, o kim mowa? Odpowiedź zależy od tego, jak dobry jest John Doe. Na rysunku 1.7 przedstawiono kilka przykładów ilustrujących wieloznaczność języka.



Rysunek 1.7. Przykłady wieloznaczności języka zaczerpnięte z Winograd Schema Challenge

Przykłady pochodzą z zestawu testów Winograd Schema Challenge [5] nazwanego tak na cześć profesora Terry'ego Winograda z Uniwersytetu Stanforda. W zestawie tym znajdują się pary zdań, które różnią się tylko kilkoma słowami, ale zmiany te często powodują odwrócenie znaczenia. Przykłady te

są zrozumiałe dla ludzi, ale nie można ich rozwiązać z użyciem większości technik NLP. Przyjrzyj się parom zdań na rysunku i związanym z nimi pytaniami. To, jak zmienia się odpowiedź wskutek zmiany jednego słowa, powinno być oczywiste. W ramach innego eksperymentu możesz skorzystać z jakiegokolwiek gotowego systemu NLP, takiego jak Google Translate, aby sprawdzić, jak wieloznaczność wpływa (lub nie wpływa) na generowane wyniki.

Powszechna wiedza

Kluczowym aspektem każdego ludzkiego języka jest „powszechna wiedza”. To zbiór wszystkich faktów, które są znane większości ludzi. Podczas każdej konwersacji zakłada się, że fakty te są znane rozmówcom, więc nie wspomina się o nich jawnie, ale mają one wpływ na znaczenie zdań. Rozważ dwa zdania: „Człowiek ugryzł psa” i „Pies ugryzł człowieka”. Wiemy, że sytuacja z pierwszego zdania raczej się nie zdarzyła, a ta druga jest bardzo prawdopodobna. Dlaczego? Ponieważ wiadomo, że ludzie zwykle nie gryzą psów, wiadomo też, że psy często gryzą ludzi. Wiedza ta jest nam potrzebna, żeby orzec, że pierwsze zdanie zapewne jest fałszywe, a drugie prawdziwe. Zauważ jednak, że ta powszechna wiedza nie została wspomniana w żadnym zdaniu. Ludzie nieustannie wykorzystują powszechną wiedzę do rozumienia i przetwarzania języka. W powyższym przykładzie dwa zdania są bardzo podobne składniowo i komputer miałby problem z ich rozróżnieniem, gdyż brakuje mu powszechnej wiedzy, którą mają ludzie. Jednym z kluczowych wyzwań NLP jest zakodowanie powszechnej ludzkiej wiedzy w modelu komputerowym.

Kreatywność

Język to nie tylko reguły; ma on również aspekt kreatywny. W każdym języku używa się różnych stylów, dialektów, rejestrów i odmian. Doskonałym przykładem kreatywności w języku są wiersze. Sprawienie, żeby maszyny rozumiały kreatywność, to trudny problem nie tylko w NLP, ale w ogóle w dziedzinie sztucznej inteligencji.

Różnorodność języków

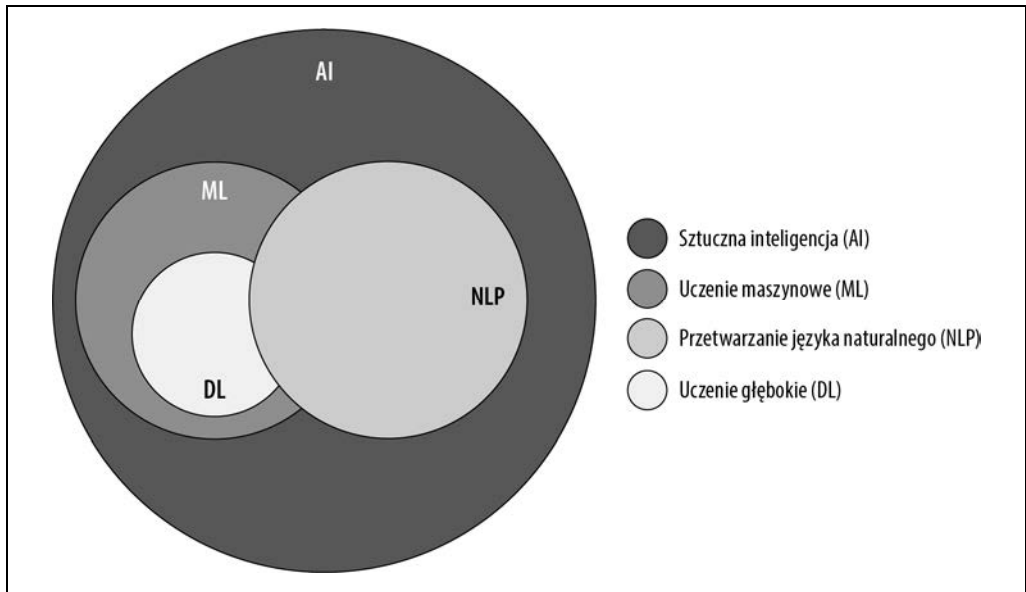
W przypadku większości języków świata nie ma bezpośredniego odwzorowania między ich słownikami. Utrudnia to przenoszenie rozwiązań NLP między językami. Rozwiązanie, które działa dobrze w jednym języku, może w ogóle nie działać w drugim. Oznacza to, że trzeba albo zbudować rozwiązanie neutralne językowo, albo budować oddzielnie rozwiązania dla każdego języka. To pierwsze jest trudne koncepcyjnie, to drugie pochłania mnóstwo pracy i czasu.

Wszystkie te problemy sprawiają, że praca z NLP jest bardzo trudna, choć satysfakcjonująca. Zanim pokażemy, jak NLP radzi sobie z tymi wyzwaniami, powinniśmy poznać typowe podejścia do rozwiązywania problemów NLP. Zaczniemy od omówienia związków uczenia maszynowego i uczenia głębokiego z NLP, a następnie przyjrzymy się różnym podejściom do NLP.

Uczenie maszynowe, uczenie głębokie i NLP — przegląd

Mówiąc nieformalnie, sztuczna inteligencja (ang. *artificial intelligence* — AI) to gałąź informatyki, która zajmuje się budowaniem systemów zdolnych do wykonywania zadań wymagających ludzkiej inteligencji. Fundamenty AI utworzono w latach 50. ubiegłego wieku podczas warsztatów

zorganizowanych w Dartmouth College [6]. Początkowo AI budowano głównie z systemów opartych na logice, heurystyce i regułach. Uczenie maszynowe (ang. *machine learning* — ML) to odnoga AI zajmująca się opracowywaniem algorytmów, które uczą się wykonywać zadania na podstawie dużej liczby przykładów, bez ręcznego kodowania reguł. Uczenie głębokie (ang. *deep learning* — DL) to odmiana uczenia maszynowego wykorzystująca sztuczne sieci neuronowe. ML, DL i NLP to poddziedziny AI, a ich wzajemne relacje zilustrowano na rysunku 1.8.



Rysunek 1.8. Relacje między NLP, ML i DL

Choć dziedziny NLP, ML i DL częściowo się nakładają, są to różne obszary badań, jak pokazuje rysunek. Podobnie jak inne wczesne prace AI, pierwsze aplikacje NLP również były oparte na regułach i heurystyce. Jednak w ciągu ostatnich dziesięcioleci do tworzenia aplikacji NLP zaczęto używać metod ML. W związku z tym warto powiedzieć tu kilka słów o ML i DL.

Celem systemów ML jest „uczenie się” wykonywania zadań na podstawie przykładów (zwanych danymi treningowymi) bez jasno sprecyzowanych instrukcji. Zwykle robi się to poprzez utworzenie liczbowej reprezentacji (zwanej cechami) danych treningowych i wykorzystanie tej reprezentacji do nauczenia systemu wzorców zawartych w danych. Algorytmy uczenia maszynowego można podzielić na trzy podstawowe paradygmaty: uczenie nadzorowane (ang. *supervised learning*), uczenie nienadzorowane (ang. *unsupervised learning*) i uczenie ze wzmocnieniem (ang. *reinforcement learning*). W uczeniu nadzorowanym celem jest nauczenie się funkcji odwzorowywania wejścia na wyjście na podstawie dużej liczby przykładów w postaci par wejście – wyjście. Pary wejście – wyjście są znane jako **dane treningowe**, a same wyjścia określa się mianem **etykiet** lub **prawdy podstawowej** (ang. *ground truth*). Przykładem problemu uczenia nadzorowanego jest nauka klasyfikowania wiadomości e-mail jako spamu lub nie spamu na podstawie tysięcy przykładów z obu kategorii. Jest to typowy scenariusz w NLP, a w tej książce znajdziesz wiele przykładów uczenia nadzorowanego, zwłaszcza w rozdziale 4.

Uczenie nienadzorowane to metody uczenia maszynowego, które mają na celu znajdowanie ukrytych wzorców w danych wejściowych bez żadnych wyjściowych danych odniesienia. To znaczy, że — w przeciwieństwie do uczenia nadzorowanego — uczenie nienadzorowane działa na dużych zbiorach nieoznakowanych danych. W NLP przykładem takiego zadania jest identyfikowanie tematów w dużych zbiorach danych tekstowych bez żadnej uprzedniej wiedzy o tych tematach. Jest to tzw. **modelowanie tematyczne**, które omówimy w rozdziale 7.

W rzeczywistych projektach NLP często używa się uczenia półnadzorowanego, w którym mamy niewielki zbiór etykietowanych danych i duży zbiór danych bez etykiet. Techniki półnadzorowane wiążą się z użyciem obu zbiorów danych w celu nauczania systemu danego zadania. Wreszcie uczenie ze wzmocnieniem polega na uczeniu systemu metodą prób i błędów i cechuje się brakiem dużej ilości danych czy to z etykietami, czy bez nich. Nauka odbywa się w izolowanym środowisku, a postępy osiąga się poprzez informacje zwrotne (nagrody lub kary). Ta odmiana uczenia nie jest (jeszcze) często spotykana w praktycznych systemach NLP. Częściej widuje się ją w takich zastosowaniach jak komputerowe szachy lub go, projektowanie pojazdów autonomicznych i robotyka.

Uczenie głębokie to uczenie maszynowe oparte na sztucznych sieciach neuronowych. Inspiracją do powstania sieci neuronowych były neurony w ludzkim mózgu i ich wzajemne interakcje. W minioniej dekadzie architektury neuronowe oparte na uczeniu głębokim wykorzystano do ulepszenia wielu inteligentnych aplikacji, takich jak rozpoznawanie obrazów i mowy oraz tłumaczenie maszynowe. Dlatego w branży pojawiło się wiele rozwiązań opartych na uczeniu głębokim, w tym aplikacje NLP.

W naszej książce wyjaśnimy, jak wykorzystuje się wszystkie te sposoby do tworzenia różnych aplikacji NLP. Opiszemy teraz różne podejścia do rozwiązywania problemów NLP.

Podejścia do NLP

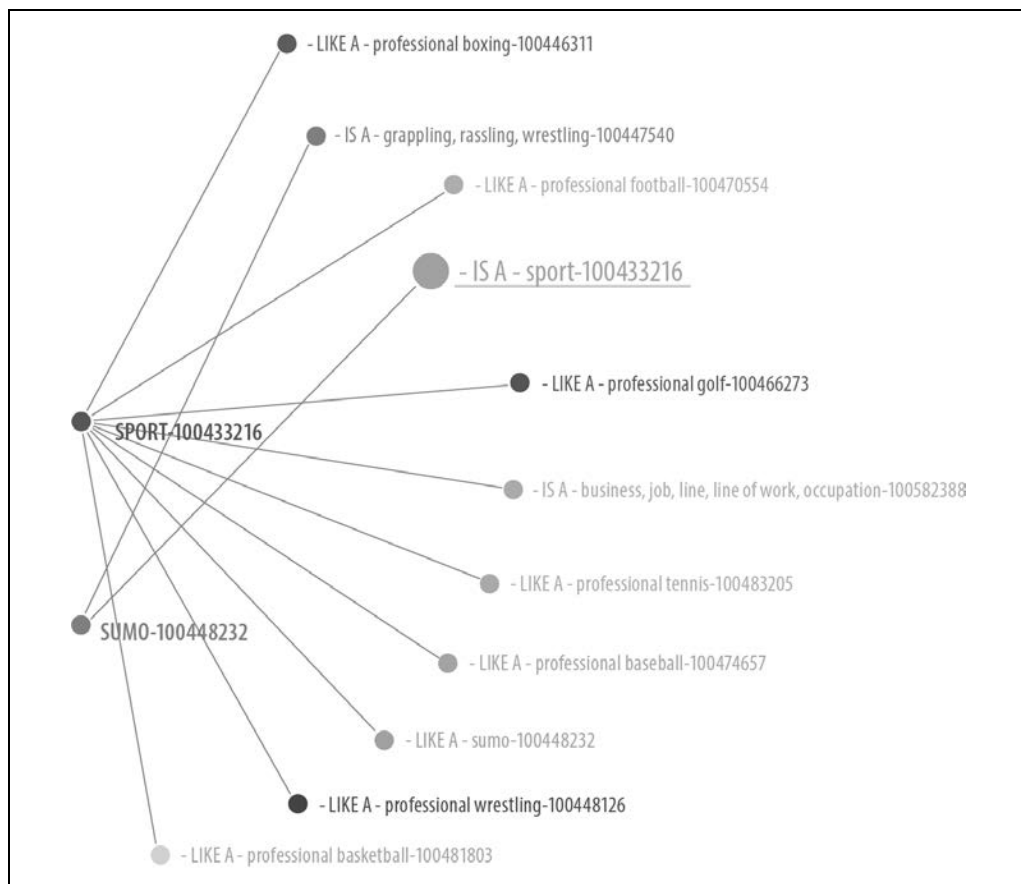
Sposoby rozwiązywania problemów NLP można podzielić na trzy kategorie: heurystyka, uczenie maszynowe i uczenie głębokie. Ten podrozdział jest krótkim wprowadzeniem do każdego podejścia — nie martw się, jeśli w pełni nie zrozumiesz wspomnianych tu koncepcji, ponieważ zostaną one opisane szczegółowo dalej w książce. Zacznijmy od omówienia NLP opartego na heurystyce.

NLP oparte na heurystyce

Podobnie jak inne wczesne systemy AI, pierwsze systemy NLP realizowały swoje zadania na podstawie reguł. Deweloperzy musieli zatem mieć specjalistyczną wiedzę w danej dziedzinie, aby sformułować reguły dołączane do programu. Systemy te wymagały również takich zasobów jak słowniki i tezaury, zwykle kompilowanych i przekształczanych w postacią cyfrową w dłuższych okresach. Przykładem projektowania reguł w celu rozwiązania problemu NLP z wykorzystaniem takich zasobów jest analiza odczuć oparta na leksykonie. Opiszemy ją krótko w rozdziale 4.

Oprócz słowników i tezaurusów tworzono również bardziej rozbudowane bazy wiedzy do wspomaganego NLP, zwłaszcza NLP opartego na regułach. Przykładem jest WordNet [7], baza słów i semantycznych relacji między nimi. Przykładami takich relacji są synonimy, hiponimy i meronimy. Synonimy to słowa o podobnym znaczeniu. Hiponimy reprezentują relację „jest typem czegoś”,

np. bejsbol, sumo i tenis to hiponimy sportu. Meronimy reprezentują relację „jest częścią czegoś”, np. ręce i nogi to meronimy ciała. Wszystkie te informacje przydają się podczas budowania systemów przetwarzania języka na podstawie reguł. Na rysunku 1.9 pokazano graficzną reprezentację takich relacji między słowami w bazie WordNet.



Rysunek 1.9. Graf bazy WordNet dla słowa „sport” [8]

Od niedawna do baz wiedzy takich jak Open Mind Common Sense [9] dodaje się także „zdroworozsądkową” wiedzę o świecie, co również wspomaga systemy oparte na regułach. Choć do tej pory mówiliśmy głównie o zasobach leksykalnych w postaci słów, systemy oparte na regułach mogą wykraczać poza słowa i wykorzystywać inne formy informacji. Niektóre z nich przedstawiono poniżej.

Wyrażenia regularne („regeksy”) to znakomite narzędzie do analizowania tekstu i budowania systemów opartych na regułach. Regex to zbiór znaków albo wzorzec używany do dopasowywania i wyszukiwania podłańcuchów w tekście. Na przykład regex „`^[a-zA-Z0-9_\-\.]+\">@([a-zA-Z0-9_\-\.]+\)\.\([a-zA-Z]{2,5}\)$`” pozwala znaleźć adresy e-mail w pewnym fragmencie tekstu. Regeksy są doskonałym sposobem dodawania wiedzy dziedzinowej do systemu NLP. Na przykład mamy skargi klientów zgłaszane za pośrednictwem czatu lub poczty e-mail i chcemy zbudować system

automatycznie identyfikujący produkt, którego dotyczy skarga. Istnieje zbiór kodów produktów, które można odwzorować na określone nazwy marek. Aby je dopasowywać, możemy użyć wyrażeń regularnych.

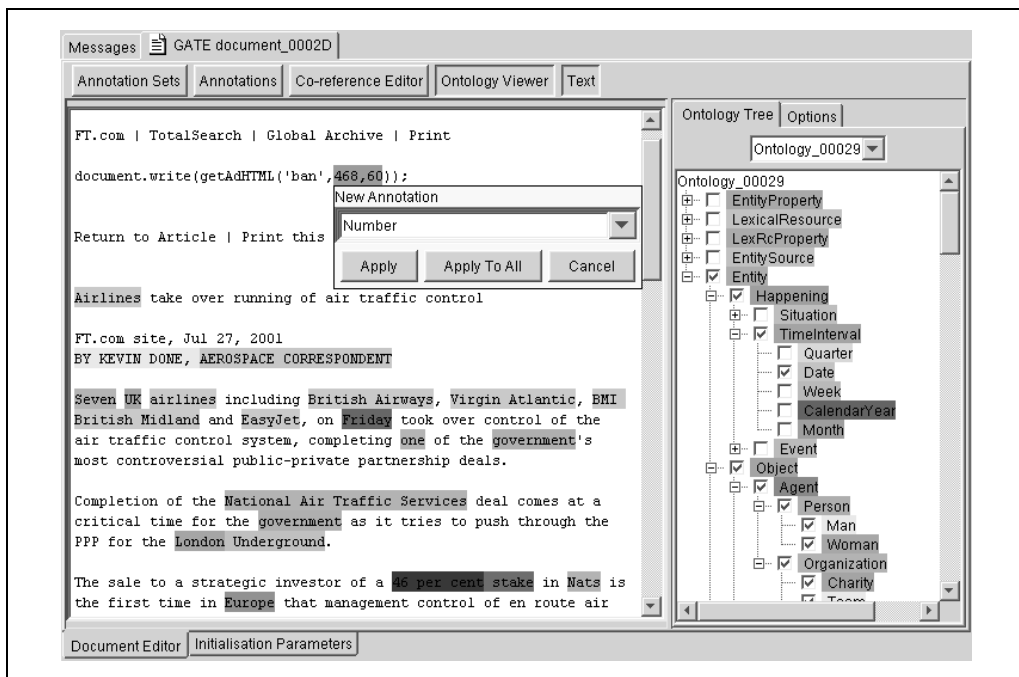
Regeksy to bardzo popularny paradygmat budowania systemów opartych na regułach. Na przykład oprogramowanie StanfordCoreNLP obejmuje TokensRegex [10], platformę do definiowania wyrażeń regularnych. Stosuje się ją do identyfikowania wzorców w tekście i tworzenia reguł na podstawie dopasowanego tekstu. Regeksów używa się do dopasowywania deterministycznego — tzn. coś albo pasuje, albo nie. Regeksy probabilistyczne to odmiana wyrażeń regularnych, która nie ma tego ograniczenia i pozwala określać prawdopodobieństwo dopasowania. Zainteresowani czytelnicy mogą przyrzeć się bibliotekom takim jak `pregex` [11].

Gramatyka bezkontekstowa (ang. *context-free grammar* — CFG) to typ formalnej gramatyki używanej do modelowania języków naturalnych. Jej twórcą jest profesor Noam Chomsky, słynny lingwista i naukowiec. CFG można wykorzystać do wyodrębniania bardziej skomplikowanych, hierarchicznych informacji, które mogłyby umknąć wyrażeniu regularnemu. Parser Earley [12] umożliwia analizę składniową tekstu według najróżniejszych gramatyk CFG. Aby modelować bardziej skomplikowane reguły, można posłużyć się językiem gramatycznym takim jak JAPE (Java Annotation Patterns Engine) [13]. JAPE łączy funkcje regeksów i gramatyk CFG i jest używany w opartych na regułach systemach NLP takich jak GATE (General Architecture for Text Engineering) [14]. GATE przydaje się do ekstrakcji tekstu w zamkniętych, dobrze zdefiniowanych dziedzinach, w których ważne są dokładność i kompletność. Na przykład JAPE i GATE wykorzystano do ekstrakcji informacji o procedurach wszczepiania implantów z raportów klinicznych [15]. Na rysunku 1.10 jako przykład systemu opartego na regułach pokazano interfejs GATE z kilkoma wyróżnionymi w tekście typami informacji.

Reguły i heurystyka wciąż odgrywają ważną rolę w cyklu życia projektów NLP. Są dobrym sposobem na budowanie początkowych wersji systemów NLP. Mówiąc najprościej: reguły i heurystyka pomagają szybko zbudować pierwszą wersję modelu i lepiej zrozumieć problem. Omówimy to szczegółowo w rozdziałach 4. i 11. Reguły i heurystyka bywają też przydatne jako dodatkowe funkcje systemów NLP opartych na uczeniu maszynowym. Używa się ich m.in. do domykania luk w systemie. Każdy system NLP zbudowany z wykorzystaniem technik statystycznych, uczenia maszynowego lub uczenia głębokiego będzie popełniał pomyłki. Niektóre pomyłki mogą być zbyt kosztowne — jak w przypadku systemu medycznego, który analizuje dokumentację pacjenta i błędnie decyduje, że nie trzeba wykonywać jakiegoś krytycznego testu. Tego rodzaju pomyłka mogłaby nawet kosztować czyjeś życie. Reguły i heurystyka to świetny sposób na wyeliminowanie takich luk z systemu produkcyjnego. Przejdźmy teraz do technik uczenia maszynowego stosowanych w NLP.

Uczenie maszynowe w NLP

Techniki uczenia maszynowego stosuje się do danych tekstowych tak samo jak do innych form danych, takich jak obrazy, mowa i dane ustrukturyzowane. Metody uczenia nadzorowanego, np. klasyfikację i regresję, intensywnie wykorzystuje się do różnych zadań NLP. Przykładowym zadaniem klasyfikacyjnym NLP jest klasyfikowanie artykułów prasowych według określonych tematów, takich



Rysunek 1.10. Narzędzie GATE

jak sport lub polityka. Natomiast metody regresyjne, które dają prognozę liczbową, można wykorzystać do szacowania ceny akcji na podstawie dyskusji o tych akcjach w mediach społecznościowych. Ponadto nienadzorowanych algorytmów grupowania można użyć do sortowania dokumentów tekstowych.

Każde podejście do NLP oparte na uczeniu maszynowym, nadzorowanym czy nie, zwykle składa się z trzech etapów: ekstrakcja cech z tekstu, użycie reprezentacji cech do uczenia modelu oraz ocenianie i ulepszanie modelu. Więcej o reprezentacji cech w odniesieniu do tekstu powiemy w rozdziale 3, a o ocenianiu — w rozdziale 2. Teraz krótko omówimy niektóre metody nadzorowanego uczenia maszynowego modeli NLP, których często używa się na drugim etapie (zastosowanie reprezentacji cech do uczenia modelu). Podstawowa wiedza o tych metodach pomoże Ci zrozumieć pojęcia omawiane w dalszych rozdziałach.

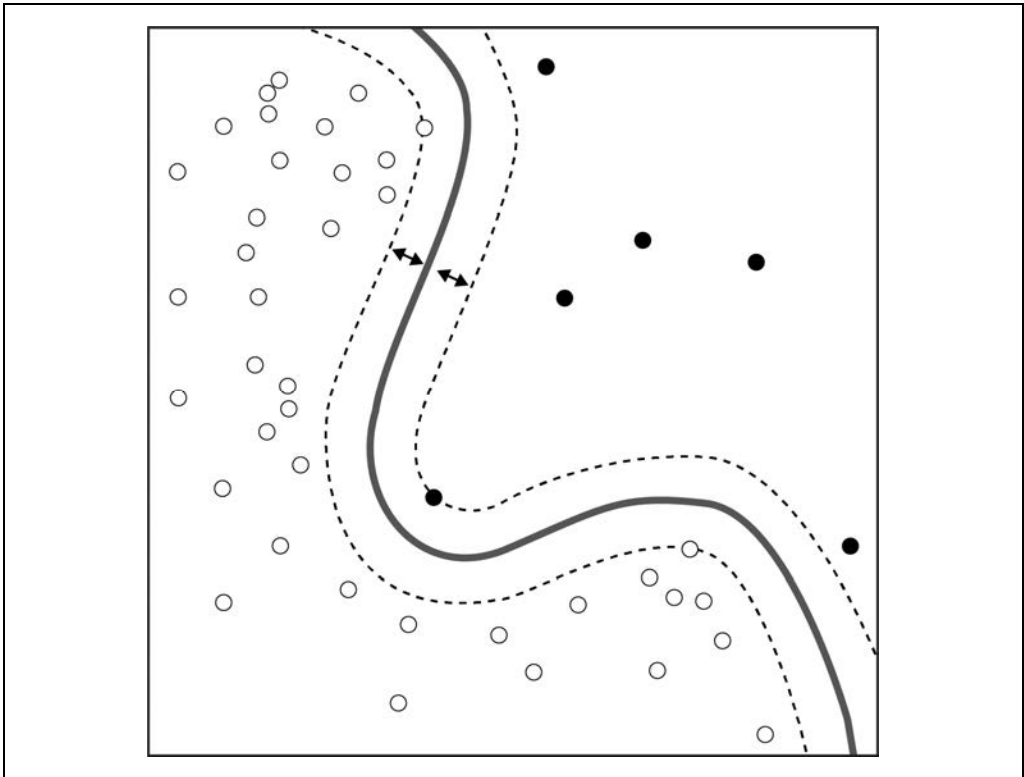
Naiwny klasyfikator bayesowski

Naiwny klasyfikator bayesowski to klasyczny algorytm do zadań klasyfikacyjnych [16], który opiera się głównie na twierdzeniu Bayesa (jak widać po nazwie). Wykorzystując twierdzenie Bayesa, oblicza prawdopodobieństwo zaobserwowania pewnej etykiety klasy na podstawie zbioru cech danych wejściowych. Charakterystyczne dla tego algorytmu jest to, że zakłada, iż każda cecha jest niezależna od innych. We wspomnianym wcześniej przykładzie klasyfikowania artykułów prasowych jednym ze sposobów liczbowego reprezentowania tekstu byłoby użycie liczb słów specyficznych dla poszczególnych dziedzin, takich jak sport lub polityka, które są obecne w tekście. Zakładamy, że liczby te nie są skorelowane. Jeśli to założenie jest uzasadnione, możemy użyć naiwnego

klasyfikatora bayesowskiego do klasyfikowania artykułów prasowych. Choć w wielu przypadkach założenie to jest wątpliwe, naiwny klasyfikator bayesowski powszechnie stosuje się jako początkowy algorytm do klasyfikowania tekstów, głównie dlatego, że jest zrozumiały oraz bardzo szybki w treningu i użyciu.

Maszyna wektorów nośnych

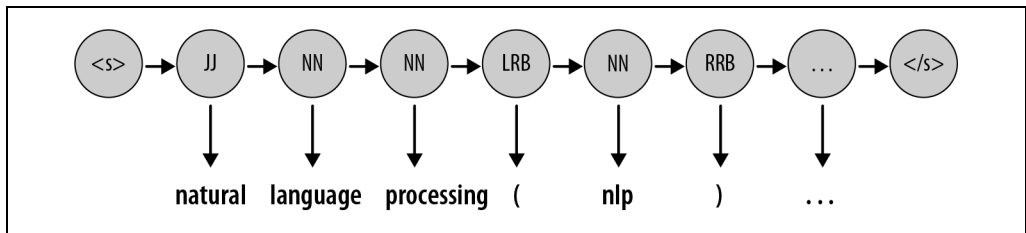
Innym popularnym algorytmem klasyfikacyjnym jest maszyna wektorów nośnych (ang. *support vector machine* — SVM) [17]. Celem każdej metody klasyfikacji jest wyznaczenie granicy decyzyjnej, która rozdziela różne kategorie tekstu (np. polityka kontra sport w naszym przykładzie klasyfikacji artykułów). Ta granica decyzyjna może być liniowa albo nieliniowa (np. okrąg). Maszyna SVM może nauczyć się zarówno liniowej, jak i nieliniowej granicy decyzyjnej w celu oddzielania punktów danych należących do różnych klas. W przypadku liniowej granicy decyzyjnej dane są reprezentowane w sposób, w którym różnice między klasami stają się oczywiste. Jeśli chodzi o dwuwymiarowe reprezentacje cech, przykład pokazano na rysunku 1.11, na którym czarne i białe punkty należą do różnych klas (np. do artykułów sportowych i politycznych). SVM uczy się optymalnej granicy decyzyjnej, maksymalizując odległość między punktami należącymi do różnych klas. Największą zaletą maszyn SVM jest odporność na zmienność i szумы w danych. Głównymi wadami są czas treningu i złe skalowanie w przypadku dużej ilości danych treningowych.



Rysunek 1.11. Dwuwymiarowa reprezentacja cech SVM

Ukryty model Markowa

Ukryty model Markowa (ang. *Hidden Markov Model* — HMM) to model statystyczny [18], który zakłada, że istnieje nieobserwowalny proces z ukrytymi stanami, który generuje dane — tzn. możemy zaobserwować dane dopiero po tym, jak zostaną wygenerowane. HMM próbuje następnie wymodelować ukryte stany na podstawie tych danych. Rozważmy zadanie NLP polegające na oznaczeniu części mowy (ang. *part of speech* — POS), które przypisuje tagi POS wyrazom w zdaniu. Zakładamy tu, że tekst jest generowany według pewnej bazowej gramatyki, która jest ukryta pod tekstem. Ukryte stany to części mowy, które definiują strukturę zdania zgodnie z gramatyką języka, ale my obserwujemy tylko słowa, które podlegają tym niewidocznym stanom. Jednocześnie modele HMM przyjmują założenie Markowa, co oznacza, że każdy ukryty stan jest zależny od poprzedniego stanu lub stanów. Ludzkie języki mają sekwencyjną naturę, a bieżące słowo w zdaniu zależy od tego, co było wcześniej. Dlatego modele HMM z tymi dwoma założeniami są bardzo skutecznym narzędziem do modelowania danych tekstowych. Na rysunku 1.12 pokazano przykład modelu HMM, który określa części mowy w danym zdaniu. Części mowy takie jak JJ (przymiotnik) i NN (rzeczownik) to stany ukryte, podczas gdy zdanie „natural language processing (nlp) ...” jest tym, co obserwujemy bezpośrednio.



Rysunek 1.12. Reprezentacja graficzna ukrytego modelu Markowa

Szczegółowe omówienie HMM w kontekście NLP znajduje się w rozdziale 8. książki *Speech and Language Processing* profesora Jurafsky’ego [19].

Warunkowe pola losowe

Warunkowe pole losowe (ang. *conditional random field* — CRF) to kolejny algorytm używany do przetwarzania danych sekwencyjnych. CRF zasadniczo przeprowadza zadanie klasyfikacyjne na każdym elemencie w sekwencji [20]. Wyobraź sobie ten sam przykład tagowania POS, w którym CRF potrafi oznaczać słowo po słowie poprzez wybieranie jednej części mowy z puli wszystkich dostępnych tagów POS. Ponieważ algorytm uwzględnia sekwencyjność danych wejściowych i kontekst tagów, jest bardziej ekspresywny niż zwykle metody klasyfikacji i zazwyczaj działa lepiej. CRF osiąga lepsze wyniki niż HMM w takich zadaniach jak tagowanie POS, które polegają na sekwencyjnej naturze języka. Algorytmy CRF, ich odmiany i zastosowania omówimy w rozdziałach 5., 6. i 9.

Przedstawiliśmy niektóre popularne algorytmy ML, które wykorzystuje się często do zadań NLP. Znajomość tych metod ML pomoże zrozumieć różne rozwiązania omawiane w tej książce. Warto też wiedzieć, kiedy warto używać poszczególnych algorytmów, o czym będzie mowa w kolejnych rozdziałach. Czytelnikom, którzy chcieliby dowiedzieć się więcej o etapach i teoretycznych szczegółach

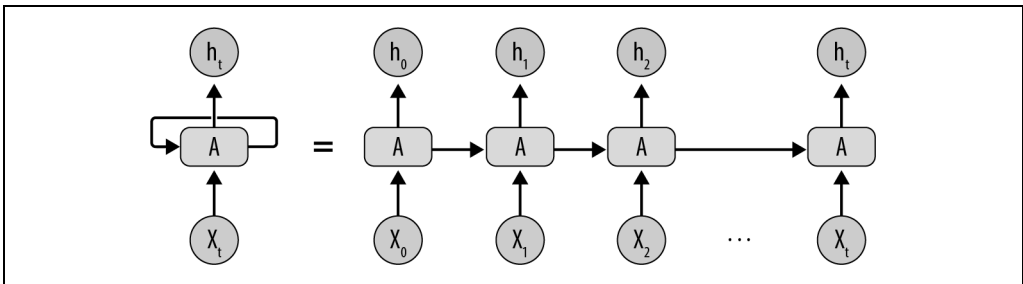
procesu uczenia maszynowego, polecamy podręcznik *Pattern Recognition and Machine Learning* Christophera Bishopa [21]. Bardziej praktyczne spojrzenie na uczenie maszynowe można znaleźć w doskonałej książce Aureliena Girona [22]. Przyjrzyjmy się teraz podejściom do NLP opartym na uczeniu głębokim.

Uczenie głębokie w NLP

Krótko wspomnieliśmy o popularnych metodach uczenia maszynowego, których intensywnie używa się w różnych zadaniach NLP. W ciągu kilku ostatnich lat zaczęto na dużą skalę wykorzystywać sieci neuronowe do przetwarzania złożonych, nieustrukturyzowanych danych. Język jest z natury skomplikowany i nieustrukturyzowany, dlatego do zrozumienia i rozwiązania problemów językowych potrzebujemy modeli o większych możliwościach reprezentacji i uczenia się. Oto kilka popularnych architektur głębokich sieci neuronowych, które obecnie są standardem w świecie NLP.

Rekurencyjne sieci neuronowe

Jak wspomniano wcześniej, język jest z natury sekwencyjny. Zdanie w dowolnym języku przepływa w jednym kierunku (np. angielski czyta się od lewej do prawej), zatem model, który potrafi progresywnie odczytywać tekst wejściowy od jednego do drugiego końca, może być bardzo przydatny do zrozumienia języka. Rekurencyjne sieci neuronowe (ang. *recurrent neural network* — RNN) są zaprojektowane specjalnie z myślą o takim sekwencyjnym przetwarzaniu i nauce. Sieci RNN mają jednostki neuronowe zdolne do zapamiętywania, co przetworzyły do tej pory. Pamięć ta jest temporalna, a informacje są zapisywane i aktualizowane w kolejnych etapach czasowych, w miarę jak RNN odczytuje następne słowo z wejścia. Na rysunku 1.13 przedstawiono rozwiniętą sieć RNN i sposób, w jaki śledzi ona dane wejściowe na różnych etapach czasowych.

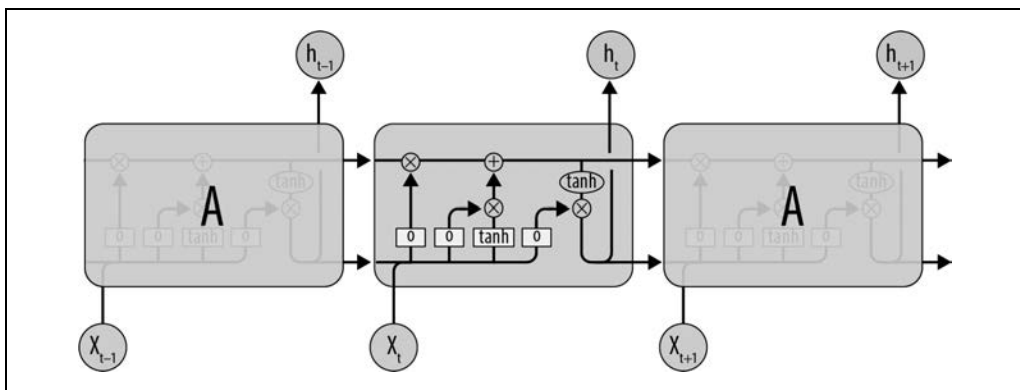


Rysunek 1.13. Rozwinięta rekurencyjna sieć neuronowa [23]

Sieci RNN mają duże możliwości i działają dobrze w różnorodnych zadaniach NLP, takich jak klasyfikacja tekstu, rozpoznawanie nazwanych encji, tłumaczenie maszynowe itd. Sieci RNN można również używać do generowania tekstu, kiedy celem jest odczytywanie poprzedniego tekstu i przewidywanie następnego słowa lub znaku. Szczegółowe omówienie wszechstronności sieci RNN oraz zakresu ich zastosowań w NLP i nie tylko można znaleźć w artykule *The Unreasonable Effectiveness of Recurrent Neural Networks* („Niedorzeczna efektywność rekurencyjnych sieci neuronowych”) [24].

Długa pamięć krótkoterminowa

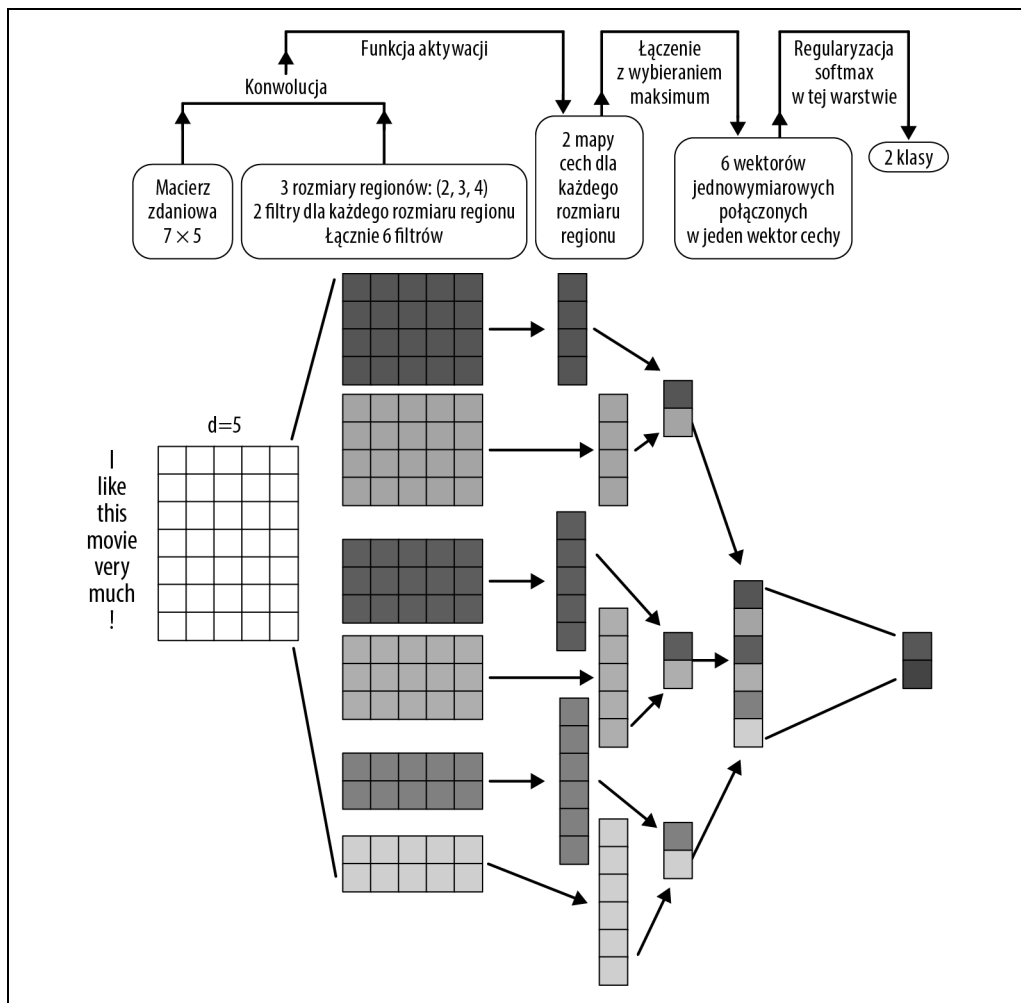
Pomimo swoich możliwości i wszechstronności sieci RNN są „zapominalskie” — nie potrafią zapamiętać dłuższych kontekstów i dlatego nie działają dobrze, gdy wprowadzany tekst jest długi, co zwykle ma miejsce w przypadku danych tekstowych. Sieci z długą pamięcią krótkotrwałą (ang. *long short-term memory* — LSTM), odmianą sieci RNN, wymyślono w celu złagodzenia tego problemu. Sieci LSTM omijają go, ignorując nieistotny kontekst i zapamiętując tylko tę część kontekstu, która jest potrzebna do rozwiązania danego zadania. Eliminuje to konieczność zapamiętywania bardzo długiego kontekstu w jednej reprezentacji wektorowej. Z tej przyczyny sieci LSTM zastąpiły sieci RNN w większości praktycznych zastosowań. Bramkowane jednostki rekurencyjne (ang. *gated recurrent unit* — GRU) to kolejna odmiana RNN, której używa się głównie do generowania języka. (Rodzinę modeli RNN szczegółowo opisuje artykuł Christophera Olaha [23]). Na rysunku 1.14 zilustrowano architekturę pojedynczej komórki LSTM. Konkretnie zastosowania LSTM w różnych aplikacjach NLP omówimy w rozdziałach 4., 5., 6. i 9.



Rysunek 1.14. Architektura komórki LSTM [23]

Konwolucyjne sieci neuronowe

Konwolucyjne sieci neuronowe (ang. *convolutional neural network* — CNN) są bardzo popularne i często używane w zadaniach wizji komputerowej, takich jak klasyfikacja obrazu, rozpoznawanie wideo itd. Odnoszą również sukcesy w NLP, zwłaszcza w zadaniach związanych z klasyfikacją tekstu. Każde słowo w zdaniu można zastąpić odpowiadającym mu wektorem słowa, a wszystkie wektory mają ten sam rozmiar (d) (zob. punkt „Osadzenia słów” w rozdziale 3.). Dzięki temu można układać je jeden na drugim, tworząc macierz lub tablicę 2D o wymiarze $n \times d$, gdzie n to liczba słów w zdaniu, a d to rozmiar wektorów słów. Macierz tę można traktować podobnie jak obraz i modelować za pomocą CNN. Główną zaletą sieci CNN jest możliwość jednoczesnego spojrzenia na grupę słów z wykorzystaniem okna kontekstowego. Na przykład próbujemy klasyfikować odczucia i otrzymujemy wejściowe zdanie typu: „Bardzo lubię ten film!”. Aby zrozumieć sens tego zdania, najlepiej przyjrzeć się słowom i różnym zbiorom sąsiadujących ze sobą słów. Sieci CNN potrafią to robić właśnie ze względu na swoją architekturę. Omówimy to dokładniej w późniejszych rozdziałach. Na rysunku 1.15 pokazano sieć CNN, która operuje na fragmencie tekstu w celu wyodrębnienia przydatnych fraz, aby ostatecznie wyznaczyć liczbę binarną wskazującą odczucia wyrażone w zdaniu w danym fragmencie tekstu.



Rysunek 1.15. Model CNN w działaniu [27]

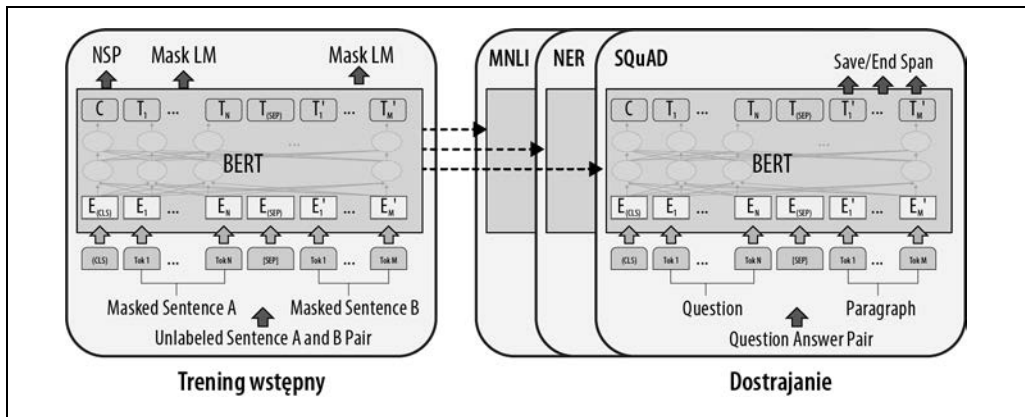
Jak pokazano na rysunku, CNN wykorzystuje zbiór warstw konwolucyjnych i łączących, aby uzyskać skondensowaną reprezentację tekstu, która następnie jest podawana na wejście w pełni połączonej warstwy w celu nauczenia sieci pewnych zadań NLP, takich jak klasyfikacja tekstu. Więcej informacji o używaniu sieci CNN do NLP można znaleźć w pozycjach [25] i [26]. Opiszemy je również w rozdziale 4.

Transformatory

Transformatory [28] to najnowszy konkurent w lidze modeli uczenia głębokiego na użytek NLP. W ciągu ostatnich dwóch lat zastosowano je do prawie wszystkich głównych zadań NLP. Modelują one kontekst tekstowy, ale nie w sposób sekwencyjny. Uwzględniając słowo na wejściu, przyglądają się wszystkim słowom wokół niego (co określa się mianem **samouwagi**) i reprezentują każde

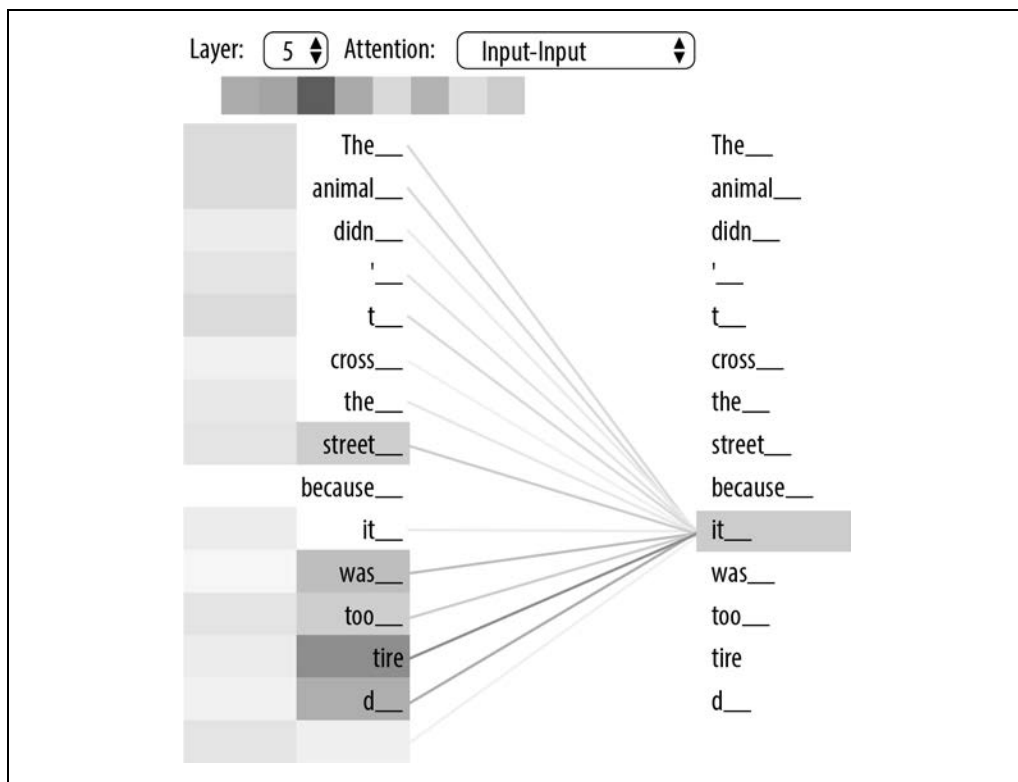
słowo w odniesieniu do jego kontekstu. Na przykład angielskie słowo „bank” może mieć różne znaczenia w zależności od kontekstu, w jakim się pojawia. Kiedy jest mowa o finansach, to „bank” prawdopodobnie oznacza instytucję finansową. Z drugiej strony, jeśli tekst dotyczy rzeki, słowo „bank” prawdopodobnie odnosi się do jej brzegu. Transformatory mogą modelować taki kontekst i dlatego intensywnie używa się ich w zadaniach NLP ze względu na większe zdolności reprezentacji w porównaniu z innymi sieciami głębkimi.

Niedawno duże transformatory zastosowano do **uczenia transferowego** z mniejszymi zadaniami podrzędnymi. Uczenie transferowe to technika AI, w której wiedzy uzyskanej podczas rozwiązywania jednego problemu używa się w innym, ale powiązonym problemie. W przypadku transformatorów pomysł polega na tym, żeby wytrenować bardzo duży tryb transformacyjny w sposób nie nadzorowany (co nazywa się **treningiem wstępnym**), aby przewidywały część zdania na podstawie reszty treści, a przez to mogły kodować wysokopoziomowe niuansy języka zawartego w tekście. Modele te szkoli się na ponad 40 GB danych tekstowych zgromadzonych z całego internetu. Przykładem dużego transformatora jest BERT (Bidirectional Encoder Representations from Transformers) [29], pokazany na rysunku 1.16, który został wstępnie wytrenowany na ogromnym zbiorze danych i udostępniony na zasadach open source przez Google’a.



Rysunek 1.16. Architektura BERT — wstępnie wytrenowany model i dostrajane modele specyficzne dla zadania

Wstępnie wytrenowany model pokazano po lewej stronie rysunku 1.16. Model ten następnie dostraja się do podrzędnych zadań NLP, takich jak klasyfikacja tekstu, ekstrakcja encji, odpowiadanie na pytania itd., jak pokazano po prawej stronie rysunku 1.16. Ze względu na ogromną ilość wstępnie nabytej wiedzy BERT efektywnie przekazuje wiedzę do zadań podrzędnych i w wielu z nich dorównuje najlepszym obecnie znanym technikom. W książce opiszemy wiele przykładów używania BERT-a do różnych zadań. Na rysunku 1.17 przedstawiono działanie mechanizmu samouwagi, który jest kluczowym komponentem transformatora. Zainteresowani czytelnicy mogą zajrzeć do artykułu [30], aby dowiedzieć się więcej o mechanizmach samouwagi i architekturze transformatorów. Model BERT i jego zastosowania omówimy w rozdziałach 4., 6. i 10.

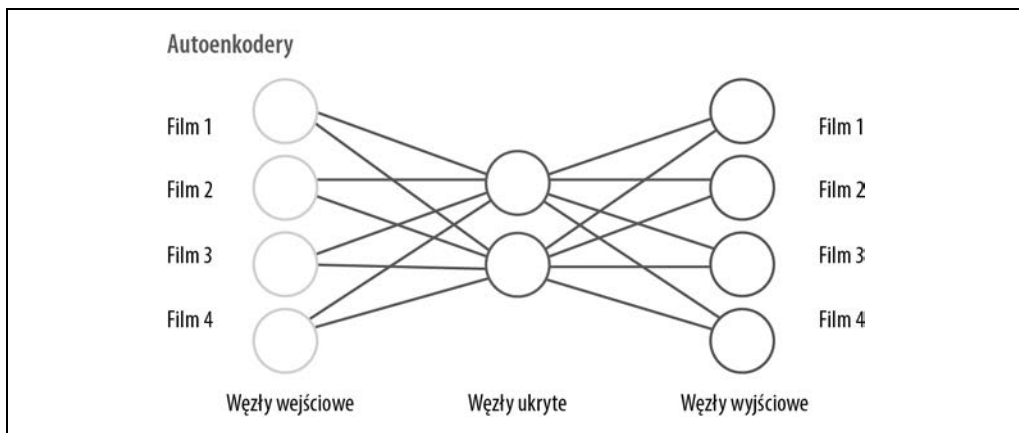


Rysunek 1.17. Mechanizm samouwagi w transformatorze [30]

Autoenkodery

Autoenkoder to sieć innego rodzaju, której używa się głównie do uczenia się skompresowanej reprezentacji wektorowej wejścia. Jeśli chcemy np. reprezentować tekst poprzez wektor, jaki jest na to dobry sposób? Możemy nauczyć się funkcji odwzorowującej tekst wejściowy na wektor. Aby ta funkcja odwzorowująca była użyteczna, „rekonstruujemy” dane wejściowe z reprezentacji wektorowej. Jest to forma uczenia nienadzorowanego, ponieważ nie potrzebujemy do niej etykiet dodawanych przez człowieka. Po treningu uzyskujemy reprezentację wektorową, która jest tekstem wejściowym zakodowanym w postaci gęstego wektora. Autoenkoderów zwykle używa się do tworzenia reprezentacji cech potrzebnej do dalszych zadań. Architekturę autoenkodera przedstawiono na rysunku 1.18.

W tym modelu warstwa ukryta zapewnia skompresowaną reprezentację danych wejściowych, która odzwierciedla ich istotę, a warstwa wyjściowa (dekoder) rekonstruuje reprezentację wejściową na podstawie reprezentacji skompresowanej. Choć architektura autoenkodera pokazana na rysunku 1.18 nie radzi sobie ze specyficznymi właściwościami danych sekwencyjnych takich jak tekst, to niektóre odmiany autoenkoderów, np. autoenkoder LSTM, działają całkiem dobrze. Więcej informacji o autoenkoderach można znaleźć w książce [31].



Rysunek 1.18. Architektura autoenkodera

W tym podrozdziale krótko przedstawiliśmy popularne architektury DL związane z NLP. Więcej ogólnych informacji o architekturach uczenia głębokiego znajdziesz w książce [31], a o stosowaniu ich w NLP — w artykule [25]. Mamy nadzieję, że to wprowadzenie pomoże Ci zrozumieć zastosowania DL omawiane dalej w naszej książce.

Zważywszy na niedawne osiągnięcia modeli DL, można by pomyśleć, że DL staje się preferowanym sposobem budowania systemów NLP. W praktyce wygląda to zupełnie inaczej. Wyjaśnijmy, z czego to wynika.

Dlaczego uczenie głębokie nie jest jeszcze „srebrną kulą” NLP?

W ciągu kilku ostatnich lat uczenie głębokie poczyniło wielkie postępy w NLP. Jeśli np. chodzi o klasyfikację tekstu, modele LSTM i CNN przewyższyły skuteczność standardowych technik uczenia maszynowego, takich jak naiwny klasyfikator bayesowski i SVM, w wielu zadaniach klasyfikacyjnych. Modele LSTM działają też lepiej w zadaniach etykietowania sekwencji, takich jak ekstrakcja encji, w porównaniu z modelami CRF. Niedawno faktycznym standardem w większości tych zadań NLP, od klasyfikacji do etykietowania sekwencji, stały się zaawansowane modele transformacyjne. Popularnym trendem jest używanie dużych (pod względem liczby parametrów) transformatorów, trenowanie ich na ogromnych zbiorach danych pod kątem ogólnych zadań NLP, takich jak modelowanie językowe, a następnie dostosowywanie ich do mniejszych zadań. To podejście (znane jako **uczenie transferowe**) przyniosło również sukcesy w innych dziedzinach, takich jak wizja i mowa komputerowa.

Pomimo tych sukcesów DL wciąż nie jest panaceum na wszystkie problemy NLP w rzeczywistych zastosowaniach. Kilka kluczowych powodów takiego stanu rzeczy opisano poniżej.

Nadmierne dopasowanie w przypadku małych zbiorów danych

Modele DL zwykle mają więcej parametrów niż tradycyjne modele ML, co oznacza, że cechują się większą ekspresywnością. Ma to swoją ciemną stronę. Brzytwa Ockhama [32] sugeruje, że jeśli wszystkie inne kryteria są jednakowe, zawsze powinno się wybierać rozwiązanie prostsze. Często zdarza się, że w fazie rozwojowej nie jest dostępna wystarczająca ilość danych

treningowych do wyszkolenia skomplikowanej sieci. W takich przypadkach lepiej wybrać prostszy model zamiast DL. Modele DL nadmiernie dopasowują się do małych zbiorów danych, co pogarsza ich zdolność generalizacji, a to z kolei prowadzi do niezadowalającego działania w środowisku produkcyjnym.

Uczenie na nielicznych przykładach i generowanie danych syntetycznych

W takich dziedzinach jak wizja komputerowa uczenie głębokie poczyniło znaczne postępy w uczeniu się na nielicznych przykładach (ang. *few-shot learning*) i w modelach, które potrafią generować obrazy wysokiej jakości [34]. W związku z tym stało się możliwe trenowanie modeli wizji opartych na DL z wykorzystaniem niewielkiej ilości danych, przez co DL znalazło znacznie więcej zastosowań w środowiskach przemysłowych. Nie opracowano jeszcze podobnych technik DL dla NLP.

Adaptacja dziedzinowa

Jeśli weźmiemy duży model DL wytrenowany na zbiorach danych należących do pewnych popularnych dziedzin (np. artykułów prasowych) i zastosujemy go do innej, nowszej domeny (np. postów w mediach społecznościowych), jego działanie może być niezadowalające. Ta ograniczona zdolność do generalizacji wskazuje, że modele DL nie zawsze są użyteczne. Na przykład modele wytrenowane na tekstach internetowych i recenzjach produktów nie będą działały dobrze w takich domenach jak prawo, media społecznościowe czy opieka zdrowotna, w których zarówno syntaktyczna, jak i semantyczna struktura języka jest specyficzna dla domeny. Potrzebujemy wyspecjalizowanych modeli, w których zakodowana jest wiedza dziedzinowa, a mogą to być po prostu specyficzne dla domeny modele oparte na regułach.

Modele interpretowalne

Pominąwszy adaptację dziedzinową, wadą modeli DL jest ograniczona kontrolowalność i interpretowalność, ponieważ w większości przypadków działają one jak czarna skrzynka. Firmy często potrzebują bardziej interpretowalnych wyników, które można wyjaśnić klientowi lub użytkownikowi końcowemu. W takich przypadkach użyteczniejsze mogą być tradycyjne techniki. Na przykład naiwny klasyfikator bayesowski do klasyfikowania odczuć może wyjaśniać wpływ, jaki mają pozytywne i negatywne słowa na przewidywanie wyrażanych odczuć.

Obecnie uzyskanie takich informacji z modelu klasyfikacyjnego opartego na LSTM jest trudne. Kontrastuje to z wizją komputerową, w której modele DL nie są czarnymi skrzynkami. W wizji komputerowej istnieje wiele technik [35], które pozwalają się dowiedzieć, dlaczego model dokonuje konkretnej prognozy. W NLP takie podejścia są rzadziej spotykane.

Zdrowy rozsądek i wiedza o świecie

Choć modele ML i DL uzyskują dobre wyniki we wzorcowych zadaniach NLP, język pozostaje enigmą dla naukowców. Oprócz składni i semantyki język obejmuje wiedzę o otaczającym nas świecie. Język używany do komunikacji polega na logicznym rozumowaniu i zdroworozsądkowym postrzeganiu zdarzeń zachodzących w świecie. Na przykład zdanie „Lubię pizzę” implikuje „Jestem zadowolony, kiedy jem pizzę”. Oto przykład bardziej złożonego rozumowania: „Jeśli Jan wyszedł z domu do ogrodu, to Jan nie jest już w domu i obecnie znajduje się w ogrodzie”. Dla człowieka to oczywiste, ale komputer potrzebuje wieloetapowego rozumowania, aby zidentyfikować zdarzenia i przewidzieć ich konsekwencje. Ponieważ ta wiedza

o świecie i zdrowy rozsądek są nieodłącznymi częściami języka, ich zrozumienie ma kluczowe znaczenie, jeśli model DL ma radzić sobie z różnymi zadaniami językowymi. Obecne modele osiągają dobre wyniki w standardowych testach, ale wciąż nie są zdolne do zdroworozsądkowego rozumienia i logicznego rozumowania. Prowadzi się prace nad zdroworozsądkową interpretacją zdarzeń i regułami logicznymi (np. rozumowaniem „jeśli — to”), ale nie są one jeszcze dobrze zintegrowane z modelami ML lub DL.

Koszt

Budowanie rozwiązań DL do zadań NLP bywa kosztowne. Wynika to z wielu przyczyn. Modele DL są znane jako pożeracze danych. Gromadzenie i etykietowanie dużego zbioru danych może pochłonąć znaczne środki. Ze względu na rozmiar modeli DL trenowanie ich w celu osiągnięcia pożądanej wydajności może nie tylko wydłużyć cykle rozwojowe, ale także prowadzić do wysokich rachunków za specjalistyczny sprzęt (GPU). Ponadto wdrażanie i utrzymywanie modeli DL bywa drogie zarówno z perspektywy wymagań sprzętowych, jak i nakładów pracy. Wreszcie ponieważ modele te są duże, mogą powodować problemy z opóźnieniami w czasie wnioskowania i być nieprzydatne w sytuacjach, gdy wymagane są małe opóźnienia. Do tej listy można jeszcze dodać dług techniczny wynikający z budowy i utrzymania „ciężkiego” modelu. Mówiąc nieformalnie: dług techniczny to koszt przyszłych przeróbek, który wynika z przedkładania szybkiego ukończenia prac nad dobre wybory projektowe i implementacyjne.

Wdrożenie w urządzeniach

W wielu scenariuszach rozwiązanie NLP musi działać bezpośrednio w urządzeniu końcowym, a nie w chmurze — np. turystyczny system tłumaczenia maszynowego, który wypowiada przetłumaczony tekst nawet bez dostępu do internetu. W takich przypadkach, ze względu na specyfikę urządzenia, rozwiązanie musi działać z ograniczoną pamięcią i zużyciem energii. Większość rozwiązań DL nie radzi sobie z takimi ograniczeniami. Trwają prace nad wdrażaniem DL w urządzeniach końcowych [36, 37, 38], ale nadal jesteśmy daleko od ogólnych rozwiązań.

W większości projektów pojawia się jeden lub wiele wymienionych wyżej czynników. Prowadzi to do dłuższych cykli projektowych i wyższych kosztów (sprzętu, pracy), a skuteczność rozwiązań jest albo porównywalna z modelami ML, albo nawet niższa. Oznacza to niski zwrot z inwestycji i często sprawia, że projekt NLP kończy się niepowodzeniem.

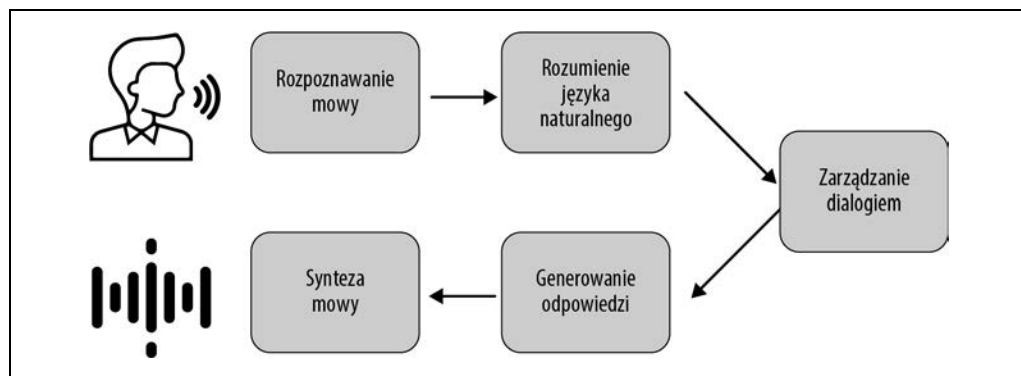
Z powyższej dyskusji wynika, że DL nie zawsze jest preferowanym rozwiązaniem we wszystkich praktycznych aplikacjach NLP. Zaczniemy więc tę książkę od omówienia fundamentalnych aspektów zadań NLP i pokazania, jak je realizować z wykorzystaniem różnych technik, od systemów opartych na regułach do modeli DL. Położymy nacisk na wymagania w zakresie danych i proces budowania modelu, a nie tylko na techniczne szczegóły poszczególnych modeli. Ze względu na szybkie postępy w tej dziedzinie przewidujemy, że w przyszłości pojawią się nowsze, bardziej zaawansowane modele DL, ale podstawowe aspekty zadań NLP nie zmienią się w zasadniczy sposób. Właśnie dlatego najpierw omówimy podstawy NLP i będziemy stopniowo je rozwijać w celu budowania modeli o rosnącej złożoności, zamiast od razu przeskakiwać do najbardziej zaawansowanych technicznie rozwiązań.

Podobnie jak profesor Zachary Lipton z Uniwersytetu Carnegiego i Mellonów i profesor Jacob Steinhardt z Uniwersytetu Kalifornijskiego w Berkeley [39] my również chcemy przestrzec przed czytaniem artykułów naukowych, prac badawczych i blogów na temat ML i NLP bez kontekstu i odpowiedniego przeszkolenia. Beżładna lektura dużej liczby innowacyjnych prac może powodować dezorientację i niezbyt precyzyjne zrozumienie tematu. Wiele najnowszych modeli DL nie jest wystarczająco interpretowalnych, aby wskazać źródła korzyści empirycznych. Lipton i Steinhardt piszą również o mieszanii terminów technicznych i niewłaściwym użyciu języka w artykułach naukowych poświęconych uczeniu maszynowemu, które często nie wskazują jasno drogi do rozwiązania danego problemu. Dlatego w tej książce dokładnie omawiamy różne koncepcje techniczne dotyczące stosowania ML w zadaniach NLP za pośrednictwem przykładów, kodu i wskazówek w rozdziałach.

Dotychczas omówiliśmy niektóre podstawowe pojęcia związane z językiem, NLP, ML i DL. Zanim podsumujemy rozdział 1., przyjrzyjmy się studium przypadku, by lepiej zrozumieć różne komponenty aplikacji NLP.

Przewodnik po NLP — agenty konwersacyjne

Głosowe agenty konwersacyjne, takie jak Amazon Alexa i Apple Siri, to jedno z najpowszechniejszych zastosowań NLP; są one znane większości ludzi. Typowy model interakcji agenta konwersacyjnego przedstawiono na rysunku 1.19.



Rysunek 1.19. Przepływ działania agentów konwersacyjnych

Oto najważniejsze komponenty NLP używane w tym przepływie:

1. **Rozpoznawanie i synteza mowy.** Są to podstawowe komponenty każdego głosowego agenta konwersacyjnego. Rozpoznawanie mowy obejmuje przekształcanie sygnałów dźwiękowych na fonemy, które są następnie transkrybowane jako słowa. Synteza mowy to proces odwrotny, polegający na przekształceniu wyników tekstowych w język mówiony. Obie te techniki bardzo się rozwinęły w ostatniej dekadzie, a w większości standardowych przypadków zalecamy realizowanie ich z wykorzystaniem chmurowych interfejsów API.

2. **Rozumienie języka naturalnego.** Jest to kolejny komponent w łańcuchu działań agenta konwersacyjnego, w którym odpowiedź użytkownika (przekształconą w tekst) analizuje się z wykorzystaniem systemu rozumienia języka naturalnego. Można to podzielić na wiele mniejszych podzadań NLP, takich jak:
- **Analiza odczuć.** Analizujemy tu odczucia wyrażone w odpowiedzi użytkownika. Zostanie to opisane w rozdziale 4.
 - **Rozpoznawanie nazwanych encji.** Tutaj identyfikujemy wszystkie ważne encje wspomniane przez użytkownika w odpowiedzi. Zostanie to opisane w rozdziale 5.
 - **Rozwikływanie odniesień.** Tutaj znajdujemy odniesienia do wyodrębnionych encji w historii konwersacji. Użytkownik może np. powiedzieć: „Avengersi byli fantastyczni”, a później odnieść się z powrotem do filmu, mówiąc: „Efekty specjalne w tym filmie były niesamowite”. W takim przypadku chcemy połączyć słowo „film” z „Avengersami”. Zostanie to krótko opisane w rozdziale 5.
3. **Zarządzanie dialogiem.** Po wyodrębnieniu użytecznych informacji z odpowiedzi użytkownika możemy chcieć zrozumieć jego intencje, np. czy pyta o fakty („Jaka jest dziś pogoda?”), czy wydaje polecenie („Zagraj utwór Mozarta”). Możemy użyć systemu klasyfikacji tekstu, aby klasyfikować odpowiedź użytkownika jako jeden ze wstępnie zdefiniowanych zamiarów. Dzięki temu agent konwersacyjny wie, o co jest proszony. Klasyfikację zamiarów omówimy w rozdziałach 4. i 6. W procesie tym system może zadać kilka dodatkowych pytań wyjaśniających, aby uzyskać dalsze informacje od użytkownika. Kiedy już domyśliliśmy się intencji użytkownika, chcemy ustalić, jakie działania powinien podjąć agent konwersacyjny, aby spełnić żądanie. Robi się to na podstawie informacji i intencji wyodrębnionych z odpowiedzi użytkownika. Właściwym działaniem może być np.: wygenerowanie odpowiedzi na podstawie informacji dostępnych w internecie, odtworzenie muzyki, przyciemnienie światła lub zadanie pytania wyjaśniającego. Opiszemy to w rozdziale 6.
4. **Generowanie odpowiedzi.** Wreszcie agent konwersacyjny generuje odpowiednie działanie do wykonania na podstawie semantycznej interpretacji intencji użytkownika i dodatkowych informacji uzyskanych w dialogu z użytkownikiem. Jak już wspomniano, agent może pobrać informacje z bazy wiedzy i wygenerować odpowiedź według wstępnie zdefiniowanego szablonu. Może np. powiedzieć: „Odtwarzam XXV symfonię” albo „Światła zostały przyciemnione”. W niektórych scenariuszach może nawet wygenerować zupełnie nową odpowiedź.

Mamy nadzieję, że to krótkie studium przypadku pokazało, jak różne komponenty NLP, które będziemy omawiać w tej książce, łączą się w jedną aplikację: agenta konwersacyjnego. W miarę lektury dowiesz się więcej o tych komponentach, a same agenty konwersacyjne zostaną dokładniej opisane w rozdziale 6.

Podsumowanie

W tym rozdziale zbadaliśmy wiele zagadnień związanych z NLP, od szerszych rozważań o tym, czym jest język, do konkretnego studium praktycznej aplikacji NLP. Opisaliśmy też, jak rzeczywiście używa się NLP, jakie są jego główne problemy i zadania i jaką rolę odgrywają w nim uczenie maszynowe i uczenie głębokie. Celem tego rozdziału było przekazanie Ci podstawowej wiedzy,

którą będziesz stopniowo uzupełniał w miarę dalszej lektury. Następne dwa rozdziały (rozdział 2. i 3.) są poświęcone fundamentalnym etapom budowania aplikacji NLP. W rozdziałach od 4. do 7. skupimy się na kluczowych zadaniach NLP i praktycznych problemach, które można rozwiązać za ich pomocą. W rozdziałach od 8. do 10. pokażemy, jak używa się NLP w różnych branżach, takich jak e-commerce, opieka zdrowotna, finanse itd. Rozdział 11. łączy wszystko w całość i wyjaśnia, jak buduje się kompleksową aplikację NLP, w kategoriach projektowania, rozwijania, testowania i wdrażania. Skoro mamy już za sobą to ogólne omówienie, zanurzymy się głębiej w świat NLP.

Bibliografia

- [1] Arria.com, *The Power of Language to your industry* (<https://www.arria.com/industry-expertise>), ostatni dostęp: 7 lutego 2023 r.
- [2] UCL, Phonetic symbols for English (<https://oreil.ly/5jnsI>), ostatni dostęp: 15 czerwca 2020 r.
- [3] Bender Emily M., *Linguistic Fundamentals for Natural Language Processing: 100 Essentials From Morphology and Syntax*, Synthesis Lectures on Human Language Technologies, 6.3 (2013): s. 1 – 184.
- [4] Bender Emily M. i Alex Lascarides, *Linguistic Fundamentals for Natural Language Processing II: 100 Essentials from Semantics and Pragmatics*, Synthesis Lectures on Human Language Technologies, 12.3 (2019): s. 1 – 268.
- [5] Levesque Hector, Ernest Davis i Leora Morgenstern, *The Winograd Schema Challenge*, The Thirteenth International Conference on the Principles of Knowledge Representation and Reasoning (2012).
- [6] Wikipedia, *Dartmouth workshop* (<https://oreil.ly/6NZGh>), ostatnia modyfikacja: 30 marca 2020 r.
- [7] Miller George A., *WordNet: A Lexical Database for English*, „Communications of the ACM”, 38.11 (1995): s. 39 – 41.
- [8] Visual Thesaurus of English Collocations, *Visual Wordnet with D3.js* (<https://oreil.ly/EY1HB>), ostatni dostęp: 15 czerwca 2020 r.
- [9] Singh Push, Thomas Lin, Erik T. Mueller, Grace Lim, Travell Perkins i Wan Li Zhu, *Open Mind Common Sense: Knowledge Acquisition from the General Public*, Meersman R. i Tari Z. (red.), *On the Move to Meaningful Internet Systems 2002: CoopIS, DOA, and ODBASE*, OTM 2002, Lecture Notes in Computer Science, vol. 2519, Berlin, Heidelberg: Springer.
- [10] The Stanford Natural Language Processing Group, Stanford TokensRegex (<https://oreil.ly/M3KnK>), (oprogramowanie), ostatni dostęp: 15 czerwca 2020 r.
- [11] Hewitt Luke, Probabilistic regular expressions (<https://oreil.ly/BqhIX>), (repozytorium na GitHubie).
- [12] Earley Jay, *An Efficient Context-Free Parsing Algorithm*, „Communications of the ACM”, 13.2 (1970): s. 94 – 102.

- [13] *Java Annotation Patterns Engine: Regular Expressions over Annotations* (<https://oreil.ly/dmdOs>), *Developing Language Processing Components with GATE Version 9 (a User Guide)*, rozdział 8., ostatni dostęp: 15 czerwca 2020 r.
- [14] General Architecture for Text Engineering (GATE) (<https://gate.ac.uk>), ostatni dostęp: 15 czerwca 2020 r.
- [15] Rosier Arnaud, Anita Burgun i Philippe Mabo, *Using Regular Expressions to Extract Information on Pacemaker Implantation Procedures from Clinical Reports*, „AMIA Annual Symposium Proceedings”, v. 2008 (2008): s. 81 – 85.
- [16] Zhang Haiyi i Di Li, *Naive Bayes Text Classifier*, 2007 IEEE International Conference on Granular Computing (GRC 2007): s. 708.
- [17] Joachims Thorsten, *Learning to Classify Text Using Support Vector Machines*, vol. 668, New York: Springer Science & Business Media, 2002, ISBN: 978-1-4615-0907-3.
- [18] Baum Leonard E. i Ted Petrie, *Statistical Inference for Probabilistic Functions of Finite State Markov Chains*, „The Annals of Mathematical Statistics”, 37.6 (1966): s. 1554 – 1563.
- [19] Jurafsky Dan i James H. Martin, *Speech and Language Processing, Third Edition* (wersja robocza) (<https://oreil.ly/sZfWl>), 2018.
- [20] Settles Burr, *Biomedical Named Entity Recognition Using Conditional Random Fields and Rich Feature Sets*, Proceedings of the International Joint Workshop on Natural Language Processing in Biomedicine and its Applications (NLPBA/BioNLP), (2004): s. 107 – 110.
- [21] Bishop Christopher M., *Pattern Recognition and Machine Learning*, New York: Springer, 2006, ISBN: 978-0-3873-1073-2.
- [22] Geron Aurelien, *Uczenie maszynowe z użyciem Scikit-Learn, Keras i TensorFlow. Wydanie II*, Helion, 2020, ISBN: 978-83-283-6002-0.
- [23] Olah Christopher, *Understanding LSTM Networks* (<https://oreil.ly/X6dwG>), 27 sierpnia 2015 r.
- [24] Karpathy Andrej, *The Unreasonable Effectiveness of Recurrent Neural Networks* (<https://oreil.ly/qTAXV>), 21 maja 2015 r.
- [25] Goldberg Yoav, *Neural Network Methods for Natural Language Processing*, Synthesis Lectures on Human Language Technologies, 10.1 (2017): s. 1 – 309.
- [26] Britz Denny, *Understanding Convolutional Neural Networks for NLP* (<https://oreil.ly/vJppc>), 7 listopada 2015 r.
- [27] Le Hoa T., Christophe Cerisara i Alexandre Denis, *Do Convolutional Networks need to be Deep for Text Classification?*, Workshops at the Thirty-Second AAAI Conference on Artificial Intelligence, 2018.
- [28] Vaswani Ashish, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser i Illia Polosukhin, *Attention Is All You Need*, *Advances in Neural Information Processing Systems*, 2017: s. 5998 – 6008.

- [29] Devlin Jacob, Ming-Wei Chang, Kenton Lee i Kristina Toutanova, *BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding* (<https://oreil.ly/xdtmX>), 11 października 2018 r.
- [30] Alammur Jay, *The Illustrated Transformer* (<https://oreil.ly/F19n3>), 27 czerwca 2018 r.
- [31] Goodfellow Ian, Yoshua Bengio i Aaron Courville, *Deep Learning*, Cambridge: MIT Press, 2016, ISBN: 978-0-262-03561-3.
- [32] Varma Nakul, *COMS 4771: Introduction to Machine Learning* (<https://oreil.ly/yRJZP>), wykład 6, slajd 7, ostatni dostęp: 15 czerwca 2020 r.
- [33] Wang Yaqing, Quanming Yao, James Kwok i Lionel M. Ni, *Generalizing from a Few Examples: A Survey on Few-Shot Learning* (<https://oreil.ly/LyMxm>), (2019).
- [34] Wang Zhengwei, Qi She i Tomas E. Ward, *Generative Adversarial Networks in Computer Vision: A Survey and Taxonomy* (<https://oreil.ly/OFvz7>), (2019).
- [35] Olah Chris, Arvind Satyanarayan, Ian Johnson, Shan Carter, Ludwig Schubert, Katherine Ye i Alexander Mordvintsev, *The Building Blocks of Interpretability*, Distill 3.3 (marzec 2018): e10.
- [36] Nan Kaiming, Sicong Liu, Junzhao Du i Hui Liu, *Deep Model Compression for Mobile Platforms: A Survey*, „Tsinghua Science and Technology”, 24.6 (2019): s. 677 – 693.
- [37] TensorFlow, *Get started with TensorFlow Lite* (<https://oreil.ly/Jxsuc>), ostatnia modyfikacja: 21 marca 2020 r.
- [38] Ganesh Prakhar, Yao Chen, Xin Lou, Mohammad Ali Khan, Yin Yang, Deming Chen, Marianne Winslett, Hassan Sajjad i Preslav Nakov, *Compressing Large-Scale Transformer-Based Models: A Case Study on BERT* (<https://oreil.ly/VSQvc>), (2020).
- [39] Lipton Zachary C. i Jacob Steinhardt, *Troubling Trends in Machine Learning Scholarship* (<https://oreil.ly/lpay1>), (2018).

A

agenty konwersacyjne, 54
AI, artificial intelligence, 38
aktualizowanie modelu, 92
algorytmy
 ekstrakcji atrybutów, 308
 ML, 80, 102
 modelowania tematycznego, 250
analiza
 aspektów, 322
 informacji medycznych, 344
 komponentów systemu dialogowego, 220
 modelu NLP, 367
 recenzji, 303, 317, 323
 analiza odczuć, 318
 aspektowa analiza odczuć, 319
 łączenie ocen z aspektami, 321
 rozumienie aspektów, 322
 regresyjna aspektów latentnych, LARA, 321
aplikacje
 NLP, 274
 NLP w e-commerce, 303
architektura
 autoenkodera, 51
 BERT, 49
 DBoW, 121
 Doc2vec, 121
 SkipGram, 117
aspektowa analiza odczuć, 319
 podejście nadzorowane, 320
 podejście nienadzorowane, 321
ASR, automatic speech recognition, 69
asystenty zdrowotne, 333
atrybuty
 danych, 87
 języków, 93
 projektu, 92

autoenkodery, 50
automatyczne rozpoznawanie mowy, ASR, 69
automatyzacja uczenia maszynowego, 370
AutoML, 371, 373
auto-sklearn, 371

B

biblioteka
 spaCy, 183
 TextBlob, 290
BioBERT, 345
BoN, bag-of-n-grams, 106
bot
 autotldr, 256
 FAQ, 205, 206
 o strukturze otwartej, 207
 oparty na przepływie, 206
BoW, Bag of Words, 104
budowanie i utrzymywanie systemu, 363
 automatyzacja uczenia maszynowego, 370
 interpretowalność, 366
 minimalizowanie długu technicznego, 369
 monitorowanie, 368
 odtwarzalność kodu i modelu, 365
 rozwiązywanie problemów, 366
 rozwijanie modeli, 364
 znajdowanie lepszych cech, 363
budowanie systemu NER, 179

C

CBoW, continuous bag of words, 114
chmura wyrazowa, 250, 281
CNN, convolutional neural network, 47, 153
CRF, conditional random field, 45
CRM, customer relationship management, 293
cykl życia projektu AI, 383

- czatbot, 203
 - FAQ, 206
 - o strukturze otwartej, 207
 - oparty na przepływie, 206
 - PizzaStop, 211
 - Rasa, 231
 - czatboty
 - generatywne, 236
 - potok budowania systemu, 208
 - studium przypadku, 233
 - szczegóły systemu dialogowego, 210
 - tworzone w Dialogflow, 211, 234
 - zastosowania, 204
 - często zadawane pytania, FAQ, 204
- ## D
- dane
 - SMTD, 275
 - adresy URL, 289
 - apostrofy, 288
 - brak gramatyki, 277
 - długość tekstu, 279
 - emotikony, 288
 - niestandardowa pisownia, 277, 290
 - nietekstowe, 287
 - stosowanie NLP, 281
 - szum, 279
 - transliteracja, 278
 - wielojęzyczność, 278
 - złączone słowa, 289
 - znaki specjalne, 278
 - znakowanie, 287
 - treningowe, 39
 - brak, 160
 - niewielka ilość, 161
 - treningowe NER, 181
 - zdrowotne i medyczne, 331
 - data science, 374
 - DBoW, distributed bag of words, 121
 - denotacja, 109
 - Dialogflow, 211, 234
 - interfejs użytkownika, 212, 213
 - testowanie agenta, 218
 - tworzenie
 - agenta, 212
 - czatbotów, 211, 234
 - encji, 215, 216
 - DL, deep learning, 39
 - dług techniczny, 369
 - długa pamięć krótkoterminowa, 47
 - DM, distributed memory, 121
 - dokumentacja zdrowotna i medyczna, 331
 - dostrajanie, 49
 - modelu BioBERT, 346
- ## E
- e-commerce, 302
 - analiza recenzji, 303, 317
 - budowanie katalogu, 306
 - hierarchia kategorii, 311
 - katalog produktów, 302
 - rekomendacje produktów, 304, 324
 - studium przypadku, 325
 - wyszukiwanie produktów, 303, 304
 - EHR, electronic health records, 334
 - ekstrakcja
 - atrybutów, 303, 306
 - bezpośrednia, 309
 - pośrednia, 310
 - cech, 80
 - encji prawniczych, 351
 - informacji, IE, 170
 - medycznych, 344
 - ogólny potok, 174
 - studium przypadku, 196
 - temporalnych, 192
 - zadania, 172
 - zastosowania, 171
 - latentnych atrybutów, 327
 - relacji, RE, 78, 188
 - podjęcie nadzorowane, 189
 - podjęcie oparte na wzorcach, 189
 - z wykorzystaniem Watson API, 190
 - słów lub fraz kluczowych, KPE, 172, 175
 - implementowanie, 176
 - tekstu, 64
 - zdarzeń, 193
 - elektroniczna dokumentacja medyczna, EHR, 334
 - elementy języka, 34
 - encje
 - nazwane
 - łączenie, 185, 186
 - rozpoznawanie, 177, 183
 - ujednoznacznianie, 185
 - tworzenie, 215, 216

EntityRuler, 183
ewaluacja, 87
 wewnętrzna, 88
 zewnętrzna, 90

F

fake newsy, 296
FAQ, frequently asked questions, 204
fastText, 148
finanse, 346
fonemy, 34

G

generowanie
 dialogu, 228
 języka naturalnego, 209
 odpowiedzi, 223
 zestawu danych QA, 339
graf bazy WordNet, 41
gramatyka bezkontekstowa, context-free
 grammar, 42

H

handel detaliczny, 302
HARVEST, 334–337
heurystyka, 40, 82
hipoteza dystrybucyjna, 109
HMM, Hidden Markov Model, 45

I

identyfikowanie
 memów, 295
 slotów, 222, 226
IDF, inverse document frequency, 107
IE, information extraction, 170
implementowanie KPE, 176
informacje temporalne, 192, 193
interfejs
 API, 139
 Azure Text Analytics API, 186
 chmurowy Dialogflow, 211
 do tłumaczenia maszynowego, 262
 Watson API, 190
 czatbota Rasa, 231
interpretowalność, 368
interwencja produktowa, 62

inżynieria cech, feature engineering, 80
 potok DL, 82
 potok NLP/ML, 80

J

język, 33
jonizator SMTD, 282

K

katalog produktów, 302
 deduplikacja produktów, 315
 dopasowywanie
 atrybutów, 316
 obrazów, 317
 tytułów, 316
 ekstrakcja atrybutów, 306
 kategoryzacja produktów, 311
 łączenie produktów, 327
 produkty
 komplementarne, 325
 substytucyjne, 325
 taksonomia produktów, 311
 wzbogacanie produktów, 313
klasyfikacja
 aktu dialogowego, 221
 tekstu, 133
 binarna, 134
 budowanie systemów, 137
 interfejsy API, 139
 interpretacja modeli, 158
 maszyna wektorów nośnych, 145
 naiwny klasyfikator bayesowski, 140
 osadzenia neuronowe, 147
 prosty klasyfikator, 138
 regresja logistyczna, 144
 sieci CNN, 155
 sieci LSTM, 156
 studium przypadku, 163
 uczenie głębokie, 152
 wieloetykietowa, 134
 wieloklasowa, 134
 wstępnie wytrenowane modele
 językowe, 157
 zastosowania, 134
klasyfikacje emocji, 343

klasyfikator, 138
 dyskryminacyjny, 144
 generatywny, 144
 probabilistyczny, 140
 wyjaśnianie prognoz, 159
kodowanie one-hot, 103
komponenty
 systemu dialogowego, 220
 wyszukiwarki, 244
konotacja, 109
kontekst, 36
kontekstowe reprezentacje tekstu, 122
KPE, keyword/keyphrase extraction, 172

L

LARA, latent aspect regression analysis, 321
leksemy, 34
leksykonowa analiza odczuć, 138
lemat, 74
LexNLP, 351
LSTM, long short-term memory, 153

Ł

łączenie nazwanych encji, NEL, 186
 z wykorzystaniem Azure API, 186

M

macierz błędów, 89, 146, 343
maszyna wektorów nośnych, SVM, 44, 145
media społecznościowe, 273
 fake newsy, 296
 memy, 295
 przetwarzanie danych SMTD, 281
memy
 identyfikowanie, 295
menedżer dialogu i zadań, 209
miary, 88
ML, machine learning, 39
model
 CBoW, 115
 przestrzeni wektorowej, VSM, 102
 SkipGram, 117
 Word2vec, 111, 113
modelowanie, 82
 budowanie modelu, 84
 heurystyka, 82

 tematyczne, 40, 240, 241, 249
 algorytmy, 250
 trenowanie modelu, 253
monitorowanie, 92, 368
morfemy, 34
MT, machine translation, 262
MVP, minimum viable product, 139

N

nadzór odległy, distant supervision, 189
naiwny klasyfikator bayesowski, 43, 140
nauka aktywna, 161
NED, named entity disambiguation, 185, 186
NEL, named entity linking, 186
NER, named entity recognition, 78, 172, 177
NLP, natural language processing, 29
NLU, natural language understanding, 209, 221
normalizacja tekstu, 76

O

obliczanie podobieństwa, 102
OCR, optical character recognition, 69
oczyszczanie tekstu, 64
 normalizacja Unikodu, 66
 oczyszczanie HTML-a, 64
 parsowanie, 64
 poprawianie błędów, 68
 poprawianie pisowni, 67
odczucia finansowe, 348
odpowiadanie na pytania, QA, 241, 264, 338
 tworzenie systemu, 266
OOV, out of vocabulary, 104, 148
opieka zdrowotna, 330
 asystenty zdrowotne, 333
 dokumentacja elektroniczna, 331, 344
 monitorowanie zdrowia psychicznego, 342
 nadzór farmakologiczny, 332
 rozliczanie pacjentów, 331
 ustalanie priorytetów, 331
optyczne rozpoznawanie znaków, OCR, 69
osadzenia, 110
 dokumentów, 150
 podśłów, 148
 słów, word embeddings, 110, 147
 wizualizacja, 123
 własne, 114
 wstępnie wytrenowane, 112

P

pamięć rozproszona, distributed memory, 121
plany działań, 93
podobieństwo, 102
 dystrybucyjne, 109
potok
 IE, 198
 klasyfikacji tekstu, 137
 NLP, 60, 360
 aktualizowanie modelu, 92
 ewaluacja, 87
 inżynieria cech, 80
 modelowanie, 82
 monitorowanie, 92
 oczyszczanie tekstu, 64
 pozyskiwanie danych, 61
 przetwarzanie wstępne, 70
 studium przypadku, 94
 wdrażanie, 91
 ogólny IE, 174
 systemu dialogowego, 209
 wyszukiwarki korporacyjnej, 246
prawda podstawowa, ground truth, 39
prawo, 350
problem OOV, 104, 120
proces
 data science, 374
 KDD, 374
 Microsoft Team Data Science, 375
produkty
 komplementarne, 325
 substytucyjne, 325
przesunięcie kowariantne, covariate shift, 363
przetwarzanie języka naturalnego, NLP, 29
 w danych SMTD, 281
 analiza odczuć, 285
 chmura wyrazowa, 281
 jonizator SMTD, 282
 obsługa klienta, 293
 popularne tematy, 283
 reprezentacja tekstu, 290
 wstępne przetwarzanie, 287
wstępne, 70
 lematyzacja, 74
 mieszanie kodów, 76
 normalizacja tekstu, 76
 przetwarzanie zaawansowane, 77

segmentacja zdań, 71
tematyzacja, 74
tokenizacja słów, 71
transliteracja, 76
 wykrywanie języka, 76
przewidywanie
 aktu dialogowego, 224
 wyników zdrowotnych, 340

R

rachunkowość i audyt, 349
Rasa NLU, 231
RE, relationship extraction, 188
recenzje, 317
regex, 41
RegexNER, 183
regresja logistyczna, 144
rekomendacje, 240, 241, 259
 praktyczne rady, 261
 produktów, 304
 tworzenie systemu, 260
 w e-commerce, 324
reprezentacja tekstu, 99
 dystrybucyjna, 110
 metody wektoryzacji, 102
 ręcznie utworzone reprezentacje cech, 126
 rozproszona, 109, 110, 120
 uniwersalna, 121
 w SMTD, 290
RNN, recurrent neural network, 46, 153
rozpoznawanie
 mowy, 209
 nazwanych encji, NER, 78, 172, 177
 biblioteka spaCy, 183
 budowanie systemu, 179
 dane treningowe, 181
 nauka aktywna, 183
rozszerzanie danych, 62
rozumienie języka naturalnego, NLU, 209, 221
rozwikływanie odniesień, 78
ryzyko kredytowe, 349

S

samouwaga, 48, 50
schemat blokowy
 analizy recenzji, 323
 potoku klasyfikacji tekstu, 137

- semantyka, 101
 - wektorowa, 110
 - sieci
 - CNN do klasyfikacji tekstu, 155
 - LSTM do klasyfikacji tekstu, 156
 - neuronowe konwolucyjne, CNN, 47, 153
 - neuronowe rekurencyjne, RNN, 46, 153
 - neuronowe z długą pamięcią krótkoterminową, LSTM, 153
 - SkipGram, 116, 117
 - składanie modeli, model stacking, 85
 - składnia, 35
 - słowa stopu, 73
 - SMTD, social media text data, 275
 - streszczanie tekstu, 240, 241, 255
 - konfigurowanie narzędzia streszczającego, 257
 - praktyczne rady, 257
 - zastosowania, 256
 - studium przypadku
 - ekstrakcja informacji, 196
 - obsługa zgłoszeń problemów, 163
 - polecenie przepisów, 233
 - produkty substytucyjne i komplementarne, 325
 - szeregowanie zgłoszeń, 94
 - wyszukiwarka dla księgarni, 248
 - SVM, support vector machine, 44, 145
 - system odpowiadania na pytania, 338
 - systemy
 - dialogowe, *Patrz także* czatboty
 - człowiek w pętli, 229
 - generowanie odpowiedzi, 223
 - identyfikacja slotów, 222, 226
 - klasyfikacja aktu dialogowego, 221
 - komponenty, 220
 - podejście kompleksowe, 228
 - potok, 208
 - przewidywanie aktu dialogowego, 224
 - szczegóły, 210
 - uczenie głębokie ze wzmocnieniem, 228
 - zbiory danych
 - ATIS, 224
 - DSTC, 224
 - MultiWoZ, 224
 - SNIPS, 224
 - odpowiadania na pytania, 264
 - rekomendujące dane tekstowe, 259
 - sztuczna inteligencja, AI, 38
 - cykl życia projektu, 383
 - dane i czas, 379
 - efektywność, 378
 - koszty, 381
 - nieprzestrzeganie właściwego procesu, 380
 - umiejętności zespołu, 377
 - zadanie ARC, 384
 - zwrot z inwestycji, 381
- Ś**
- ścieżki decyzyjne, 87, 92
- T**
- tablica pomyłek, 89
 - TensorFlow Model Analysis, 366
 - TextBlob, 286, 290
 - TextEvaluator, 127
 - TF, term frequency, 107
 - tłumaczenie maszynowe, MT, 241, 262
 - praktyczne rady, 263
 - używanie interfejsu API, 262
 - zwrotne, 63
 - transformatory, 48
 - trening wstępny, 49
 - modelu BioBERT, 346
 - trenowanie modelu tematycznego, 253
 - typy czatbotów, 207
- U**
- uczenie
 - głębokie, DL, 38, 39, 46, 51, 81
 - w klasyfikacji tekstu, 152
 - ze wzmocnieniem, 228
 - maszynowe, ML, 38, 39, 42
 - automatyzacja, 370
 - nadzorowane, supervised learning, 39
 - nienadzorowane, unsupervised learning, 39
 - się bez danych, 160
 - transferowe, 49, 51, 86, 162
 - ze wzmocnieniem, reinforcement learning, 39, 228
 - ujednoznacznianie nazwanych encji, NED, 185, 186
 - ukryty model Markowa, HMM, 45
 - UMLS, Unified Medical Language System, 336
 - uzupełnianie szablonów, 194

V

VSM, vector space model, 102

W

warunkowe pole losowe, CRF, 45, 180

wdrażanie, 91

oprogramowania NLP, 360

wektoryzacja, 102

kodowanie one-hot, 103

TF-IDF, 107

worek n-gramów, 106

worek słów, 104, 121

wizualizacja

danych MNIST, 124

osadzeń, 123

t-SNE, 125

Word2vec, 111, 113

worek

n-gramów, bag-of-n-grams, 106

słów, Bag of Words, 104

słów rozproszony, distributed bag
of words, 121

wstępnie wytrenowane modele językowe, 157

wyrażenia regularne, 41

wyszukiwanie, 241, 242

fasetowe, 306

w e-commerce, 304

wyszukiwarki

budowanie, 247

komponenty, 244

korporacyjne, 246

Z

zadanie

IE

ekstrakcja informacji temporalnych, 173,
192

ekstrakcja relacji, 173, 188

ekstrakcja słów/fraz kluczowych, 172

ekstrakcja zdarzeń, 173, 193

normalizacja informacji temporalnych,
192

rozpoznawanie nazwanych encji, 172, 177

ujednoznacznianie i łączenie nazwanych
encji, 173, 185

uzupełnianie szablonu, 173, 194

NLP, 32

modelowanie tematyczne, 241, 249

odpowiadanie na pytania, 241, 264

rekomendacje, 241, 259

streszczanie tekstu, 241, 255

tłumaczenie maszynowe, 241, 262

wyszukiwanie, 241, 242

zarządzanie relacjami z klientami, CRM, 293

zastosowanie

czatboty, 204

IE, 171

klasyfikacja tekstów, 134

NLP, 30

NLP w finansach, 348

streszczanie tekstów, 256

zespalande modeli, model ensembling, 85

zwracanie informacji, 242

PROGRAM PARTNERSKI

— GRUPY HELION —



1. ZAREJESTRUJ SIĘ
2. PREZENTUJ KSIĄŻKI
3. ZBIERAJ PROWIZJĘ

Zmień swoją stronę WWW w działający bankomat!

Dowiedz się więcej i dołącz już dzisiaj!

<http://program-partnerski.helion.pl>

GRUPA
Helion 

Ta książka pokazuje anatomię rzeczywistych systemów!

Zachary Lipton, naukowiec i autor książki *Dive Into Deep Learning*

Systemy przetwarzania języka naturalnego charakteryzuje złożoność i unikatowość. Większość podręczników ogranicza się do omówienia problematyki NLP na uproszczonych przykładach i dobrze zdefiniowanych zbiorach danych. Zawarta w nich wiedza jednak nie wystarczy, aby rozwiązać pojawiające się problemy, a następnie zbudować i wdrożyć rzeczywistą aplikację opartą na NLP, z uwzględnieniem specyfiki danej branży i z poszanowaniem najlepszych praktyk.

Książka jest adresowana do wszystkich, którzy chcą budować, rozwijać i skalować systemy NLP w środowisku biznesowym, a także dostosowywać je do swojej branży. Opisuje tworzenie rzeczywistych aplikacji NLP. Omawia pełny cykl życia typowego projektu NLP, od zbierania danych po wdrożenie i monitorowanie modelu. Przedstawia studia przypadków i przewodniki dziedzinowe, pozwalające na zbudowanie systemu NLP od podstaw. Wyczerpująco wyjaśnia, w jaki sposób adaptować rozwiązania do potrzeb różnych branż, takich jak opieka zdrowotna, handel detaliczny i media społecznościowe. Prezentuje szeroką gamę zadań, od klasyfikacji tekstu poprzez odpowiadanie na pytania po ekstrakowanie informacji i systemy dialogowe. Poszczególne zagadnienia są zilustrowane fragmentami kodu, ułatwiającymi zrozumienie logiki omawianych systemów.

W książce między innymi:

- najważniejsze koncepcje związane z NLP
- implementowanie aplikacji NLP z wykorzystaniem uczenia maszynowego
- dostrajanie rozwiązań NLP do konkretnych problemów biznesowych
- skuteczne techniki wydawania, wdrażania i rozwijania systemów NLP
- najlepsze praktyki i strategie NLP dla liderów biznesowych

Sowmya Vajjala pracuje w kanadyjskiej Narodowej Radzie Badań Naukowych. Budowała wielojęzyczne systemy NLP.

Bodhisattwa Majumder jest doktorantem na Uniwersytecie Kalifornijskim w San Diego. Tworzył systemy NLP w Google AI i Microsoft Research.

Anuj Gupta jest dyrektorem w firmie Vahan. Kierował wieloma zespołami zajmującymi się uczeniem maszynowym.

Harshit Surana jest współzałożycielem firmy DeepFlux. Prowadził badania nad NLP i uczeniem maszynowym w MIT Media Lab.

	KOD KORZYŚCI Sięgnij po więcej! ▶	
 helion.pl	ISBN 978-83-8322-726-9	
 HELION SA ul. Kościuszki 1c 44-100 Gliwice tel. 32 250 99 63 helion@helion.pl	 9 788383 227269	
Cena: 109,00 zł		