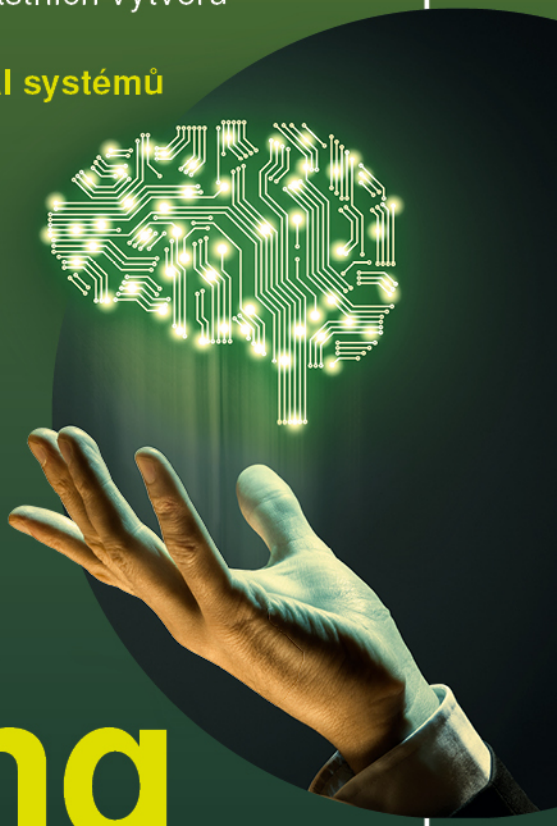
EDICE
myslíme v ...

knihovna programátora

- Výklad základních principů hlubokého učení i pokročilých dovedností
- Tvorba systému hlubokého učení pro počítačové vidění, časové řady, text i generování vlastních výtvorů (například obrázků)
- **Způsob fungování moderních AI systémů typu ChatGPT**
- Popis rozdílů při spouštění programů na CPU, GPU a FPU
- Práce s webovým prostředím Collaboration, které umožňuje používat GPU a FPU na serveru



FRANÇOIS CHOLLET

Deep learning

v jazyku Python

2., přepracované a rozšířené vydání

KNIHOVNY KERAS, TENSORFLOW

MANNING

GRADA

EDICE
myslíme v ...

knihovna programátora

FRANÇOIS CHOLLET

Deep learning

v jazyku Python

KNIHOVNY KERAS, TENSORFLOW

2., přepracované a rozšířené vydání

GRADA
Publishing

François Chollet

Deep Learning v jazyku Python

Knihovny Keras, TensorFlow
2., přepracované a rozšířené vydání

Přeloženo z anglického originálu knihy Françoise Cholleta Deep Learning with Python, vydaného v roce 2021 nakladatelstvím Manning Publications Co, Spojené státy americké.

Vydala Grada Publishing, a.s.
U Průhonu 22, Praha 7
obchod@grada.cz, www.grada.cz
tel.: +420 234 264 401
jako svou 8701. publikaci

Přeložil: Rudolf Pecinovský
Odpovědný redaktor: Petr Somogyi
Návrh vnitřního layoutu a zlom: Rudolf Pecinovský
Počet stran 528
První vydání, Praha 2023
Vytiskla TISKÁRNA V RÁJI, s.r.o., Pardubice

ISBN 978-80-271-7081-4 (pdf)
ISBN 978-80-271-5133-2 (print)

Stručný obsah

Stručný obsah 5
Podrobný obsah 6
Předmluva 14
Poděkování 16
O knize 17
O autorovi 21
O obálce 22
Kapitola 1 **Co je hluboké učení** 24
Kapitola 2 **Než začneme: matematické** stavební bloky neuronových sítí 51
Kapitola 3 Úvod do Keras a TensorFlow 93
Kapitola 4 **Začínáme s neuronovými sítěmi: Klasifikace** a regrese 121
Kapitola 5 **Základy strojového učení** 148
Kapitola 6 **Univerzální pracovní postup strojového učení** 181
Kapitola 7 **Ponořme se do práce s** Keras 201
Kapitola 8 **Úvod do hlubokého učení pro počítačové vidění** 233
Kapitola 9 **Pokročilé hluboké učení pro počítačové vidění** 270
Kapitola 10 **Hluboké učení pro časové řady** 313
Kapitola 11 **Hluboké učení pro text** 343
Kapitola 12 **Generativní hluboké učení** 402
Kapitola 13 **Osvědčené postupy pro reálný svět** 453
Kapitola 14 **Závěry** 474
Příloha A Terminologický slovník 512
Rejstřík 523

Podrobný obsah

Stručný obsah 5
Podrobný obsah 6
Předmluva 14
Poděkování 16
O knize 17
Komu je kniha určena 17
O kódu 18
Diskusní fórum liveBook 19
Použité typografické konvence 19
Odbočka – podšeděný blok 20
O autorovi 21
O obálce 22
Kapitola 1 **Co je hluboké učení** 24
1.1 **Umělá inteligence, strojové učení a hluboké učení** 24
1.1.1 **Umělá inteligence** (artificial intelligence) 25
1.1.2 **Strojové učení** 26
1.1.3 **Učení se pravidel a reprezentaci z dat** 28
1.1.4 **Hloubka v hlubokém učení** 30
1.1.5 **Pochopení toho, jak hluboké učení funguje, ve třech krocích** 32
1.1.6 **Čeho hluboké učení dosud dosáhlo** 34
1.1.7 **Nevěřte krátkodobému humbuku** 35
1.1.8 **Příslib AI** 36
1.2 **Před hlubokým učením: stručná historie strojového učení** 37
1.2.1 **Pravděpodobnostní** modelování 38
1.2.2 **Rané neuronové sítě** 38
1.2.3 Jádrové metody (kernel methods) 38
1.2.4 Rozhodovací stromy, náhodné lesy a stroje na posílení gradientu 40
1.2.5 **Zpět k neuronovým sítím** 41
1.2.6 **Co dělá hluboké učení odlišným** 42
1.2.7 **Krajina moderního strojového učení** 43
1.3 **Proč hluboké učení? Proč teď?** 45
1.3.1 Hardware 45
1.3.2 Data 46
1.3.3 Algoritmy 47
1.3.4 Nová vlna investic 48
1.3.5 **Demokratizace hlubokého učení** 49
1.3.6 Vydrží to? 49
Kapitola 2 **Než začneme: matematické** stavební bloky neuronových sítí 51
2.1 První pohled na **neuronovou síť** 52
2.2 **Reprezentace dat pro neuronové sítě** 56
2.2.1 **Skaláry (tenzory nultého řádu, 0D tenzory)** 56

2.2.2 **Vektory (tenzory 1. řádu, 1D tenzory)** ... 57
2.2.3 **Matice (tenzory 2. řádu, 2D tenzory)** ... 57
2.2.4 **Tenzory tří a vícedimenzionální** ... 57
2.2.5 **Klíčové atributy** ... 58
2.2.6 Manipulace s tenzory v *NumPy* ... 59
2.2.7 Pojem dávek dat ... 60
2.2.8 **Příklady datových tenzorů v reálném světě** ... 60
2.2.9 Vektorová data ... 61
2.2.10 **Časové řady nebo sekvenční data** ... 61
2.2.11 Obrazová data ... 62
2.2.12 Video data ... 62
2.3 Nástroje neuronových sítí: tenzorové operace ... 63
2.3.1 Elementové operace (element-wise operations) ... 64
2.3.2 Vysílání (broadcasting) ... 65
2.3.3 **Tenzorový součin** ... 66
2.3.4 **Změna tvaru (přetváření) tenzoru** (tensor reshaping) ... 69
2.3.5 Geometrická interpretace tenzorových operací ... 69
2.3.6 **Geometrická interpretace hlubokého učení** ... 73
2.4 Motor neuronových sítí: optimalizace založená na gradientu ... 73
2.4.1 Co je derivace? ... 75
2.4.2 Derivace tenzorové operace: gradient ... 76
2.4.3 Stochastický gradientní sestup ... 77
2.4.4 **Zřetězení derivací: algoritmus zpětného šíření** ... 80
Pravidlo řetězu ... 81
Automatická derivace s výpočetními grafy ... 81
`GradientTape` v TensorFlow ... 86
2.5 **Ohlédnutí za naším prvním příkladem** ... 86
2.5.1 **Reimplementace prvního příkladu od nuly v TensorFlow** ... 88
Jednoduchá třída `NaiveDense` ... 88
Jednoduchá sekvenční třída ... 89
Generátor dávek ... 89
2.5.2 **Běh jednoho tréninkového kroku** ... 90
2.5.3 Úplný trénovací cyklus ... 91
2.5.4 Vyhodnocení modelu ... 91
2.6 Shrnutí kapitoly ... 92
Kapitola 3 Úvod do Keras a TensorFlow ... 93
3.1 Co je TensorFlow ... 93
3.2 Co je Keras ... 94
3.3 **Keras a TensorFlow: stručná historie** ... 96
3.4 **Nastavení pracovního prostoru pro hluboké učení** ... 96
3.4.1 Notebooky Jupyter: **preferovaný způsob provádění experimentů s hlubokým učením** ... 97
3.4.2 Používání **laboratoře Colaboratory** ... 98
První kroky s Colaboratory ... 98
Instalace balíčků pomocí PIP ... 100
Použití běhového prostředí GPU ... 100
3.5 První kroky s TensorFlow ... 101
3.5.1 **Konstantní tenzory a proměnné** ... 101
3.5.2 **Operace s tenzory: provádění** matematických operací v TensorFlow ... 104
3.5.3 Druhý pohled na API `GradientTape` ... 104
3.5.4 **Souhrnný příklad: lineární klasifikátor v čistém TensorFlow** ... 105
3.6 **Anatomie neuronové sítě:** pochopení základních rozhraní Keras API ... 110
3.6.1 **Vrstvy: základní kameny hlubokého učení** ... 110
`Layer` – základní třída vrstvy v *Keras* ... 111
Automatické odvozování tvaru: vytváření vrstev za běhu ... 112
3.6.2 **Od vrstev k modelům** ... 113

3.6.3 **Krok „překlad“: konfigurace procesu učení** ... 115
3.6.4 **Výběr ztrátové funkce** ... 117
3.6.5 **Porozumění metodě `fit()`** ... 117
3.6.6 **Monitorování ztrát a metrik na validačních datech** ... 118
3.6.7 Odvození: použití modelu po tréninku ... 119
3.7 Shrnutí kapitoly ... 120
Kapitola 4 **Začínáme s neuronovými sítěmi: Klasifikace** a regrese ... 121
Glosář klasifikace a regrese ... 121
4.1 **Klasifikace filmových recenzí: příklad** binární klasifikace ... 123
4.1.1 Databáze IMDB ... 123
4.1.2 **Příprava dat** ... 124
4.1.3 Sestavení modelu ... 125
Co jsou aktivační funkce a proč jsou potřebné ... 128
4.1.4 **Ověření vašeho přístupu** ... 128
4.1.5 **Použití vycvičeného modelu k vytváření předpovědí** na nových datech ... 131
4.1.6 Další experimenty ... 132
4.1.7 Shrnutí ... 132
4.2 **Klasifikace zpravodajství: příklad klasifikace do více tříd** ... 133
4.2.1 Soubor dat *Reuters* ... 133
4.3.2 **Příprava dat** ... 134
4.2.3 Sestavení modelu ... 135
4.2.4 **Ověření vašeho přístupu** ... 136
4.2.5 Generování predikcí pro nová data ... 138
4.2.6 **Jiný způsob zacházení se štítky a ztrátou** ... 138
4.2.7 **Důležitost dostatečně velkých mezivrstev** ... 139
4.2.8 Další experimenty ... 139
4.2.9 Shrnutí ... 139
4.3 **Predikce cen nemovitostí: příklad regrese** ... 140
4.3.1 **Soubor údajů o cenách bydlení v Bostonu** ... 140
4.3.2 **Příprava dat** ... 141
4.3.3 Sestavení modelu ... 141
4.3.4 **Ověření vašeho přístupu použitím k**-násobné validace ... 142
4.3.5 **Generování předpovědí na nových datech** ... 146
4.3.5 Shrnutí ... 146
4.4 Shrnutí kapitoly ... 147
Kapitola 5 **Základy strojového učení** ... 148
5.1 **Zobecnění: cíl strojového učení** ... 148
5.1.1 **Přeučení a podučení** ... 149
Zašuměná trénovací data ... 149
Nejednoznačné rysy ... 151
Vzácné rysy a falešné korelace ... 151
5.1.2 **Povaha zobecnění v hlubokém učení** ... 154
Hypotéza o mnohotvárnosti ... 155
Interpolace jako zdroj zobecnění ... 156
Proč hluboké učení funguje ... 157
Trénovací data jsou nejdůležitější ... 158
5.2 **Vyhodnocování modelů strojového učení** ... 160
5.2.1 **Trénovací, validační a testovací sady** ... 160
Jednoduchá validace na odebraných datech (simple hold-out validation) ... 161
K-násobné křížová validace ... 162
Opakovaná k-násobná validace s promícháním ... 163
5.2.2 **Překonání základní referenční úrovně** ... 163
5.2.2 **Věci, které je třeba mít na paměti** ... 164
5.3 **Zlepšení odpovědí modelu** ... 165
5.3.1 **Ladění klíčových parametrů gradientního sestupu** ... 165
5.3.2 **Využití lepších předloh architektury** ... 166

5.3.3 Zvyšování kapacity modelu ... 167
5.4 **Zlepšení zobecnění** ... 169
5.4.1 Kurátorství datových sad ... 169
5.4.2 **Inženýrství rysů** ... 170
5.4.3 **Včasné zastavení** ... 171
5.4.4 Regularizace modelu ... 172
Zmenšení velikosti sítě ... 172
Přidání regularizace vah ... 175
Přidání dropout ... 177
5.5 Shrnutí kapitoly ... 179
Kapitola 6 **Univerzální pracovní postup strojového učení** ... 181
Poznámka k etice ... 182
6.1 Definice úkolu ... 183
6.1.1 Rámcový popis problému ... 183
6.1.2 **Shromáždění souboru dat** ... 185
Investice do infrastruktury pro anotaci dat ... 186
Pozor na nereprezentativní údaje ... 186
Problém zkreslení výběru vzorku ... 187
6.1.3 **Porozumění datům** ... 188
6.1.4 **Zvolte si měřítko úspěchu** ... 189
6.2 Vyvinutí modelu ... 189
6.2.1 **Příprava dat** ... 189
Vektorizace ... 190
Normalizace hodnot ... 190
Zpracování chybějících hodnot ... 190
6.2.2 **Výběr hodnoticího protokolu** ... 191
6.2.3 **Překonání základní referenční úrovně** ... 191
Výběr správné ztrátové funkce ... 192
6.2.4 **Rozšíření: vývoj modelu, který se přizpůsobí** ... 193
6.2.5 **Regularizace a vyladění modelu** ... 193
6.3 Nasazení modelu ... 194
6.3.1 **Vysvětlete svou práci zúčastněným stranám a** stanovte **očekávání** ... 194
6.3.2 Odeslání odvozovacího modelu ... 195
Nasazení modelu jako REST API ... 195
Nasazení modelu v zařízení ... 196
Nasazení modelu v prohlížeči ... 197
Optimalizace odvozovacího (inferenčního) modelu ... 198
6.3.3 **Sledování modelu „v divočině“** ... 198
6.3.4 **Udržujte svůj model** ... 199
6.4 Shrnutí kapitoly ... 199
Kapitola 7 **Ponořme se do práce s** Keras ... 201
7.1 **Spektrum pracovních postupů** ... 202
7.2 **Různé způsoby vytváření modelů** *Keras* ... 202
7.2.1 **Sekvenční model** ... 203
7.2.2 Funkcionální API ... 205
Jednoduchý příklad ... 205
Modely s více vstupy a výstupy ... 207
Trénování modelu s více vstupy a výstupy ... 208
Síla funkcionálního API: přístup k propojení vrstev ... 209
7.2.3 **Definice potomků třídy `Model`** ... 211
Přepsání předchozího příkladu jako modelu s podtřídou ... 211
Pozor: co modely s podtřídou nepodporují ... 213
7.2.4 **Míchání a kombinování různých součástí** ... 213
7.2.5 Pamatujte: pro danou práci používejte správný nástroj ... 214
7.3 **Použití vestavěných trénovacích a** vyhodnocovacích **cyklů** ... 215
7.3.1 Psaní vlastních metrik ... 216

7.3.2 **Použití zpětných volání** ... 217
Zpětná volání `EarlyStopping` a `ModelCheckpoint` ... 218
7.3.3 **Psaní vlastních zpětných volání** ... 219
7.3.4 Monitorování a vizualizace pomocí TensorBoard ... 221
7.4 **Psaní vlastních školicích a hodnoticích cyklů** ... 223
Další druhy učení ... 224
7.4.1 Školení versus odvozování ... 225
7.4.2 Použití metrik na nízké úrovni ... 226
7.4.3 Kompletní cyklus školení a hodnocení ... 226
7.4.4 Zrychlete pomocí funkce `tf.function` ... 228
7.4.5 Využití funkce `fit()` s vlastním trénovacím cyklem ... 229
7.5 Shrnutí kapitoly ... 231
Kapitola 8 **Úvod do hlubokého učení pro počítačové vidění** ... 233
8.1 **Úvod do konvolučních neuronových sítí** – CNN ... 234
8.1.1 **Konvoluční operace** ... 236
Porozumění hraničním efektům a vycpávkám ... 239
Porozumění konvolučním krokům ... 240
8.1.2 Operace sdružování dle maxima (max pooling) ... 241
8.2 **Trénování CNN od nuly na malé sadě dat** ... 243
8.2.1 **Význam hlubokého učení pro úlohy s malými daty** ... 243
8.2.2 Stažení dat ... 244
Stažení datové sady Kaggle v nástroji Google Colaboratory ... 245
8.2.3 Sestavení modelu ... 247
8.2.4 **Předzpracování dat** ... 249
Porozumění objektům `Dataset` frameworku TensorFlow ... 249
8.2.5 **Použití rozšíření dat** (data augmentation) ... 253
8.3 **Použití předtrénované CNN** ... 256
8.3.1 **Extrakce rysů** ... 257
Rychlá extrakce rysů bez rozšíření dat ... 261
Extrakce rysů s rozšířením dat ... 263
8.3.2 **Jemné doladění předtrénovaného modelu** ... 265
8.4 Shrnutí kapitoly ... 268
Kapitola 9 **Pokročilé hluboké učení pro počítačové vidění** ... 270
9.1 **Tři základní úlohy počítačového vidění** ... 270
9.2 **Příklad segmentace obrazu** ... 272
9.3 Moderní vzory architektury CNN ... 280
9.3.1 Modularita, hierarchie a opakované použití ... 281
O významu ablačních studií ve výzkumu hlubokého učení ... 283
9.3.2 **Zbytková připojení** (residual connections) ... 283
9.3.3 Dávková normalizace ... 287
9.3.4 **Hloubkově oddělitelné konvoluce** ... 289
Koevoluce hardwaru, softwaru a algoritmů ... 290
9.3.5 Složení: mini model podobný Xception ... 292
9.4 **Interpretace toho, co se CNN naučí** ... 294
9.4.1 Vizualizace mezilehlých aktivací ... 294
9.4.2 **Vizualizace filtrů CNN** ... 300
Rozdíl mezi `model.predict(x)` a `model(x)` ... 302
9.4.3 **Zobrazení teplotních map aktivací třídy** ... 306
9.5 Shrnutí kapitoly ... 312
Kapitola 10 **Hluboké učení pro časové řady** ... 313
10.1 **Různé druhy úloh časových řad** ... 313
10.2 **Příklad předpovědi teploty** ... 314
Vždy hledejte v datech periodicitu ... 317
10.2.1 **Příprava dat** ... 317
Porozumění `timeseries_dataset_from_array()` ... 318

10.2.2 **Základní referenční úroveň bez strojového učení** ... 320
10.2.3 **Vyzkoušejme základní model strojového učení** ... 321
10.2.4 **Vyzkoušejme 1D konvoluční model** ... 323
10.2.5 **První rekurentní základní referenční úroveň** ... 325
10.3 **Porozumění rekurentním neuronovým sítím** ... 326
10.3.1 Rekurentní vrstvy v *Keras* ... 329
10.4 **Pokročilé využití rekurentních neuronových sítí** ... 333
10.4.1 **Použití rekurentního dropoutu v boji proti přeučení** ... 334
Výkonnost RNN za běhu ... 335
10.4.2 Stohování rekurentních vrstev ... 336
10.4.3 **Použití obousměrných RNN** ... 337
10.4.4 **Pokračujme ještě dále** ... 340
Trhy a strojové učení ... 341
10.5 Shrnutí kapitoly ... 341
Kapitola 11 **Hluboké učení pro text** ... 343
11.1 Zpracování **přirozeného jazyka: z ptačí** perspektivy ... 343
11.2 **Příprava textových dat** ... 345
11.2.1 Standardizace textu ... 346
11.2.2 **Dělení textu (tokenizace)** ... 347
Porozumění N-gramům a sadám slov ... 348
11.2.3 Indexování slovní zásoby ... 348
11.2.4 Použití vrstvy `TextVectorization` ... 350
Použití vrstvy `TextVectorization` v datovodu `tf.data` nebo jako **součást** modelu ... 352
11.3 **Dva přístupy k reprezentaci skupin slov: sady** a posloupnosti ... 354
11.3.1 **Příprava dat recenzí filmů na IMDB** ... 354
11.3.2 **Zpracování slov jako sady: přístup sada slov (**Bag-of-Words approach) ... 356
Jednotlivá slova s binárním kódováním (1-gramy, unigramy) ... 357
Bigramy (2-gramy) s binárním kódováním ... 359
Bigramy s kódováním TF-IDF ... 360
Porozumění normalizaci TF-IDF ... 361
Export modelu, který zpracovává nezpracované řetězce ... 362
11.3.3 **Zpracování slov jako sekvence: přístup sekvenčního modelu** ... 363
První praktický příklad ... 363
Porozumění slovním vnořením ... 365
Učení vkládání slov pomocí vrstvy `Embedding` ... 367
Porozumění výplním a maskování ... 368
Použití předem natrénovaných slovních vnoření ... 370
11.4 Architektura Transformátor ... 372
11.4.1 **Porozumění sebepozornosti** ... 373
Zobecněná sebepozornost: model dotaz-klíč-hodnota ... 376
11.4.2 Pozornost více hlav (multi-head attention) ... 377
11.4.3 Transformátor ... 378
Ukládání vlastních vrstev ... 380
Použití pozičního kódování k opětovnému vložení informací o pořadí ... 383
Všechno dohromady: transformátor pro klasifikaci textu ... 384
11.4.4 **Kdy použít sekvenční modely místo modelů typu „sada slov“?** ... 385
11.5 **Nad rámec klasifikace textu: učení sekvence na sekvenci** ... 386
11.5.1 **Příklad strojového překladu** ... 387
11.5.2 **Učení sekvence na sekvenci pomocí RNN** ... 390
11.5.3 **Učení sekvence na sekvenci pomocí Transformátoru** ... 395
Dekodér Transformátoru ... 396
Vše dohromady: Transformátor pro strojový překlad ... 398
11.6 Shrnutí kapitoly ... 401
Kapitola 12 **Generativní hluboké učení** ... 402
12.1 Generování textu ... 404
12.1.1 **Stručná historie generativního hlubokého učení pro** generování sekvencí ... 404

12.1.2 **Jak generujete sekvenční data?** 405
12.1.3 **Význam strategie výběru vzorků** 406
12.1.4 Implementace generování textu pomocí Keras 407
Příprava dat 408
Model založený na transformátoru sekvence-sekvenci 409
12.1.5 **Zpětné volání pro generování textu se vzorkováním s proměnnou teplotou** 411
12.1.6 **Závěrečné shrnutí** 415
12.2 DeepDream 415
12.2.1 Implementace DeepDream v Keras 416
12.2.2 **Závěrečné shrnutí** 422
12.3 **Přenos neuronového stylu** 423
12.3.1 Ztráta obsahu 424
12.3.2 Ztráta stylu 424
12.3.3 **Přenos neuronového stylu v Keras** 425
12.3.4 **Závěrečné shrnutí** 431
12.4 **Generování obrazů pomocí variačních autoenkodérů** 431
12.4.1 **Vzorkování z latentních prostorů obrazů** 431
12.4.2 **Koncepční vektory pro úpravy obrázků** 432
12.4.3 **Variační autoenkodéry** 433
12.4.4 Implementace VAE pomocí Keras 436
12.4.5 **Závěrečné shrnutí** 441
12.5 **Úvod do generativních soupeřících sítí** 442
12.5.1 Schematická implementace GAN 443
12.5.2 **Sada triků** 444
12.5.3 Získání souboru dat CelebA 445
12.5.4 Diskriminátor 446
12.5.5 Generátor 447
12.5.6 **Soupeřící síť** 448
12.5.7 **Závěrečné shrnutí** 451
12.6 Shrnutí kapitoly 452
Kapitola 13 **Osvědčené postupy pro reálný svět** 453
13.1 **Využití vašich modelů na maximum** 454
13.1.1 **Optimalizace hyperparametrů** 454
Používání KerasTuneru 455
Maximalizace a minimalizace cíle optimalizace 457
Umění vytvořit správný vyhledávací prostor 460
Budoucnost ladění hyperparametrů: automatizované strojové učení 460
13.1.2 **Kombinování modelů** 461
13.2 **Rozšiřování školení modelů** 463
13.2.1 **Zrychlení tréninku na GPU se smíšenou přesností** 464
Porozumění přesnosti reálných čísel 464
Poznámka ke kódování reálných čísel 465
Pozor na výchozí hodnoty `dtype` 466
Trénink se smíšenou přesností v praxi 467
13.2.2 Školení s více grafickými procesory 467
Získání dvou nebo více GPU 468
Synchronní školení s jedním hostitelem a více zařízeními 468
Tipy pro výkon `tf.data` 469
13.2.3 Trénování TPU 470
Použití TPU přes Google Colab 470
Pozor na úzká místa I/O 472
Využití slučování kroků ke zlepšení využití TPU 472
13.3 Shrnutí kapitoly 472
Kapitola 14 **Závěry** 474
14.1 **Přehled klíčových pojmů** 474
14.1.1 **Různé přístupy k AI** 475

14.1.2 **Čím je hluboké učení v oblasti strojového učení výjimečné** ... 475
14.1.3 **Jak uvažovat o hlubokém učení** ... 476
14.1.4 **Klíčové podpůrné technologie** ... 477
14.1.5 **Univerzální pracovní postup strojového učení** ... 478
14.1.6 **Klíčové síťové architektury** ... 479
Hustě propojené sítě ... 480
CNN (konvoluční neuronové sítě) ... 481
RNN ... 482
Transformátory ... 482
14.1.7 Prostor možností ... 483
14.2 **Omezení hlubokého učení** ... 485
14.2.1 **Riziko antropomorfizace modelů strojového učení** ... 486
14.2.2 Automaty vs. inteligentní agenti ... 488
14.2.3 Lokální generalizace vs. extrémní generalizace ... 489
14.2.4 **Účel inteligence** ... 491
14.2.5 **Šplhání po spektru zobecnění** ... 492
14.3 **Nastavení směru k větší obecnosti umělé inteligence** ... 493
14.3.1 **O důležitosti stanovení správného cíle: pravidlo zkratky** ... 493
14.3.2 Nový cíl ... 495
14.4 **Zavádění inteligence: chybějící ingredience** ... 497
14.4.1 Inteligence jako citlivost na abstraktní analogie ... 497
14.4.2 Dva póly abstrakce ... 498
Analogie zaměřená na hodnoty (value-centric analogy) ... 499
Analogie zaměřená na program (program-centric analogy) ... 500
Poznávání jako kombinace obou druhů abstrakce ... 501
14.4.3 **Chybějící polovina obrázku** ... 501
14.5 **Budoucnost hlubokého učení** ... 502
14.5.1 Modely jako programy ... 503
14.5.2 **Spojení hlubokého učení a syntézy programů** ... 504
Integrace modulů hlubokého učení a algoritmických modulů do hybridních systémů ... 505
Využití hlubokého učení k vyhledávání programů ... 506
14.5.3 **Celoživotní učení a modulární opakované použití podprogramů** ... 507
14.5.4 Dlouhodobá vize ... 508
14.6 **Zůstaňte v obraze v rychle se vyvíjející oblasti** ... 509
14.6.1 **Cvičení na reálných úlohách pomocí Kaggle** ... 510
14.6.2 **Přečtěte si o nejnovějším vývoji na arXiv** ... 510
14.6.3 Prozkoumejte ekosystém Keras ... 511
14.7 **Závěrečné slovo** ... 511
Příloha A Terminologický slovník ... 512
A.1 Používané zkratky ... 512
GRU ... 513
A.2 **Řazený dle anglických termínů** ... 513
A.3 **Řazený dle českých termínů** ... 518
Rejstřík ... 523

Předmluva

Držíte-li tuto knihu v ruce, jste si asi vědomi mimořádného pokroku, jehož hluboké učení pro oblast umělé inteligence v nedávné době dosáhlo. Za pouhých pět let jsme se od téměř nepoužitelného počítačového vidění a zpracování přirozeného jazyka dostali až k vysoce výkonným systémům nasazeným ve velkém měřítku v produktech, které používáme každý den.

Důsledky tohoto náhlého pokroku zasahují téměř do všech odvětví. Hluboké učení již uplatňujeme v rozsáhlé škále důležitých problémů v tak odlišných oblastech, jako je lékařské zobrazování, zemědělství, autonomní řízení, vzdělávání, prevence katastrof a výroba.

Přesto se domnívám, že hluboké učení je stále v počátcích. Zatím využilo jen malou část svého potenciálu. Postupem času se dostane do všech oblastí, kde může pomoci – tato transformace bude trvat několik desetiletí.

Abychom mohli začít používat technologii hlubokého učení na všechny problémy, které by mohla vyřešit, musíme ji zpřístupnit co největšímu počtu lidí, včetně těch, kteří nejsou odborníky: tedy i lidem, kteří nejsou výzkumníky nebo postgraduálními studenty. Aby hluboké učení dosáhlo plného potenciálu, musíme ho radikálně zdemokratizovat.

Domnívám, že nyní jsme na vrcholu historického přechodu, kdy hluboké učení opouští vědecké laboratoře a výzkumná či vývojová oddělení velkých technologických společností a stává se všudypřítomnou součástí sady nástrojů každého vývojáře, ne nepodobně trajektorii vývoje webových stránek na konci 90. let. Téměř každý si nyní může pro svou firmu nebo komunitu vytvořit webové stránky nebo webovou aplikaci, k jejichž vytvoření by v roce 1998 potřeboval malý tým inženýrů specialistů. V nepříliš vzdálené budoucnosti bude moci kdokoli s nápadem a základními dovednostmi v oblasti kódování vytvářet inteligentní aplikace, které se budou učit z dat.

Když jsem v březnu 2015 vydal první verzi frameworku *Keras* pro hluboké učení, neměl jsem na mysli demokratizaci umělé inteligence. Již několik let jsem se zabýval výzkumem v oblasti strojového učení a vytvořil jsem *Keras*, aby mi pomohl při mých vlastních experimentech.

Od roku 2015 však do oblasti hlubokého učení vstoupily statisíce nováčků; mnozí z nich si *Keras* vybrali jako svůj nástroj. Když jsem sledoval, jak desítky chytrých lidí používají *Keras* nečekaným a efektivním způsobem, začal jsem se velmi zajímat o dostupnost a demokratizaci umělé inteligence. Uvědomil jsem si, že čím více se tyto technologie šíří, tím jsou užitečnější a cennější. Přístupnost se rychle stala cílem vývoje

Keras a během několika krátkých let dosáhla komunita vývojářů v tomto směru fantastických úspěchů. Dali jsme hluboké učení do rukou statisíců lidí, kteří ho následně používají k řešení problémů, jež byly donedávna považovány za neřešitelné.

Kniha, kterou držíte v ruce, je dalším krokem na cestě k zpřístupnění hlubokého učení co největšímu počtu lidí. *Keras* vždy potřeboval doprovodný kurz, který by se současně zabýval základy hlubokého učení, osvědčenými postupy a způsoby použití frameworku *Keras*.

V letech 2016 a 2017 jsem se snažil takový kurz vytvořit, což vedlo k první verzi této knihy, která vyšla v prosinci 2017. Rychle se stala bestsellerem v oblasti strojového učení, prodalo se jí přes 50 000 výtisků a byla přeložena do 12 jazyků.

Oblast hlubokého učení však rychle postupuje vpřed. Od prvního vydání došlo k mnoha důležitým změnám – objevilo se *TensorFlow 2*, vzrostla popularita architektury *Transformátor* a mnoho dalšího. A tak jsem se koncem roku 2019 rozhodl svou knihu aktualizovat.

Původně jsem si docela naivně myslel, že bude obsahovat zhruba 50 % nového obsahu, nakonec bude zhruba stejně dlouhá jako první vydání. V praxi se po dvou letech práce ukázalo, že je více než o třetinu delší a obsahuje asi 75 % nového obsahu. Spíše než o aktualizaci tedy jde o zcela novou knihu.

Napsal jsem ji s důrazem na to, aby koncepty hlubokého učení a jejich implementace byly co nejpřístupnější. Přitom jsem nemusel nic hloupě vysvětlovat, protože pevně věřím, že v hlubokém učení nejsou žádné obtížné myšlenky. Doufám, že pro vás bude tato kniha cenná a že vám umožní začít vytvářet inteligentní aplikace a řešit problémy, na kterých vám záleží.

Poznámka překladatele a redakce k názvu knihy: V předkládané publikaci se snažíme maximálně dodržovat českou terminologii, ale u názvu knihy jsme dali přednost anglickému termínu *deep learning*. Domníváme se, že je ve větším povědomí mezi čtenáři než jeho český ekvivalent *hluboké učení*.

Poděkování

Nejprve bych rád poděkoval komunitě *Keras* za to, že umožnila vznik této knihy. Za posledních šest let se *Keras* rozrostl na stovky přispěvatelů do open source a více než milion uživatelů. Díky vašim příspěvkům a zpětné vazbě se *Keras* stal tím, čím je dnes.

Z osobních důvodů bych rád poděkoval své ženě za její nekonečnou podporu při vývoji frameworku *Keras* a psaní této knihy.

Rád bych také poděkoval společnosti *Google* za podporu projektu *Keras*. Bylo fantastické, že *Keras* byl přijat jako vysokoúrovňové API frameworku *TensorFlow*. Hladká integrace mezi frameworky *Keras* a *TensorFlow* je velmi přínosná jak pro uživatele *TensorFlow*, tak pro uživatele *Keras*, a zpřístupňuje hluboké učení většině.

Chtěl bych rovněž poděkovat lidem z nakladatelství *Manning*, kteří se zasloužili o vznik této knihy: nakladateli Marjanu Baceovi a všem členům redakčního a produkčního týmu včetně Michaela Stephense, Jennifer Stoutové, Aleksandara Dragosavljeviče a mnoha dalších, kteří pracovali v zákulisí.

Velký dík patří technickým recenzentům: Billy O'Callaghan, Christian Weisstanner, Conrad Taylor, Daniela Zapata Riesco, David Jacobs, Edmon Begoli, Edmund Ronald, Ph.D., Hao Liu, Jared Duncan, Kee Nam, Ken Fricklas, Kjell Jansson, Milan Šarenac, Nguyen Cao, Nikos Kanakaris, Oliver Korten, Raushan Jha, Sayak Paul, Sergio Govoni, Shashank Polasa, Todd Cook a Viton Vitanis – a všem dalším, kteří nám poslali zpětnou vazbu k návrhu knihy.

Po technické stránce patří zvláštní poděkování Frances Buontempo, technické redaktorce knihy, a Karstenu Strøbækovi, který se postaral o korektury.

O knize

Tato kniha byla napsána pro každého, kdo by chtěl zkoumat hluboké učení od začátku nebo rozšířit své chápání hlubokého učení. Ať už jste praktický inženýr pracující na strojovém učení, vývojář softwaru nebo vysokoškolský student, najdete na těchto stránkách cenné informace.

Hluboké učení prozkoumáte přístupným způsobem – začnete jednoduše a pak se dopracujete k nejmodernějším technikám. Zjistíte, že výklad je rovnoměrně rozdělen mezi intuitivní témata, teorií a praktické postupy. Vyhýbá se matematickému zápisu a místo toho raději vysvětluje základní myšlenky strojového učení a hlubokého učení prostřednictvím podrobných ukázek kódu a intuitivních mentálních modelů. Učit se budete na bohatých příkladech kódu, které obsahují rozsáhlé komentáře, praktická doporučení a jednoduchá vysvětlení na vysoké úrovni všeho, co potřebujete vědět, abyste mohli začít používat hluboké učení k řešení konkrétních problémů.

Příklady kódu využívají framework *Keras* pro hluboké učení v jazyce *Python*, jehož numerickým „procesorem" je *TensorFlow 2*. Obsahují moderní osvědčené postupy *Keras* a *TensorFlow 2* z roku 2021.

Po přečtení této knihy budete dobře rozumět tomu, co je hluboké učení, kdy je použitelné a jaká jsou jeho omezení. Budete znát standardní pracovní postupy pro přístup k problémům strojového učení a jejich řešení a budete vědět, jak řešit běžně se vyskytující problémy. Budete schopni používat *Keras* k řešení reálných problémů od počítačového vidění po zpracování přirozeného jazyka: klasifikace obrázků, segmentace obrázků, předpovídání časových řad, klasifikace textů, strojový překlad, generování textů a dalších oblastí.

Komu je kniha určena

Kniha je určena lidem se zkušenostmi s programováním v jazyce Python, kteří chtějí začít se strojovým učením a hlubokým učením. Může však být cenná i pro mnoho dalších čtenářů:

- Jste-li datový odborník pracující se strojovým učením, poskytne vám pevný a praktický úvod do hlubokého učení, nejrychleji se rozvíjející a nejvýznamnější podskupiny strojového učení.

- Jste-li odborník na hluboké učení, který by chtěl začít s frameworkem *Keras*, nabídne vám ten nejlepší rychlokurz.
- Jste-li vysokoškolský student studující hluboké učení ve formálním prostředí, využijete tuto knihu jako praktický doplněk k výuce, který vám pomáhá poznat a pochopit chování hlubokých neuronových sítí, a seznámíte se s klíčovými osvědčenými postupy.

Kniha bude užitečná i jako úvod do základních i pokročilých konceptů hlubokého učení pro technicky zaměřené lidi, kteří pravidelně nekódují.

K pochopení příkladů kódu budete potřebovat přiměřenou znalost jazyka *Python*. Kromě toho se vám bude hodit znalost knihovny *NumPy*, i když není nutná. Nepotřebujete předchozí zkušenosti se strojovým učením nebo hlubokým učením. Kniha dostatečně pokrývá všechny potřebné základy. Nepotřebujete ani pokročilé matematické znalosti – pro práci s knihou by měla stačit středoškolská matematika.

O kódu

Kniha obsahuje mnoho příkladů zdrojového kódu jak v číslovaných výpisech, tak v řádcích s běžným textem. V obou případech je zdrojový kód naformátován `písmem s pevnou šířkou`, aby se odlišil od běžného textu.

V řadě případů byl původní zdrojový kód přeformátován; přidali jsme zalomení řádků a upravili odsazení tak, aby se přizpůsobilo prostoru, který je v knize k dispozici. Protože je kód popsán v textu, byly z výpisů odstraněny komentáře. Na druhou stranu je řada výpisů doplněna poznámkami ke kódu, které upozorňují na důležité koncepty.

Poznámka překladatele:
Autor sice v doprovodných programech komentáře nemá a označuje místo toho klíčové pasáže kódu různými zalomenými šipkami s popisky, ale to si může dovolit v anglickém vydání, kde se náklady na toto pracné řešení rozpočítají mezi množství prodaných výtisků. V českém překladu jsem proto plovoucí šipky zrušil a jejich popisky vložil do kódu jako komentáře umístěné před příkaz označený v originále šipkou.

Všechny doprovodné příklady kódu uvedené v této knize jsou k dispozici na webové stránce https://www.manning.com/books/deep-learning-with-python-secondedition a jako notebooky *Jupyter* na *GitHubu* https://github.com/fchollet/deep-learning-with-python-notebooks. Lze je spustit přímo v prohlížeči prostřednictvím *Google Collaboratory*, hostovaného prostředí pro notebooky *Jupyter*, které můžete používat zdarma. K tomu, abyste mohli začít s hlubokým učením, vám tedy stačí připojení k internetu a webový prohlížeč na stolním počítači.

Diskusní fórum liveBook

Zakoupení knihy *Deep Learning with Python, second edition*, zahrnuje bezplatný přístup do soukromého webového fóra provozovaného *Manning Publications*, kde můžete knihu komentovat, pokládat technické dotazy a získat pomoc od autora a ostatních uživatelů. Přístup najdete na https://livebook.manning.com/#!/book/deeplearning-with-python-second-edition/discussion. Více informací o fóru nakladatelství *Manning* a pravidlech chování najdete také na adrese https://livebook.manning.com/#!/discussion.

Závazek nakladatelství *Manning* vůči čtenářům spočívá v tom, že poskytujeme prostor pro smysluplný dialog mezi jednotlivými čtenáři a mezi čtenáři a autorem. Nejedná se o závazek k nějaké konkrétní míře účasti ze strany autora, jehož příspěvek do fóra zůstává dobrovolný (a neplacený). Doporučujeme vám, abyste zkusili autorovi položit nějaké skutečně podnětné otázky, aby vzbudily jeho zájem! Fórum a archiv předchozích diskusí budou přístupné z webových stránek nakladatelství po celou dobu, po níž bude kniha k dispozici.

Použité typografické konvence

K tomu, abyste se v textu lépe vyznali a také abyste si vykládanou látku lépe zapamatovali, používáme několik prostředků pro odlišení a zvýraznění textu.

Termíny První výskyt nějakého termínu a další texty, které je třeba zvýraznit, vysazujeme **tučně**.

Název Názvy firem a jejích produktů vysazujeme *kurzivou*. Kurzivou budou také názvy kapitol, podkapitol a oddílů, na které v textu odkazujeme.

Citace Texty, které si můžete přečíst na displeji, například názvy polí v dialogových oknech či názvy příkazů v nabídkách, uvedeme tučným **bezpatkovým písmem**.

`Program` Identifikátory a další části programů zmíněné v běžném textu budou `neproporcionálním písmem`, které je v elektronických verzích pro zvýraznění tmavě červené.

Kromě výše zmíněných částí textu, které považujeme za důležité zvýraznit nebo alespoň odlišit od okolního textu, najdete v textu ještě řadu doplňujících poznámek a vysvětlivek. Všechny budou v jednotném rámečku, který bude označen ikonou charakterizující druh informace, kterou vám chce poznámka či vysvětlivka předat.

Symbol jin-jang bude uvozovat poznámky, s nimiž se setkáte na počátku každé kapitoly. Zde vám vždy prozradíme, co se v dané kapitole naučíte.

Obrázek knihy označuje poznámku týkající se používané terminologie. Tato poznámka většinou upozorňuje na další používané termíny označující stejnou skutečnost nebo na konvence, které se k probírané problematice vztahují.

Píšící ruka označuje obyčejnou poznámku, která pouze doplňuje informace z hlavního výkladu o nějakou zajímavost.

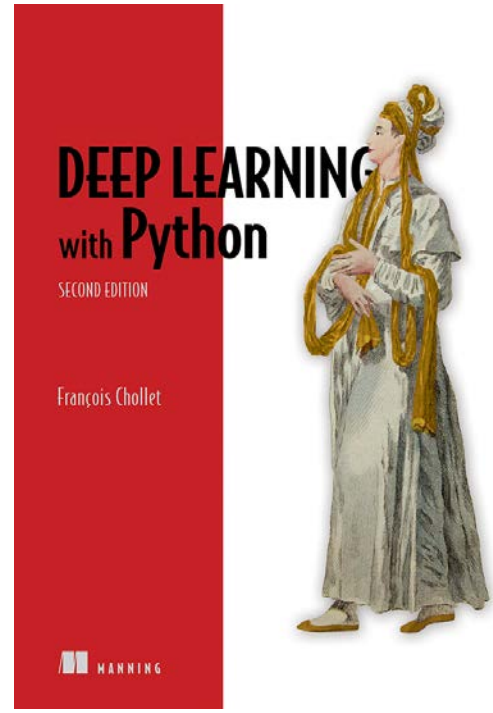
Odbočka – podšeděný blok

Občas je potřeba vysvětlit něco, co nezapadá přímo do okolního textu. V takových případech používáme podšeděný blok se silnou čarou po straně. Tento podšeděný blok představuje drobnou odbočku od ostatního výkladu. Nadpis podšeděného bloku pak najdete i v podrobném obsahu mezi nečíslovanými nadpisy.

O autorovi

François Chollet se věnuje hlubokému učení ve společnosti *Google* v kalifornském *Mountain View*. Je tvůrcem frameworku pro hluboké učení *Keras*, stejně jako přispěvatelem do frameworku strojového učení *TenzorFlow*. Dále provádí výzkum hlubokého učení se zaměřením na počítačové vidění a aplikaci strojového učení na formální uvažování. Jeho práce byly publikovány na významných konferencích v této oblasti, jako je Konference o počítačovém vidění a rozpoznávání vzorů (*Conference on Computer Vision and Pattern Recognition* – CVPR), Konference a workshop o systémech neurálního zpracování informací (*Conference and Workshop on Neural Information Processing Systems* – NIPS), Mezinárodní konference o učení se reprezentace (*International Conference on Learning Representations* – ICLR) a další.

O obálce



Následující text se vztahuje k obálce, která byla použita nakladatelstvím *Manning* u originální verze knihy. U českého vydání jsme v zájmu zachování koncepce obálek ediční řady Myslíme v…, do níž jsme knihu zařadili, původní obálku nepoužili. Nakladatelství *Manning* zobrazuje na obálkách svých knih nejrůznější dobové úbory a seznamuje čtenáře s jejich historickým a zeměpisným kontextem. Zde je tedy originální obálka včetně překladu doprovodného textu:

Na obálce knihy je vyobrazení s názvem „Oděv perské dámy v roce 1568“. Ilustrace je převzata z čtyřdílné knihy *A Collection of the Dresses of Different Nations, Ancient and Modern* (Sbírka oděvů různých národů, starověkých a moderních), od Thomase Jefferyse, vydané v Londýně mezi lety 1757 a 1772. Na titulní stránce je uvedeno, že jde o ručně kolorovanou mědirytinu, zvýrazněnou arabskou gumou.

Thomas Jefferys (1719–1771), známý jako „geograf krále Jiřího III.“, byl anglický kartograf, který byl jeho předním dodavatelem map. Vytvářel a tiskl mapy pro vládní a jiné úřední subjekty, je autorem široké škály komerčních map a atlasů, zejména Severní Ameriky. Jeho kartografická práce u něj vyvolala zájem o zvyklosti odívání v zemích, jež zkoumal a mapoval a které jsou v této sbírce skvěle zobrazeny. Fascinace vzdálenými zeměmi a cestování za potěšením představovaly koncem 18. století poměrně nové jevy. Sbírky jako tato byly populární a představovaly obyvatele jiných zemí aktivním turistům i těm, kteří se o ně zajímali z pohodlí svého křesla.

Rozmanitost kreseb v Jefferysových svazcích živě vypovídá o jedinečnosti a individualitě obyvatel naší planety před 200 lety. Oblečení se od té doby změnilo a rozmanitost podle regionů a zemí, dříve tak bohatá, zmizela. Nyní je často těžké odlišit

obyvatele jednoho kontinentu od druhého. Snad jsme, viděno optimisticky, vyměnili kulturní a vizuální rozmanitost za různorodější osobní život, nebo za rozmanitější a zajímavější intelektuální a technické možnosti.

V době, kdy je obtížné odlišit jednu počítačovou knihu od druhé, oslavuje nakladatelství *Manning* inovativnost a iniciativu oboru výpočetní techniky prostřednictvím obalů knih založených na bohaté rozmanitosti regionálního života před dvěma stoletími, kterou Jefferysovy obrazy nyní znovu přivádějí k životu.

Kapitola 1
Co je hluboké učení

Tato kapitola probírá:

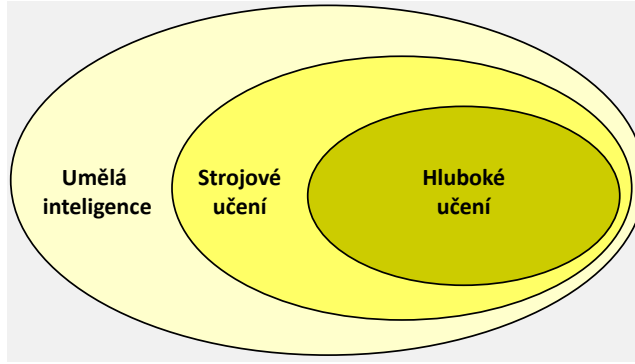
- Vysokoúrovňové definice základních konceptů.
- Časovou osu vývoje strojového učení.
- Klíčové faktory pro rostoucí popularitu a budoucí potenciál hlubokého učení.

V uplynulých letech byla umělá inteligence (AI) předmětem intenzivního mediálního humbuku. Strojové učení, hluboké učení a AI jsou tématem nesčetných článků, často mimo odborné publikace. Slibují budoucnost plnou inteligentních chatbotů, autodiagnostických vozidel a virtuálních asistentů – budoucnost někdy vykreslenou černě a jindy utopickou, kde lidská práce bude vzácná a o největší část ekonomických aktivit se postarají roboti nebo AI agenti. Pro budoucího nebo současného odborníka na strojové učení je důležité, aby byl v šumu rozpoznán signál, abyste mohli říci, že se vývoj světa oproti přehnaným tiskovým zprávám mění. Naše budoucnost je v sázce a je to budoucnost, v níž hrajete aktivní roli: po přečtení této knihy budete jedním z těch, kteří rozvíjejí agenty AI. Takže bychom si měli odpovědět na otázky: Čeho hluboké učení doposud dosáhlo? Jak je to významné? Kam máme nyní směřovat? Máme věřit tomu, že je to humbuk?

Tato kapitola vám poskytne základní informace o umělé inteligenci, strojovém učení a hlubokém učení.

1.1 Umělá inteligence, strojové učení a hluboké učení

Nejprve musíme jasně definovat, o čem mluvíme, když zmíníme AI (Artificial Intelligence – umělá inteligence). Co je umělá inteligence, strojové učení a hluboké učení (viz obrázek 1.1)? Jaký je jejich vzájemný vztah?



Obrázek 1.1:
Umělá inteligence, strojové učení a hluboké učení

1.1.1 Umělá inteligence (artificial intelligence)

Umělá inteligence (artificial intelligence AI) se zrodila v padesátých letech, kdy se hrstka průkopníků z rozvíjejícího se oboru počítačové vědy začala ptát, zda můžeme přimět počítače, aby „myslely" – je to otázka, kterou jsme ještě nezodpověděli. Stručná definice pole by byla následující: *pokusit se automatizovat intelektuální úlohy, které normálně provádějí lidé.*

Ačkoli se mnohé základní myšlenky rodily již v předchozích letech či dokonce desetiletích, „umělá inteligence" konečně vykrystalizovala jako oblast výzkumu v roce 1956, kdy John McCarthy, tehdy mladý docent matematiky na *Dartmouth College*, uspořádal letní workshop s následujícím návrhem:

> *Studie má vycházet z domněnky, že každý aspekt učení nebo jiné vlastnosti inteligence lze v zásadě popsat tak přesně, že jej lze simulovat pomocí stroje. Bude se snažit zjistit, jak přimět stroje, aby používaly jazyk, vytvářely abstrakce a pojmy, řešily druhy problémů, které jsou nyní vyhrazeny lidem, a zdokonalovaly se. Domníváme se, že v jednom nebo více z těchto problémů lze dosáhnout významného pokroku, pokud na něm bude pečlivě vybraná skupina vědců pracovat společně po celé léto.*

Na konci léta workshop skončil, aniž by se podařilo zcela vyřešit hádanku, kterou si předsevzal prozkoumat. Přesto se ho zúčastnilo mnoho lidí, kteří se stali v této oblasti průkopníky, a odstartoval intelektuální revoluci, jež trvá dodnes.

Zjednodušeně lze umělou inteligenci popsat jako snahu o automatizaci intelektuálních úkolů, které obvykle vykonávají lidé. Umělá inteligence jako taková je obecný obor, který zahrnuje strojové učení a hluboké učení, ale také mnoho dalších přístupů, které nemusí zahrnovat žádné učení. Vezměte si, že až do 80. let 20. století se ve většině učebnic AI slovo „učení" vůbec neobjevilo.

Například první šachové programy zahrnovaly pouze pevně zakódovaná pravidla vytvořená programátory a nedaly se považovat za strojové učení. Ve skutečnosti se

většina odborníků poměrně dlouhou dobu domnívala, že umělé inteligence na lidské úrovni lze dosáhnout tím, že programátoři ručně vytvoří dostatečně velký soubor explicitních pravidel pro manipulaci se znalostmi uloženými v explicitních databázích. Tento přístup je známý jako symbolická umělá inteligence. Byl dominantním paradigmatem v umělé inteligenci od 50. do konce 80. let 20. století a své největší popularity dosáhl v době rozmachu expertních systémů v 80. letech 20. století.

Ačkoli se symbolická umělá inteligence ukázala jako vhodná pro řešení dobře definovaných logických problémů, jako je například hra v šachy, ukázalo se, že je obtížné stanovit explicitní pravidla pro řešení složitějších fuzzy problémů, jako je klasifikace obrazu, rozpoznávání řeči nebo překlad přirozeného jazyka. Na místo symbolické AI nastoupil nový přístup: strojové učení.

1.1.2 Strojové učení

Ve viktoriánské Anglii byla lady Ada Lovelace přítelkyní a spolupracovnicí Charlese Babbage, vynálezce analytického stroje (*Analytical Engine*), prvního známého univerzálního počítače (tehdy ještě mechanického). Ačkoli byl tento stroj vizionářský a daleko předběhl svoji dobu (byl navržen v třicátých a čtyřicátých létech 19. století), nebyl navržen jako univerzální počítač, protože pojem obecného výpočtu ještě neexistoval. Byl pouze navržen k využití mechanických operací při automatizaci některých výpočtů z oblasti matematické analýzy – proto název *analytický stroj*.

Jako takový byl intelektuálním potomkem dřívějších pokusů o zakódování matematických operací do podoby ozubeného kola, jako byl *Pascaline* nebo Leibnizův krokový počítací stroj, zdokonalená verze *Pascaline*. *Pascaline*, který navrhl Blaise Pascal v roce 1642 (ve svých 19 letech!), byl první mechanickou kalkulačkou na světě – uměl sčítat, odčítat, násobit nebo dokonce dělit číslice.

V roce 1843 Ada Lovelace k vynálezu analytického stroje poznamenala:

> *„Analytický stroj nemá žádnou snahu cokoli vytvořit. Může dělat pouze to, u čeho víme, jak nařídit, aby to fungovalo... Jeho doménou je pomoci zpřístupnit to, s čím jsme se již seznámili.“*

Postřeh lady Lovelace je i s odstupem 178 let stále působivý. Mohl by univerzální počítač něco „vytvořit“, nebo by byl vždy vázán na tupé vykonávání procesů, kterým my lidé plně rozumíme? Mohl by být vůbec někdy schopen nějaké originální myšlenky? Mohl by se učit ze zkušeností? Mohl by projevit kreativitu?

Průkopník umělé inteligence Alan Turing ve svém přelomovém článku „*Computing Machinery and Intelligence*“[1] z roku 1950, který představil Turingův test a klíčové

[1] A. M. Turing, *Computing Machinery and Intelligence,* Mind 59, no. 236 (1950): s. 433–460.

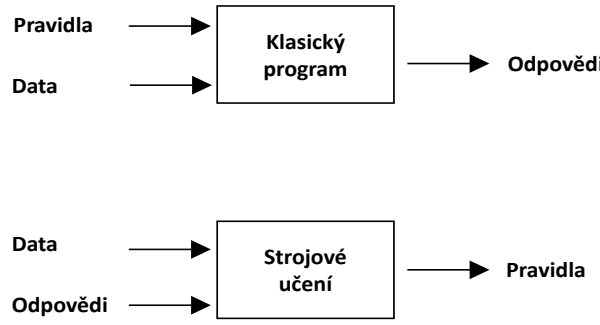
koncepty, jež později formovaly umělou inteligenci,[2] citoval její poznámku jako „cíl lady Lovelace".

Turing zastával v té době velmi provokativní názor, že počítače mohou být v zásadě schopné napodobit všechny aspekty lidské inteligence.

Obvyklým způsobem, jak přimět počítač k užitečné práci, je nechat lidského programátora napsat pravidla – počítačový program, podle kterého se mají vstupní data přeměnit na správné odpovědi, stejně jako lady Lovelace napsala instrukce, které má krok za krokem provést analytický stroj.

Strojové učení to obrací: stroj se podívá na vstupní data a odpovídající odpovědi a zjistí, jaká by měla být pravidla (viz obrázek 1.2). Systém strojového učení se spíše trénuje než explicitně programuje. Je mu předkládáno mnoho příkladů relevantních pro danou úlohu a on v těchto příkladech nachází statistickou strukturu, která nakonec systému umožní vymyslet pravidla pro automatizaci úlohy.

Pokud byste například chtěli automatizovat úlohu označování fotografií z dovolené, mohli byste systému strojového učení předložit mnoho příkladů lidmi označených fotografií a systém by se naučil statistická pravidla pro přiřazování konkrétních obrázků ke konkrétním značkám.



Obrázek 1.2:
Strojové učení jako nové programové paradigma

Ačkoli se strojové učení začalo rozvíjet teprve v devadesátých letech 20. století, tak se díky dostupnosti rychlejšího hardwaru a větších souborů dat rychle stalo nejpopulárnější a nejúspěšnější podoblastí umělé inteligence.

Strojové učení souvisí s matematickou statistikou, ale od statistiky se liší v několika důležitých ohledech obdobně, jako medicína souvisí s chemií, ale medicínu nelze redukovat na chemii, protože se zabývá vlastními odlišnými systémy s vlastními odlišnými vlastnostmi.

Na rozdíl od statistiky má strojové učení tendenci zabývat se velkými, komplexními soubory dat (jako je soubor dat o milionech obrázků, z nichž každý se skládá z desítek

[2] Ačkoli je Turingův test někdy interpretován jako doslovný test – cíl, kterého by měl obor umělé inteligence dosáhnout, Turing ho považoval pouze za konceptuální prostředek ve filozofické diskusi o povaze poznání.

tisíc pixelů), pro které by klasická statistická analýza, jakou je Bayesova analýza, byla nepraktická. V důsledku toho strojové učení (a zejména hluboké učení) vykazuje poměrně málo matematické teorie – možná až příliš málo – a je v podstatě inženýrskou disciplínou.

Na rozdíl od teoretické fyziky nebo matematiky je strojové učení velmi praktický obor, který se řídí empirickými poznatky a je hluboce závislý na pokroku v oblasti softwaru a hardwaru.

1.1.3 Učení se pravidel a reprezentaci z dat

Abychom definovali hluboké učení a pochopili rozdíl mezi hlubokým učením a jinými přístupy k strojovému učení, potřebujeme mít nejprve představu o tom, co dělají algoritmy strojového učení. Právě jsem uvedl, že strojové učení zjistí pravidla pro řešení úlohy zpracování dat, a to pomocí příkladů toho, co se očekává. K tomu, abychom stroj přiměli se učit, potřebujeme tři věci:

- *Vstupní datové body* – Například je-li úlohou rozpoznávání řeči, mohou být těmito datovými body zvukové soubory lidí, kteří mluví. Je-li úlohou označování obrázků, budou jimi obrázky.
- *Příklady očekávaného výstupu* – V úloze rozpoznávání řeči by to mohly být lidmi pořízené přepisy zvukových souborů. Při označování obrázků by očekávanými výstupy mohla být označení jako „pes“, „kočka“ a tak dále.
- *Způsob měření, zda algoritmus dělá dobrou práci* – To je nezbytné pro určení vzdálenosti mezi aktuálním a očekávaným výstupem algoritmu. Měření se používá jako zpětnovazební signál pro úpravu způsobu fungování algoritmu. Toto nastavení označujeme jako učení.

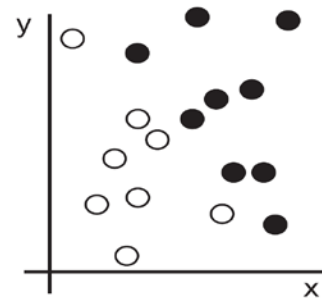
Model strojového učení přeměňuje své vstupní údaje na smysluplné výstupy, což je proces, který se „naučil“ při expozici známých příkladů vstupů a výstupů. Proto je hlavním problémem strojového učení a hlubokého učení *smysluplná transformace dat*: jinými slovy, naučit se užitečné *reprezentace* (*representation*) dostupných vstupních dat, které nás přibližují očekávanému výstupu.

Než půjdeme dále, ujasněme si, co je to reprezentace? V jádru je to jiný způsob, jak nahlížet na data – reprezentovat nebo kódovat data. Například barevný obraz může být zakódován ve formátu RGB (červená-zelená-modrá) nebo ve formátu HSV (odstín-sytost-hodnota): jedná se o dvě různé reprezentace stejných dat.

Některé úlohy, které mohou být u jedné reprezentace obtížné, mohou být u jiných snadné. Například úloha „vybrat všechny červené obrazové body v obraze“ je jednodušší ve formátu RGB, zatímco úloha „snížit sytost obrazu“ je jednodušší ve formátu HSV. Modely strojového učení se zaměřují na hledání vhodných reprezentací pro svá vstupní data – ruční transformace dat do podoby, která je přívětivější pro konkrétní klasifikaci dat.

Pojďme úlohu konkretizovat. Uvažujme osy x a y a body reprezentované souřadnicemi v systému (x, y), jak je znázorněno na obrázku 1.3.

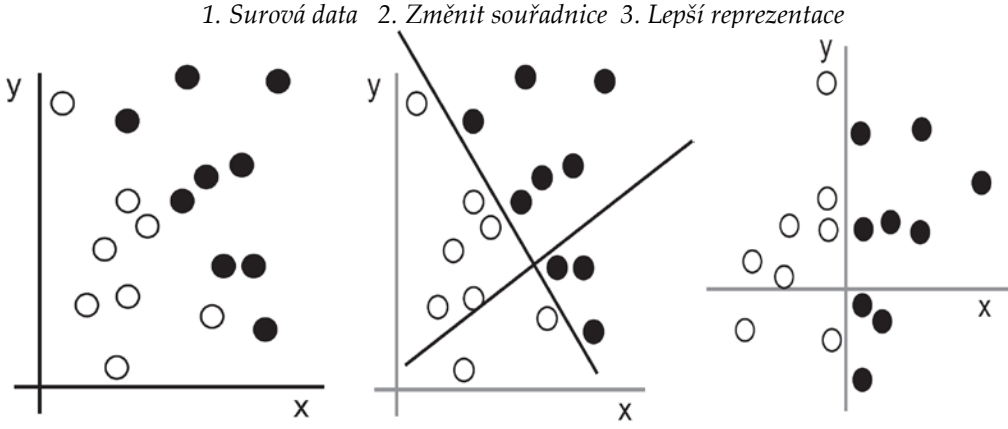
Jak vidíte, máme pár bílých bodů a několik černých bodů. Řekněme, že chceme vytvořit algoritmus, který může obdržet souřadnice (x, y) bodu a vrátit informaci, zda je tento bod pravděpodobně černý nebo bílý. V tomto případě:



Obrázek 1.3:
Malý vzorek dat

- Vstupy jsou souřadnice našich bodů.
- Očekávané výstupy jsou barvy našich bodů.
- Mírou toho, zda náš algoritmus dělá dobrou práci, může být například procentní podíl bodů, které jsou správně klasifikovány.

Potřebujeme reprezentaci našich údajů, jež jasně oddělí bílé body od černých. Jedna transformace, kterou bychom mohli využít (mezi mnoha jinými možnostmi), by byla změna souřadnic znázorněná na obrázku 1.4.



Obrázek 1.4:
Změna souřadnic

V tomto novém souřadnicovém systému mohou být souřadnice našich bodů považovány za nové zobrazení našich dat. A je to dobré! S touto reprezentací lze černobílý problém klasifikace vyjádřit jako jednoduché pravidlo: „černé body jsou takové, že `x > 0`“, nebo „bílé body jsou takové, že `x < 0`“. Toto nové zobrazení v zásadě řeší problém klasifikace.

V tomto případě jsme změnu souřadnic definovali ručně: použili jsme naši lidskou inteligenci, abychom vymysleli vlastní vhodnou reprezentaci dat. Je to v pořádku pro takto extrémně jednoduchý problém, ale mohli byste postupovat stejně, kdyby úloha spočívala v klasifikaci obrázků ručně psaných číslic? Dokázali byste zapsat explicitní, počítačem proveditelné transformace obrazu, které by osvětlily rozdíl mezi 6 a 8, mezi 1 a 7, a to ve všech druzích různých rukopisů?

To je do jisté míry možné. Pravidla založená na reprezentaci číslic, jako je „počet uzavřených smyček“ nebo vertikální a horizontální histogramy pixelů, dokážou slušně rozlišit ručně psané číslice. Ale ruční hledání takových užitečných reprezentací je náročné, a jak si dokážete představit, výsledný systém založený na pravidlech je křehký – jeho údržba představuje noční můru.

Pokaždé, když narazíte na nový příklad rukopisu, který porušuje vaše pečlivě promyšlená pravidla, budete muset přidat nové transformace dat a nová pravidla, přičemž musíte zohlednit jejich interakci s každým předchozím pravidlem.

Pravděpodobně si říkáte, že když je tento proces tak bolestivý, mohli bychom ho zautomatizovat? Co kdybychom zkusili systematicky vyhledávat různé sady automaticky generovaných reprezentací dat a na nich založených pravidel, přičemž bychom jako zpětnou vazbu použili procenta správně klasifikovaných číslic v nějaké vývojové sadě dat? Pak bychom prováděli strojové učení.

Učení v kontextu strojového učení popisuje proces automatického hledání transformací dat, které vytvářejí užitečné reprezentace nějakých dat, řízené nějakými zpětnými vazbami – reprezentacemi signálů, které jsou vhodné pro jednodušší pravidla řešící danou úlohu.

Těmito transformacemi mohou být změny souřadnic (jako v našem příkladu klasifikace 2D souřadnic) nebo pořízení histogramu pixelů a počítání smyček (jako v našem příkladu klasifikace číslic), ale mohou to být také lineární projekce, translace, nelineární operace (například „vyber všechny body tak, že `x > 0`“) atd.

Algoritmy strojového učení obvykle nejsou při hledání těchto transformací kreativní, ale pouze prohledávají předem definovanou množinu operací – tzv. *prostor hypotéz*. Například v příkladu klasifikace 2D souřadnic byla naším prostorem hypotéz množina všech možných změn souřadnic.

Technicky je tedy strojové učení *hledání užitečných reprezentací některých vstupních dat v předem definovaném prostoru možností pomocí pokynů ze zpětnovazebního signálu*. Tato jednoduchá myšlenka umožňuje řešit pozoruhodně širokou škálu intelektuálních úloh, od rozpoznávání řeči po autonomní řízení vozidel.

Nyní, když už víte, co máme na mysli pod pojmem učení, se podívejme na to, co dělá hluboké učení zvláštním.

1.1.4 Hloubka v hlubokém učení

Hluboké učení je specifickou podskupinou strojového učení: nový přístup k učení se reprezentací z dat, který klade důraz na učení se následujících vrstev stále smysluplnějších reprezentací. *Hloubka (deep)* v *hlubokém učení* není odkazem na jakýkoli druh hlubšího porozumění dosaženého tímto přístupem; spíše to znamená myšlenku postupných vrstev reprezentací.

Počet vrstev přispívajících k modelu dat se nazývá *hloubka modelu*. Jiné vhodné názvy pro pole by mohly být *učení* se *vrstvených reprezentací* (*layered representations learning*) nebo *učení* se *hierarchických reprezentací* (*hierarchical representations learning*).

Moderní hluboké učení často zahrnuje desítky nebo dokonce stovky po sobě jdoucích vrstev reprezentací – a všechny se automaticky učí při trénování. Mezitím se jiné přístupy k strojovému učení zaměřují na učení pouze jedné nebo dvou vrstev reprezentací dat (řekněme, že vezmou histogram pixelů a pak použijí klasifikační pravidlo) – proto se někdy označují jako *mělké učení* (*shallow learning*).

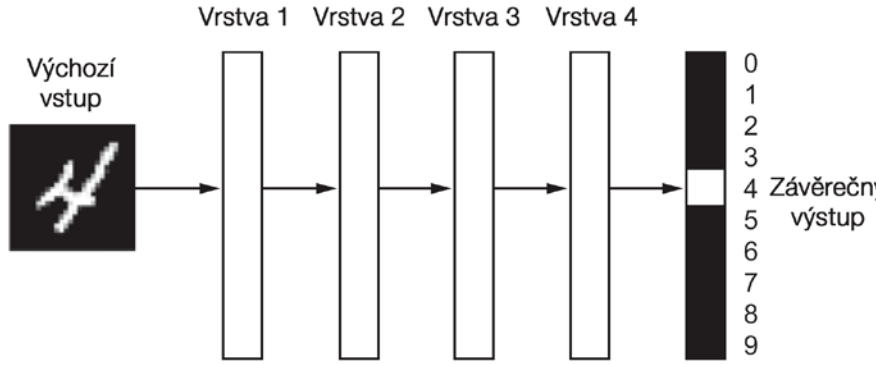
V hlubokém učení se tyto vrstvené reprezentace učí prostřednictvím modelů nazývaných neuronové sítě, strukturované ve vrstvách na sobě (a to doslova).

Termín neuronová síť je odkazem na neurobiologii, ale přestože některé z ústředních konceptů v hlubokém učení byly částečně inspirovány naším chápáním mozku (zejména zrakové kůry), modely hlubokého učení nejsou modely mozku. Neexistuje žádný důkaz o tom, že mozek implementuje něco jako učící se mechanismy používané v moderních modelech hlubokého učení.

Můžete se setkat s populárně vědeckými články, které prohlašují, že hluboké učení funguje jako mozek nebo je modelováno podle mozku, ale tak to není.

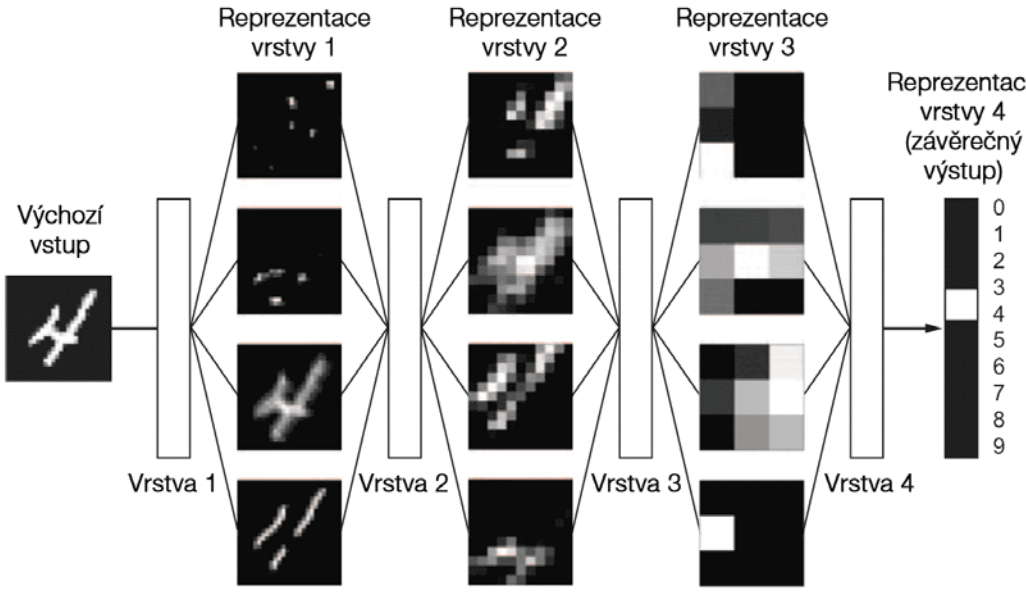
Pro nováčky v oboru by bylo matoucí a kontraproduktivní, aby v hlubokém učení hledali jakoukoli souvislost s neurobiologií; takovouto berličku mystiky a tajemství nepotřebujete. Můžete zapomenout na vše, co jste si přečetli o hypotetických vazbách mezi hlubokým učením a biologií. Pro naše účely je hluboké učení matematickým rámcem pro učení reprezentací z dat.

Jak tedy vypadají reprezentace vzniklé na základě algoritmů hlubokého učení? Podívejme se, jak síť s několika vrstvami (viz obrázek 1.5) transformuje obraz číslice, aby rozpoznal, jaká je to číslice.



Obrázek 1.5:
Hluboká neuronová síť pro klasifikaci číslic

Jak vidíte na obrázku 1.6, síť transformuje číslicový obrázek na reprezentace, které se stále více liší od původního obrazu a stále více informují o výsledném výsledku. Představte si hlubokou síť jako vícestupňovou operaci destilace informací, kde informace procházejí po sobě následujícími filtry a vyjdou pokaždé o něco lépe vyčištěné (řešíte-li nějakou úlohu, je to skutečně užitečné).



Obrázek 1.6:
Hluboké reprezentace naučené modelem pro klasifikaci číslic

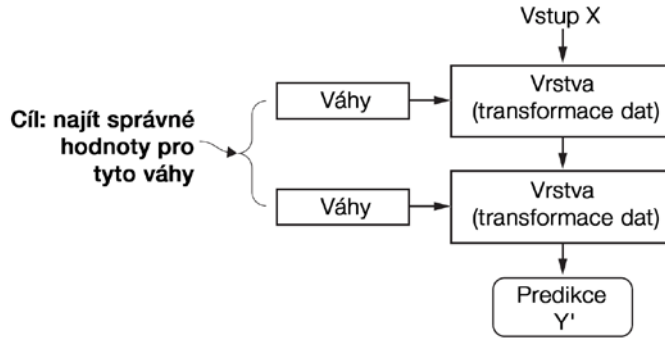
Takže právě tohle je hluboké učení z technického hlediska: vícestupňový způsob, jak se naučit datové reprezentace. Je to jednoduchý nápad, ale jak se ukázalo, i velmi jednoduché mechanismy, jsou-li dostatečně dimenzované, mohou nakonec vypadat jako magie.

1.1.5 Pochopení toho, jak hluboké učení funguje, ve třech krocích

Nyní tedy víte, že strojové učení je zaměřené na mapování vstupů (například obrazů) na cíle (například označení „kočka“), což se provádí pozorováním mnoha příkladů vstupů a odpovídajících cílů. Také víte, že hluboké neuronové sítě dělají toto mapování vstupů na cíle prostřednictvím hluboké posloupnosti jednoduchých transformací dat (vrstev) a že se tyto transformace dat naučí načtením vyřešených příkladů. Nyní se podívejme, jak se toto učení konkrétně děje.

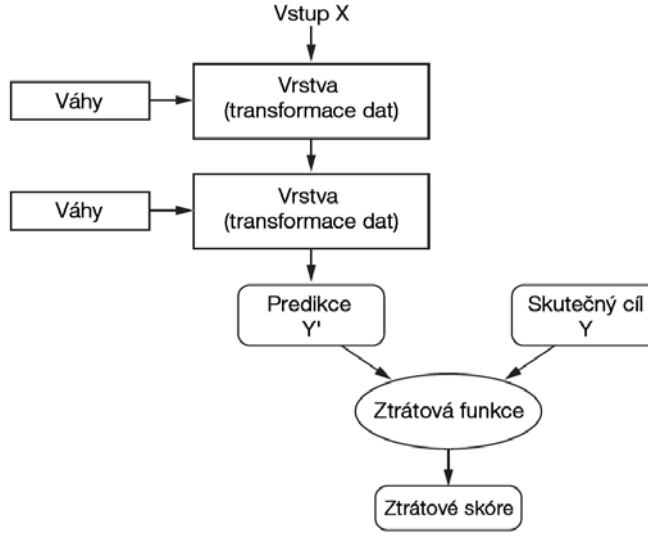
Specifikace toho, co vrstva dělá se svými vstupními daty, je uložena ve vahách vrstvy, což je v podstatě spousta čísel. Z technického hlediska bychom řekli, že transformace realizovaná vrstvou je parametrizována podle jejích vah (viz obrázek 1.7). (Váhy se někdy označují jako parametry vrstvy.)

V tomto kontextu učení znamená zjišťování množiny hodnot pro váhy všech vrstev v síti tak, aby síť správně mapovala příklady vstupů na přidružené cíle. Ale tady je problém: hluboká neuronová síť může obsahovat desítky milionů parametrů. Nalezení správné hodnoty pro všechny z nich může vypadat jako náročná úloha, obzvláště vzhledem k tomu, že změna hodnoty jednoho parametru ovlivní chování všech ostatních!



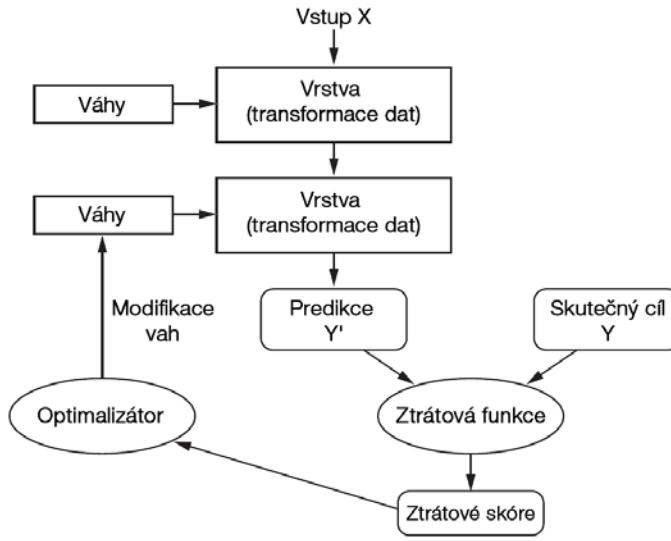
Obrázek 1.7:
Neuronová síť je parametrizována svými vahami

Chcete-li něco řídit, musíte to být schopni sledovat. Chcete-li řídit výstup neuronové sítě, musíte být schopni měřit, jak daleko je tento výstup od toho, co jste očekávali. To je úloha *ztrátové funkce* (*loss function*) sítě, někdy nazývané také *cílová funkce* (*objective function*) nebo *nákladová funkce* (*cost function*). Ztrátová funkce přebírá predikce sítě a skutečný cíl (to, co chcete z výstupu sítě) a vypočítá *ztrátové skóre* (*loss score*) vzdálenosti, jež zachycuje, jak dobře si síť vedla v tomto konkrétním příkladu (viz obrázek 1.8).



Obrázek 1.8:
Ztrátová funkce měří kvalitu výstupu sítě

Základním trikem hlubokého učení je použití tohoto skóre jako zpětnovazebního signálu, podle nějž upravíme hodnoty vah ve směru, který sníží *ztrátové skóre* (*loss score*) pro daný příklad (viz obrázek 1.9). Toto nastavení je úlohou optimalizátoru, který implementuje *algoritmus zpětného šíření* (*backpropagation algorithm*), jenž je hlavním algoritmem v hlubokém učení. V následující kapitole si vysvětlíme podrobněji, jak tento algoritmus funguje.



Obrázek 1.9:
Ztrátové skóre se používá jako zpětná vazba pro nastavení vah

Zpočátku jsou vahám sítě přiřazeny náhodné hodnoty, takže síť pouze implementuje řadu náhodných transformací. Samozřejmě, jejich výstup je daleko od toho, co by měl v ideálním případě být, a ztrátové skóre je proto velmi vysoké. Ale s každým příkladem, který síť zpracovává, jsou váhy nastaveny trochu jinak (pokud možno správným směrem) a ztrátové skóre se snižuje.

Jedná se o *trénovací cyklus* (*training loop*), který se mnohokrát opakuje (obvykle desítky iterací přes tisíce příkladů). Nakonec nastaví hodnoty vah, které minimalizují ztrátovou funkci. Síť s minimální ztrátou, tedy síť, pro kterou jsou výstupy tak blízko cíle, jak jen mohou být, označujeme termínem *natrénovaná síť* (*trained network*).

Vše je založeno na jednoduchém mechanismu, který vše nastaví a skončí jako kouzlo.

1.1.6 Čeho hluboké učení dosud dosáhlo

I když je hluboké učení poměrně starou částí strojového učení, jeho význam začal růst až po roce 2010. Během několika málo let přineslo revoluci s pozoruhodnými výsledky v úlohách zaměřených na vnímání, a dokonce i na zpracování přirozeného jazyka – tedy v úlohách zahrnujících dovednosti, které se lidem zdají přirozené a intuitivní, ale strojům dlouho unikaly.

Hluboké učení dosáhlo zejména následujících průlomů v historicky obtížných oblastech strojového učení:

- Klasifikace obrazu na úrovni blízké člověku.
- Rozpoznávání řeči na téměř lidské úrovni.
- Přepis rukopisu na téměř lidské úrovni.

- Dramatické zdokonalení strojového překladu.
- Výrazné vylepšení převodu textu na řeč.
- Digitální asistenti, jako jsou *Google Assistant a Amazon Alexa.*
- Autonomní řízení na úrovni člověka.
- Vylepšené cílení reklam, jak je používá *Google*, *Baidu* a *Bing.*
- Vylepšení výsledků vyhledávání na webu.
- Schopnost odpovědět na otázky položené v přirozeném jazyce.
- Nadlidská schopnost hraní hry Go.

Stále zkoumáme plný rozsah toho, co všechno hluboké učení dokáže. S velkým úspěchem jsme ho začali aplikovat na nejrůznější problémy, které byly ještě před několika lety považovány za neřešitelné – automatický přepis desítek tisíc starověkých rukopisů uložených ve vatikánském Apoštolském archivu, detekce a klasifikace chorob rostlin na polích pomocí jednoduchého chytrého telefonu, pomoc onkologům nebo radiologům při interpretaci lékařských zobrazovacích dat, předpovídání přírodních katastrof jako povodně, hurikány nebo dokonce zemětřesení, a tak dále.

S každým dalším milníkem se blížíme době, kdy nám hluboké učení bude pomáhat při každé činnosti a ve všech oblastech lidské činnosti – ve vědě, medicíně, výrobě, energetice, dopravě, vývoji softwaru, zemědělství, a dokonce i v umělecké tvorbě.

1.1.7 Nevěřte krátkodobému humbuku

I když hluboké učení vedlo v posledních letech k dosažení pozoruhodných úspěchů, očekávání, čeho bude schopné dosáhnout v příštím desetiletí, mají tendenci být mnohem vyšší, než je reálné očekávat. Přestože některé aplikace, jakými jsou například autonomní automobily, jsou již na dosah, je pravděpodobné, že další zůstanou dlouho nepoužitelné – například věrohodné dialogové systémy, strojový překlad na úrovni člověka napříč libovolnými jazyky a porozumění přirozenému jazyku na úrovni člověka. Zejména řeči o všeobecné inteligenci na úrovni člověka by se neměly brát příliš vážně. Riziko vysokého očekávání v krátkodobém horizontu spočívá v tom, že pokud ho technologie nenaplní, investice do výzkumu vyprší a další pokrok se na dlouhou dobu zpomalí.

Dříve se to už stalo. Dvakrát v minulosti prošla AI cyklem intenzivního optimismu, následovaného zklamáním a skepticismem (a následným nedostatečným financováním). Začalo to symbolickou AI v šedesátých letech 20. století. V raných počátcích vylétla očekávání vysoko.

Jeden z nejznámějších průkopníků a zastánců symbolického přístupu byl Marvin Minsky, který v roce 1967 prohlásil: „*Během jedné generace ... bude problém umělé inteligence v podstatě vyřešen.*“ O tři roky později, v roce 1970, predikci upřesnil: „*Za tři až osm let budeme mít stroj s obecnou inteligencí průměrného člověka.*“

V roce 2021 se zdá, že takovéto úspěchy jsou hudbou daleké budoucnosti – tak daleké, že nemůžeme předvídat, jak dlouho to bude trvat – avšak v šedesátých a sedmdesátých letech minulého století několik odborníků věřilo, že úspěch klepe na dveře (což si řada lidí myslí dodnes). O několik let později, když se tato vysoká očekávání nepodařilo naplnit, se výzkumníci a vládní fondy od tohoto pole výzkumu odvrátily, což znamenalo začátek první AI zimy (odkazuji zde na jadernou zimu, protože to bylo krátce po vrcholu studené války).[3]

Nebylo to naposledy. V osmdesátých letech se mezi velkými společnostmi začala prosazovat nová vlna symbolické AI a expertních systémů. Několik počátečních příběhů o úspěchu vyvolalo vlnu investic, přičemž korporace po celém světě otevíraly vlastní oddělení AI, aby pro ně vyvinuly expertní systémy. Kolem roku 1985 podniky vynakládaly na technologii více než 1 miliardu dolarů ročně, ale počátkem devadesátých let 20. století se tyto systémy ukázaly jako nákladné, obtížně měřitelné a omezené – a zájem zmizel. Začala tak druhá zima AI.

Možná jsme v současné době svědky třetího cyklu očekávání a zklamání. AI je prozatím ve fázi intenzivního optimismu. Nejlepší je krátkodobá očekávání zklidnit a ujistit se, že lidé, kteří nejsou příliš obeznámeni s technickou stránkou oblasti, mají jasnou představu o tom, co hluboké učení může a nemůže přinést.

1.1.8 Příslib AI

Ačkoli můžeme mít od AI nereálná krátkodobá očekávání, dlouhodobý obraz vypadá jasně. Začínáme s aplikací hlubokého učení na řadu důležitých problémů, pro něž by se mohla ukázat jako přínosná, od lékařských diagnóz až po digitální asistenty.

Výzkum AI se v uplynulých pěti letech výrazně urychlil, a to z velké části díky úrovni financování, jaká se v dosavadní historii AI nikdy neprojevila. Nicméně mezi produkty a procesy, které tvoří náš svět, se tohoto pokroku dosud dostalo poměrně málo. Většina výsledků výzkumu v oblasti hlubokého učení se dosud nevyužívá, nebo se přinejmenším neuplatňuje v celé řadě problémů v nejrůznějších odvětvích, v nichž by se uplatnit mohla. Váš lékař ani váš účetní zřejmě AI stále nepoužívá.

Pravděpodobně ani vy nepoužíváte technologie AI v každodenním životě. Samozřejmě, můžete zadávat vašemu smartphonu jednoduché otázky a získat přiměřené odpovědi, můžete získat poměrně užitečné produktové doporučení na Amazon.com a můžete hledat „narozeniny“ ve službě *Google Photos* a okamžitě najít fotky narozeninové párty své dcery z posledního měsíce.

To je ale pouze vzdálený výkřik z oblasti, kde jsme si tyto technologie zvykli používat. Podobné nástroje jsou však stále jen doplňky k našemu každodennímu životu. AI musí teprve přejít do stavu, kdy bude základem toho, jak pracujeme, myslíme a žijeme.

---

[3] *Pozn. překladatele:* V současné době se odhaduje, že stroj dosáhne inteligence průměrného člověka někdy po roce 2029.

Je těžko uvěřitelné, že by AI mohla mít velký dopad na svět, protože ještě není rozšířena – stejně jako bylo v roce 1995 obtížné uvěřit v budoucí dopad internetu. V té době většina lidí ani netušila, jak pro ně bude internet důležitý a jakým způsobem se změní jejich život. Totéž platí dnes pro hluboké učení a AI. Ale AI přichází, o tom není pochyb. Ve vzdálené budoucnosti bude AI vaším asistentem, dokonce i přítelem; zodpoví vaše otázky, pomůže vzdělávat vaše děti a sledovat vaše zdraví. Dodá potraviny k vašim dveřím a povede vás z bodu A do bodu B. Bude to vaše rozhraní ke stále složitějšímu a informačně náročnějšímu světu. A ještě důležitější je, že AI pomůže lidstvu jako celku postoupit vpřed tím, že pomůže lidským vědcům v nových průlomových objevech ve všech vědních oborech od genomiky po matematiku.

Na cestě se můžeme setkat s několika neúspěchy a možná i novou zimou, jakou v letech 1998 až 1999 zažil internetový průmysl. Tehdy po přehnaných očekáváních utrpěl havárii, v jejímž důsledku na počátku tohoto století vyschly investice. Ale jednou se tam dostaneme. AI nakonec aplikujeme na téměř každý proces, který tvoří naši společnost a každodenní život, stejně jako dnes aplikujeme internet.

Nevěřte krátkodobému humbuku, ale věřte v dlouhodobou vizi. Může to chvíli trvat, než AI uplatní svůj skutečný potenciál – potenciál, o němž se zatím nikdo neodvážil ani snít. AI však přichází a fantastickým způsobem změní náš svět.

1.2 Před hlubokým učením: stručná historie strojového učení

Hluboké učení dosáhlo v historii umělé inteligence dosud nevídané úrovně pozornosti veřejnosti a investic. Není to však první úspěšná forma strojového učení. Je jisté, že většina algoritmů strojového učení, jež se dnes používají, nejsou algoritmy hlubokého učení.

Hluboké učení není vždy tím správným nástrojem pro práci – někdy není dostatek dat pro to, aby se mohlo aplikovat, někdy je problém lépe řešen jiným algoritmem. Představuje-li hluboké učení váš první kontakt se strojovým učením, můžete se ocitnout v situaci, kdy držíte v ruce jen hluboce se učící kladivo a každý problém se strojovým učením vám začne připadat jako hřebík. Jediným způsobem, jak do této pasti nespadnout, je seznámit se i s jinými přístupy a vyzkoušet je tam, kde je to vhodné.

Podrobná diskuse o klasických metodách strojového učení je mimo rámec této knihy, ale krátce je projdeme a popíšeme historický kontext, v němž byly vytvořeny. To nám umožní umístit hluboké učení do širšího kontextu strojového učení a lépe pochopit, odkud hluboké učení pochází a proč na tom záleží.

1.2.1 Pravděpodobnostní modelování

Pravděpodobnostní modelování (*probabilistic modeling*) spočívá v aplikaci principů statistiky na analýzu dat. Jedná se o jednu z nejranějších forem strojového učení a je dodnes rozšířené. Jedním z nejznámějších algoritmů v této kategorii je algoritmus *Naivní Bayes*.

Naivní Bayes (*Naive Bayes*) je typ učícího se klasifikátoru založeného na použití Bayesovy věty. Předpokládá, že *rysy* (vlastnosti, příznaky) vstupních dat jsou nezávislé (silný nebo „naivní" předpoklad – odtud pochází název).

Tato forma analýzy dat se používala před nástupem počítačů a byla aplikována ručně po desetiletí před první implementací na počítači (s největší pravděpodobností od padesátých let). Bayesova věta a základy statistiky pocházejí z osmnáctého století, a to je vše, co potřebujete vědět, abyste mohli začít klasifikační systémy typu *Naivní Bayes* používat.

Úzce související model je *logistická regrese* (zkratka *logreg*), která je někdy považována za „*Hello world*" moderního strojového učení. Nenechte se zmást názvem – spíše než regresní algoritmus je *logreg* klasifikační algoritmus. Stejně jako *Naivní Bayes*, *logreg* předběhl výpočetní techniku o řadu let, ale přesto je dnes stále užitečný díky své jednoduché a všestranné povaze. Často je to první věc, kterou se datový vědec pokusí na datové sadě vyzkoušet, aby získal představu o dané klasifikační úloze.

1.2.2 Rané neuronové sítě

Rané iterace neuronových sítí byly zcela nahrazeny moderními variantami popsanými na stránkách této knihy, ale je užitečné si uvědomit, jak hluboké učení vzniklo. Přestože základní myšlenky neuronových sítí byly ve formě her zkoumány již v padesátých letech 20. století, trvalo desítky let, než se tento přístup rozběhl.

Po dlouhou dobu chyběl způsob, jak velké neuronové sítě účinně trénovat. To se změnilo v polovině osmdesátých let, kdy několik lidí nezávisle na sobě znovu objevilo *algoritmus zpětného šíření* (*Backpropagation algorithm*) – způsob, jak trénovat posloupnosti parametrizovaných operací optimalizací *gradientního sestupu* (později v knize tyto koncepty přesně definujeme) – a začali jej používat pro neuronové sítě.

První úspěšná praktická aplikace neuronových sítí přišla v roce 1989 od společnosti *Bell Labs*, když Yann LeCun zkombinoval starší nápady konvolučních neuronových sítí a zpětné propojení a aplikoval je na problém klasifikace ručně psaných číslic. Výslednou síť nazvanou *LeNet* používala v devadesátých letech poštovní služba Spojených států k automatizaci čtení poštovních směrovacích čísel na obálkách.

1.2.3 Jádrové metody (kernel methods)

Díky tomuto prvnímu úspěchu začaly v devadesátých letech neuronové sítě získávat mezi výzkumnými pracovníky určitý respekt. Rychle se proslavil nový přístup k strojovému učení – jádrové metody, které záhy poslaly původní neuronové sítě rychle zpět do zapomnění.

Jádrové metody (*kernel methods*) jsou skupinou klasifikačních algoritmů, z nichž nejznámější je *metoda podpůrných vektorů* označovaná často jako *podpůrný vektorový stroj* (*support vector machine* – SVM). Moderní formulace SVM byla vyvinuta Vladimírem Vapnikem a Corinnou Cortesovou na počátku devadesátých let v laboratořích *Bell Labs* a publikována v roce 1995,[4] i když starší, lineární formulace byla publikována Vladimírem Vapnikem a Alexey Chervonenkisem již v roce 1963.[5]

Obrázek 1.10: *Rozhodovací hranice*

Tyto algoritmy se snaží řešit problémy s klasifikací nalezením *rozhodovacích hranic* (*decision boundaries*) mezi dvěma sadami bodů patřících do dvou různých kategorií (viz obrázek 1.10). Rozhodovací hranice může být považována za čáru nebo plochu oddělující vaše trénovací data do dvou oblastí odpovídajících dvěma kategoriím. Chcete-li klasifikovat nové datové body, stačí zkontrolovat, na které straně rozhodovací hranice jsou.

SVM postupuje při hledání těchto hranic ve dvou krocích:

1. Data jsou mapována na novou vícerozměrnou reprezentaci, v níž může být rozhodovací hranice vyjádřena jako nadrovina (pokud by data byla dvourozměrná jako na obrázku 1.10, byla by nadrovinou přímka).
2. Dobrá rozhodovací hranice (rozdělující nadrovina) se vypočítá tak, že se snažíme maximalizovat vzdálenost mezi touto nadrovinou a nejbližšími datovými body z každé třídy, krok nazvaný *maximalizace odstupu* (*maximizing the margin*). Umožňuje to, aby se hranice dobře zobecňovala na nové příklady mimo soubor trénovacích dat.

Technika mapování dat na vícerozměrnou reprezentaci, kde se klasifikace zjednodušuje, může vypadat dobře na papíře, ale v praxi je často výpočetně neřešitelná. Zde přichází *jádrový trik* (*kernel trick* – klíčová myšlenka, po níž jsou jádrové metody pojmenovány). Jeho podstatou je nalezení nové rozhodovací nadroviny v novém reprezentačním prostoru. Nemusíte explicitně počítat souřadnice bodů v novém prostoru – stačí vypočítat vzdálenost mezi dvojicemi bodů v daném prostoru, což může být provedeno efektivně pomocí *jádrových funkcí*.

Jádrová funkce je výpočetně zvládnutelná operace, která mapuje libovolnou dvojici bodů ve vašem počátečním prostoru na vzdálenost mezi těmito body v prostoru nové reprezentace. Zcela přitom vynechá explicitní výpočet nové reprezentace. Jádrové funkce jsou typicky vytvářeny ručně, nikoli naučením z dat. V případě SVM je naučena pouze rozdělující nadrovina.

V době, kdy byly vyvinuty, poskytovaly SVM v jednoduchých klasifikačních problémech špičkové výsledky a patřily k málu metod strojového učení podpořených rozsáhlou teorií, takže je bylo možné podrobit seriózní matematické analýze. Byly

[4] Vladimir Vapnik, Corinna Cortes, *Support-Vector Networks*, Machine Learning 20, no. 3 (1995): s. 273–297.

[5] Vladimir Vapnik, Alexey Chervonenkis, *A Note on One Class of Perceptrons*, Automation and Remote Control 25 (1964).

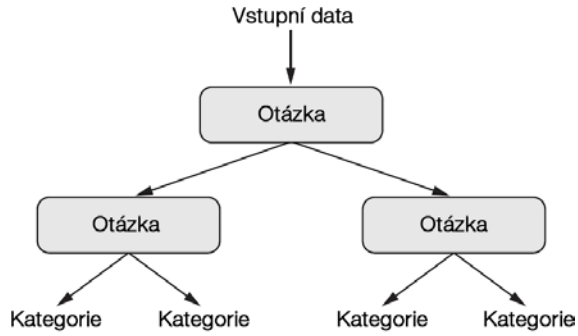
proto pochopitelné a snadno interpretovatelné. Díky těmto užitečným vlastnostem byly SVM po dlouhou dobu extrémně populární.

Ukázalo se však, že SVM jsou obtížně škálovatelné pro velké datové sady a neposkytují dobré výsledky pro problémy vnímání, jako je například klasifikace obrazů. Vzhledem k tomu, že SVM je mělká metoda, její aplikace na problémy vnímání vyžaduje nejprve ruční nalezení užitečné reprezentace (krok nazvaný *inženýrství rysů – feature engineering*), což je obtížné a křehké.

Chcete-li například použít SVM ke klasifikaci ručně psaných číslic, nemůžete začít od nezpracovaných pixelů; měli byste nejprve ručně najít užitečné reprezentace, které činí problém schůdnějším – například histogramy pixelů, o nichž jsem se zmínil dříve.

1.2.4 Rozhodovací stromy, náhodné lesy a stroje na posílení gradientu

Rozhodovací stromy (*decision trees*) jsou datové struktury, které vám umožňují klasifikovat vstupní datové body nebo predikovat výstupní hodnoty pro dané vstupy (viz obrázek 1.11). Jsou snadno vizualizovatelné a interpretovatelné. Rozhodovací stromy naučené z dat začaly od roku 2000 získávat významnou pozornost a kolem roku 2010 už byly často upřednostňovány před jádrovými metodami.



Obrázek 1.11:
Rozhodovací strom: parametry, které se naučil, jsou otázky týkající se dat. Otázkou může být například „Je koeficient 2 v datech větší než 3,5?“

Zvláště algoritmus *Náhodný les* (*Random Forest*) produkuje robustní, prakticky uchopitelné učení se rozhodovacího stromu, zahrnující budování velkého počtu specializovaných rozhodovacích stromů a následné kombinování jejich výstupů.

Náhodné lesy jsou použitelné na širokou škálu problémů – lze říci, že jsou téměř vždy druhým nejlepším algoritmem pro jakoukoli úlohu mělkého učení. Když se v roce 2010 rozběhla populární soutěžní webová stránka *Kaggle* (http://kaggle.com), získaly náhodné lesy na této platformě rychle velkou oblibu, která jim vydržela až do roku 2014, kdy otěže převzaly *stroje na posílení gradientu* (*gradient boosting machines*).

Metoda na posílení gradientu je, podobně jako náhodný les, technika strojového učení založená na kombinování slabých predikčních modelů, obecně rozhodovacích stromů. Využívá *posílení gradientu* (*gradient boosting*), což je způsob, jak iterativně zlepšit

jakýkoli strojový model natrénováním nových modelů, které se specializují na řešení slabých míst předchozích modelů.

Použití techniky posílení gradientu vede při aplikaci na rozhodovací stromy k modelům, které většinu času striktně převyšují náhodné lesy, přičemž mají podobné vlastnosti. Dnes je to jeden z nejlepších, ne-li nejlepší algoritmus pro práci s daty nereprezentujícími vnímání. Vedle hlubokého učení je to jedna z nejčastěji používaných technik v soutěžích *Kaggle*.

1.2.5 Zpět k neuronovým sítím

Okolo roku 2010 se neuronové sítě vědecké komunitě téměř zcela vyhýbaly, avšak řada lidí, kteří na nich stále pracovali, začala dělat významné objevy: skupiny Geoffreyho Hintona na univerzitě v Torontu, Yoshua Bengio na univerzitě v Montrealu, Yann LeCun z Newyorské univerzity a IDSIA ve Švýcarsku.

V roce 2011 začal Dan Ciresan ze společnosti IDSIA s hlubokými neuronovými sítěmi natrénovanými s GPU vítězit v akademických soutěžích v klasifikaci obrazů – byl to první praktický úspěch moderního hlubokého učení. Zlomový okamžik však přišel v roce 2012, kdy se Hintonova skupina zúčastnila každoroční soutěže *ImageNet* (*ImageNet Large Scale Visual Recognition Challenge*, zkráceně ILSVRC) v oblasti klasifikace obrazu ve velkém měřítku.

Tato soutěž byla v té době značně obtížná a spočívala v klasifikaci barevných obrazů s vysokým rozlišením do 1000 různých kategorií po trénování na 1,4 milionu snímků. V roce 2011 měl vítězný model založený na klasických přístupech k počítačovému vidění pouze 74,3% úspěšnost typu top-5.[6] V roce 2012 pak tým vedený Alexem Krizhevskym s poradcem Geoffrey Hintonem dosáhl top-5 správnosti 83,6 %, což byl významný průlom. V soutěži začaly dominovat hluboké konvoluční neuronové sítě. Vítěz v roce 2015 dosáhl správnosti 96,4 % a klasifikační úloha na *ImageNetu* byl považována za zcela vyřešený problém.

Od roku 2012 se hluboké konvoluční neuronové sítě (*convolutional neural network, convnet, CNN*) staly výchozím algoritmem pro všechny úlohy počítačového vidění, obecněji pro všechny úlohy související s vnímáním. Na hlavních konferencích o počítačovém vidění bylo v letech 2015 a 2016 téměř nemožné nalézt prezentace, které by nezmiňovaly nějakou jejich formu. Současně hluboké učení nalezlo uplatnění i v mnoha dalších typech problémů, jako je zpracování přirozeného jazyka. V široké škále aplikací úplně nahrazuje SVM a rozhodovací stromy.

Například Evropská organizace pro jaderný výzkum CERN používala pro analýzu dat o částicích z detektoru ATLAS u *Large Hadron Collider* (LHC) metody analýzy založené na rozhodovacích stromech. Nakonec ale CERN přešel na hluboké neuronové sítě založené na *Keras* kvůli jejich vyššímu výkonu a snadnému trénování na velkých datových sadách.

---

[6] *Pozn. překladatele:* Úspěšnost typu top-5 říká, u kolika procent klasifikací byla správná klasifikace mezi pěti označenými jako nejpravděpodobnější. Úspěšnost typu top-1 říká, v kolika procentech případů byla klasifikace správná.

1.2.6 Co dělá hluboké učení odlišným

Hlavním důvodem, proč hluboké učení tak rychle roste, je to, že v řadě případů nabízí lepší výkon. Není to ale jediný důvod. Hluboké učení také usnadňuje řešení problémů, protože zcela automatizuje to, co bylo nejdůležitějším krokem v postupech strojového učení: konstrukci rysů.

Předchozí techniky strojového učení – mělké učení – zahrnovaly pouze transformaci vstupních dat do jednoho nebo dvou po sobě jdoucích reprezentačních prostorů, obvykle prostřednictvím jednoduchých transformací, jako jsou například mnoharozměrné nelineární projekce (SVM) nebo rozhodovací stromy. Ale vylepšených reprezentací vyžadovaných komplexními problémy není obecně možné těmito technikami dosáhnout. Lidé museli vynaložit velké úsilí, aby počáteční vstupní data zpřístupnili pro zpracování těmito metodami – museli pro svá data ručně vytvářet dobré vrstvy reprezentací.

Toto předzpracování je označováno jako *inženýrství rysů* (*feature engineering*). Na druhou stranu hluboké učení tento krok zcela automatizuje. S hlubokým učením se učíte všechny rysy v jednom průchodu, aniž byste je museli sami vyvíjet. Značně to zjednodušilo pracovní postupy strojového učení, protože časté sofistikované, mnohastavové zřetězené procesy byly nahrazeny jediným, jednoduchým, hlubokým učením modelu.

Můžete se zeptat: je-li podstatou problému zavedení více vrstev reprezentací, nemohlo by opakované použití mělkých metod emulovat účinky hlubokého učení?

V praxi po sobě jdoucí aplikace metod mělkého učení přinášejí rychle klesající výnosy, protože optimální první vrstva reprezentace v třívrstvém modelu není optimální první vrstvou v jednovrstvém nebo dvouvrstvém modelu.

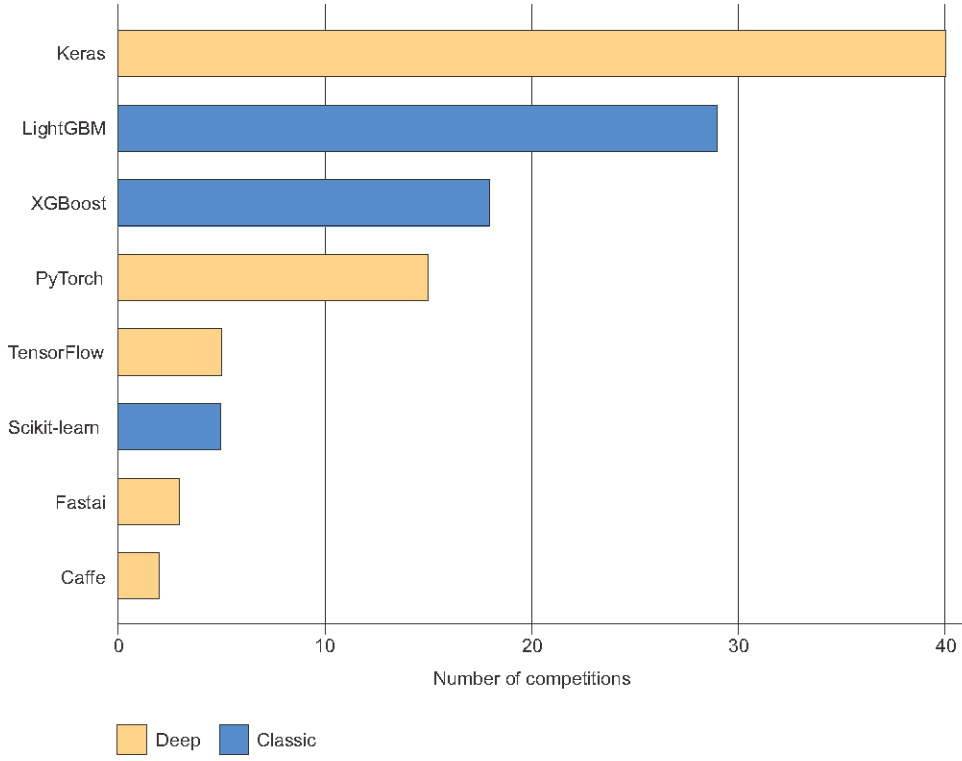
Klíčové na hlubokém učení je to, že umožňuje modelu, aby se všechny vrstvy reprezentace učily *současně* (někdy se říká hladově – greedily). Díky společnému učení rysů se vždy, když model upraví jeden ze svých vnitřních rysů, této změně automaticky přizpůsobí všechny ostatní rysy, které na daném rysu závisejí, aniž by vyžadovaly lidský zásah. Vše je pod dohledem jediného zpětnovazebního signálu: každá změna v modelu slouží konečnému cíli. Je to mnohem výkonnější než nenasytné skládání mělkých modelů, protože to umožňuje učit se složité, abstraktní reprezentace rozdělením na dlouhé řady meziprostorů (vrstev); každý prostor je pouze jednoduchou transformací vzdálen od předchozího.

Toto jsou dvě základní charakteristiky způsobu, jak se hluboké učení učí z dat: *postupné vytváření stále složitějších reprezentací po jednotlivých vrstvách* a skutečnost, *že se tyto mezistupňové inkrementální reprezentace učí společně*, přičemž každá vrstva je aktualizována tak, aby sledovala jak reprezentační potřeby vrstvy nad ní, tak potřeby vrstvy pod ní. Díky těmto dvěma vlastnostem je hluboké učení společně mnohem úspěšnější než předchozí přístupy ke strojovému učení.

1.2.7 Krajina moderního strojového učení

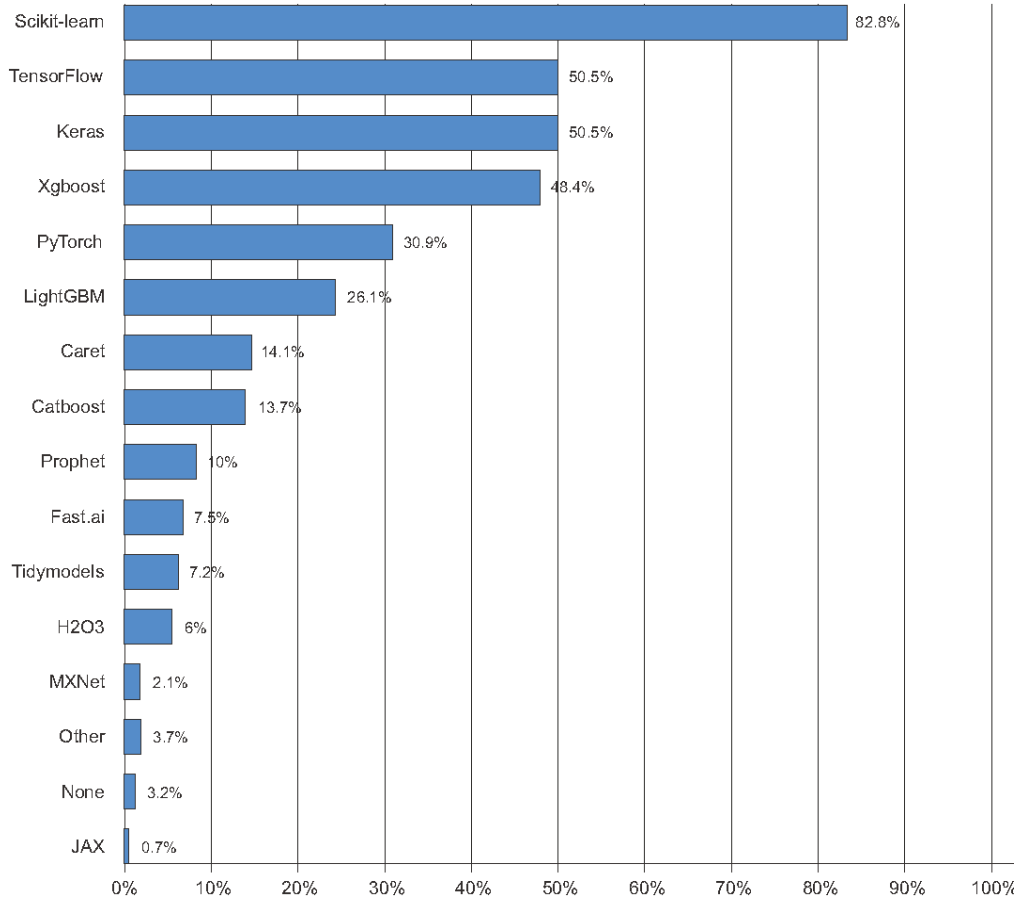
Skvělý způsob, jak získat představu o současném prostředí algoritmů a nástrojů strojového učení, je podívat se na soutěže ve strojovém učení na *Kaggle*. Díky vysoce konkurenčnímu prostředí (některé soutěže mají tisíce účastníků a nabízejí ceny v milionech dolarů) a širokému spektru problémů strojového učení nabízí *Kaggle* realistický způsob, jak posoudit, co funguje a co ne. Takže jaký druh algoritmu spolehlivě vyhrává soutěže? Jaké nástroje využívají nejlepší účastníci?

Na začátku roku 2019 provedla společnost *Kaggle* průzkum, v němž se zeptala týmů, které se od roku 2017 umístily na prvních pěti místech v jakékoli soutěži, jaký primární softwarový nástroj v soutěži použily (viz obrázek 1.12). Ukázalo se, že nejlepší týmy mají tendenci používat buď metody hlubokého učení (nejčastěji prostřednictvím frameworku *Keras*), nebo stromy využívající posílení gradientu (nejčastěji prostřednictvím knihoven *LightGBM* nebo *XGBoost*).



Obrázek 1.12:
Nástroje strojového učení používané nejlepšími týmy na Kaggle

Nejde jen o šampiony soutěží. *Kaggle* také každoročně pořádá průzkum mezi profesionály v oblasti strojového učení a datové vědy po celém světě. Díky desítkám tisíc respondentů je tento průzkum jedním z nejspolehlivějších zdrojů informací o stavu oboru. Na obrázku 1.13 je znázorněno procentuální využití různých softwarových rámců pro strojové učení.



Obrázek 1.13:
Využití nástrojů v odvětví strojového učení a datové vědy
(Zdroj: www.kaggle.com/ kaggle-survey-2020)

Od roku 2016 do roku 2020 dominovaly celému odvětví strojového učení a datové vědy dva přístupy: hluboké učení a gradientní stromy. Posílení gradientu se používá pro problémy, kde jsou k dispozici strukturovaná data, zatímco hluboké učení se používá pro percepční problémy, jako je klasifikace obrazu.

Uživatelé stromů s posílením gradientu obvykle používají framework *Scikit-learn*, *XGBoost* nebo *LightGBM*. Většina praktických uživatelů hlubokého učení používá *Keras*, často v kombinaci se systémem jeho nadřazeného frameworku *TensorFlow*. Společným bodem těchto nástrojů je, že všechny jsou knihovnami nebo frameworky *Pythonu*: *Python* je zdaleka nejrozšířenějším jazykem pro strojové učení a datovou vědu.

Tyto dvě techniky byste tedy měli znát nejlépe, chcete-li být dnes v aplikovaném strojovém učení úspěšní: posílený gradient pro problémy mělkého učení a hluboké učení pro problémy vnímání. Z technického hlediska to znamená, že budete muset znát *Scikit-learn*, *XGBoost* a *Keras* – tři frameworky, které v současnosti dominují soutěžím *Kaggle*. S touto knihou v ruce jste tomuto cíli o velký krok blíže.

1.3 Proč hluboké učení? Proč teď?

Dvě klíčové myšlenky hlubokého učení pro počítačové vidění, konvoluční neuronové sítě a zpětná propagace, byly dobře známé již v roce 1990. Algoritmus dlouhé krátkodobé paměti (*Long Short-Term Memory* – LSTM), který je základem hlubokého učení pro časové řady, byl vyvinut v roce 1997 a od té doby se již téměř nezměnil. Proč tedy hluboké učení vzlétlo až po roce 2012? Co se v těchto dvou desetiletích změnilo?

Obecně platí, že k pokroku ve strojovém učení vedou tři technické síly:

- Hardware.
- Sady dat a referenční hodnoty.
- Pokroky v algoritmizaci.

Vzhledem k tomu, že oblast je řízena spíše experimentálními nálezy než teorií, je pokrok v algoritmizaci možný pouze tehdy, když jsou k dispozici vhodné datové a hardwarové nástroje, které umožní vyzkoušet nové nápady (nebo vylepšení starších nápadů, jak tomu často bývá). Strojové učení není matematika ani fyzika, kde lze dosáhnout významného pokroku pomocí pera a kusu papíru. Je to inženýrská věda.

Skutečné překážky v letech 1990–2000 představovala data a hardware. Hlavními změnami během této doby bylo rozšíření a zrychlení internetu a vyvinutí vysoce výkonných grafických čipů pro potřeby herního trhu.

1.3.1 Hardware

Mezi roky 1990 a 2010 došlo ke zhruba 5000násobnému zrychlení procesorů. Výsledkem je, že v dnešní době můžete na svém notebooku provozovat malé modely s hlubokým učením, což by před 25 lety nebylo možné.

Nicméně typické modely hlubokého učení, které se používají při počítačovém vidění nebo při rozpoznávání řeči, vyžadují mnohem větší výpočetní výkon než to, co může poskytnout váš notebook.

Na počátku tohoto století investovaly společnosti jako NVIDIA a AMD miliardy dolarů do vývoje rychlých, masivně paralelních čipů (grafických procesorových jednotek – GPU), které oživovaly grafiku stále více fotorealistických videoher. Byly to levné, jednoúčelové superpočítače navržené tak, aby na obrazovce poskytovaly komplexní 3D scény v reálném čase.

Tato investice přinesla užitek i vědecké komunitě, když v roce 2007 společnost NVIDIA uvedla pro svoji řadu GPU programovací rozhraní CUDA (zkratka z *Compute Unified Device Architecture* – https://developer.nvidia.com/about-cuda). Malý počet GPU začalo nahrazovat masivní seskupení procesorů v různých vysoce paralelizovatelných aplikacích, počínaje modelováním fyziky.

Hluboké neuronové sítě, skládající se převážně z mnoha násobení malých matic, jsou rovněž vysoce paralelizovatelné. Kolem roku 2011 proto někteří výzkumníci za-

čali psát CUDA implementace neuronových sítí – mezi prvními byli Dan Ciresan[7] a Alex Krizhevsky.[8]

Došlo k tomu, že superpočítače pro novou generaci aplikací umělé inteligence dotoval herní trh. Velké věci někdy začínají jako hry. Grafický procesor NVIDIA Titan RTX, který na konci roku 2019 stál 2500 dolarů, dnes dokáže dosáhnout špičkového výkonu 16 teraFLOPS v jednoduché přesnosti (16 bilionů operací `float32` za sekundu). To je přibližně 500krát vyšší výpočetní výkon než nejrychlejší superpočítač na světě z roku 1990, *Intel Touchstone Delta*.

Na počítači Titan RTX trvá trénování modelu *ImageNet*, který by vyhrál soutěž ILSVRC kolem roku 2012 nebo 2013, jen několik hodin. Mezitím velké společnosti trénují modely hlubokého učení na klastrech se stovkami GPU.

Odvětví hlubokého učení se navíc přesouvá za hranice grafických procesorů a investuje do stále specializovanějších a efektivnějších čipů pro hluboké učení. V roce 2016 společnost *Google* na své výroční konferenci I/O odhalila svůj projekt *Tensor Processing Unit* (TPU): nový design čipu, který byl od základu vyvinut pro provozování hlubokých neuronových sítí výrazně rychleji a mnohem úsporněji než špičkové GPU.

Dnes, v roce 2020, představuje třetí iterace karty TPU výpočetní výkon 420 teraFLOPS. To je 10 000krát více než u *Intel Touchstone Delta* z roku 1990.

Tyto karty TPU jsou určeny k sestavení do rozsáhlých konfigurací, tzv. „pods“. Jeden modul (1024 karet TPU) dosahuje maximálního výkonu 100 petaFLOPS. Pro představu, to je asi 10 % špičkového výpočetního výkonu současného největšího superpočítače IBM Summit v *Oak Ridge National Lab*, který se skládá z 27 000 grafických procesorů NVIDIA a dosahuje špičkového výkonu přibližně 1,1 exaFLOPS.

1.3.2 Data

AI je někdy označována jako nová průmyslová revoluce. Je-li hluboké učení parní stroj této revoluce, potom jsou data jeho uhlí: surovina, která ovládá naše inteligentní stroje, bez nichž by nic nebylo možné. Pokud jde o data, tak kromě exponenciálního pokroku v oblasti hardwaru pro ukládání dat za posledních 20 let (podle Moorova zákona), představoval rozhodující průlom vzestup internetu, což umožnilo shromažďovat a distribuovat velmi rozsáhlé datové soubory pro strojové učení.

Dnes pracují velké společnosti s obrazovými daty, videodaty a daty z přirozených jazyků, která nemohla být shromažďována bez internetu. Například uživatelsky generované obrazové značky na *Flickru* jsou pokladnicí pro počítačové vidění, stejně jako videa na *YouTube*. A *Wikipedie* je klíčovou datovou sadou pro zpracování přirozeného jazyka.

---

[7] Viz *Flexible, High Performance Convolutional Neural Networks for Image Classification*, Proceedings of the 22nd International Joint Conference on Artificial Intelligence (2011), www.ijcai.org/Proceedings/11/Papers/210.pdf.

[8] Viz *ImageNet Classification with Deep Convolutional Neural Networks*, Advances in Neural Information Processing Systems 25 (2012), http://mng.bz/2286.

Existuje-li jedna datová sada, která byla katalyzátorem vzestupu hlubokého učení, je to datová sada *ImageNet*, jež se skládá ze 1,4 milionu snímků, z nichž každý byl ručně zařazen do jedné z 1000 kategorií. Ale to, co dělá *ImageNet* výjimečným, není jen jeho velikost, ale také s ním spojená každoroční soutěž.[9]

Jak od roku 2010 demonstruje *Kaggle*, veřejné soutěže jsou vynikajícím způsobem, jak motivovat výzkumníky a inženýry, aby posouvali hranice. Společné benchmarky, o jejichž překonání výzkumníci soutěží, výrazně pomohly vzestupu hlubokého učení tím, že zdůraznily jeho úspěšnost v porovnání s klasickými přístupy strojového učení.

1.3.3 Algoritmy

Hardware a data nebyly jediným problémem. Až do konce roku 2000 chyběl spolehlivý způsob učení velmi hlubokých neuronových sítí. V důsledku toho byly neuronové sítě stále poměrně mělké – používaly většinou jen jednu nebo dvě vrstvy reprezentací. Proto také nemohly obstát proti rafinovanějším mělkým metodám, jako jsou SVM a náhodné lesy. Klíčovým problémem bylo *šíření gradientu* (*gradient propagation*) přes řadu vrstev. Zpětnovazební signál používaný k trénování neuronových sítí se s rostoucím počtem vrstev vytrácel.

To se změnilo v letech 2009 až 2010 s nástupem několika jednoduchých, ale důležitých algoritmických vylepšení, které umožnily lepší šíření gradientu:

- Lepší *aktivační funkce* pro neuronové vrstvy.
- Lepší *schémata inicializace vah*, počínaje předtrénováním vrstev, které bylo ale rychle opuštěno.
- Lepší *optimalizační schémata*, jako jsou *RMSProp* a *Adam*.

Teprve když tato zlepšení umožnila trénovat modely s 10 nebo více vrstvami, začalo hluboké učení zářit.

Konečně pak byly v letech 2014, 2015 a 2016 objeveny ještě pokročilejší způsoby, jak napomoci šíření gradientů, jako je dávková normalizace, zbytková připojení a hloubkově oddělitelné konvoluce.

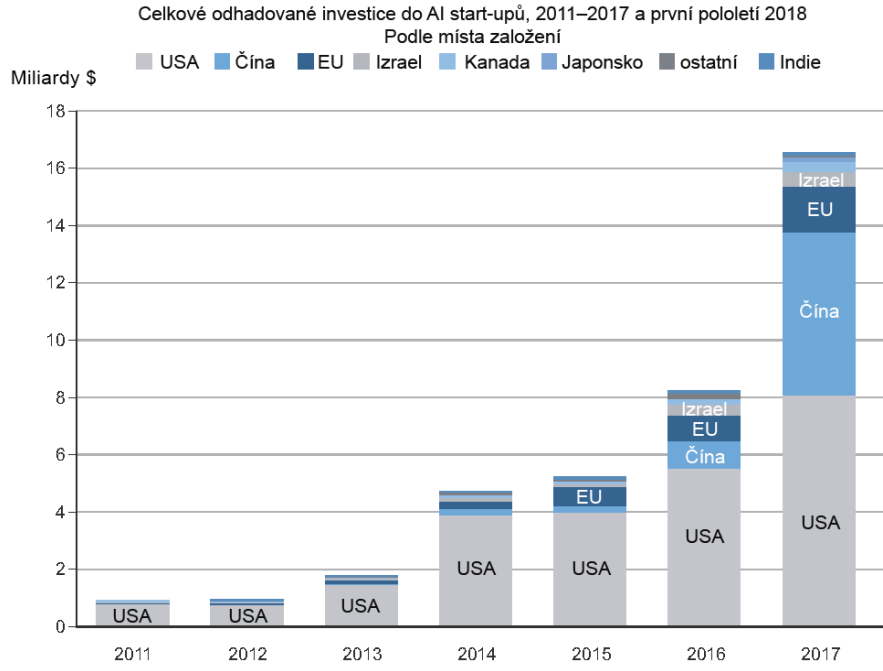
Dnes můžeme trénovat libovolně hluboké modely od nuly. Umožnilo to používat extrémně velké modely, které mají značnou reprezentační sílu, což znamená, že kódují velmi bohaté prostory hypotéz. Tato extrémní škálovatelnost je jednou z určujících charakteristik moderního hlubokého učení. Rozsáhlé modelové architektury, které obsahují desítky vrstev a desítky milionů parametrů, přinesly zásadní pokrok jak v počítačovém vidění (například architektury jako *ResNet*, *Inception* nebo *Xception*), tak ve zpracování přirozeného jazyka (například rozsáhlé architektury založené na *Transformátorech* jako *BERT*, *GPT-3* nebo *XLNet*).

---

[9] *The ImageNet Large Scale Visual Recognition Challenge* (ILSVRC), www.image-net.org/challenges/LSVRC.

1.3.4 Nová vlna investic

Vzhledem k tomu, že se v letech 2012 až 2013 stalo hluboké učení novým nejmodernějším postupem v oblasti počítačového vidění a dalších oblastí vnímání, vzali ho vedoucí představitelé oboru na vědomí. Postupně se spustila vlna investic, která daleko přesahovala vše, co bylo zatím v historii AI pozorováno (viz obrázek 1.14).



Obrázek 1.14:
Odhad celkových investic OECD do startupů s umělou inteligencí (Zdroj: http://mng.bz/zGN6)

V roce 2011, těsně před tím, než hluboké učení získalo pozornost, se celkové investice rizikového kapitálu v AI pohybovaly kolem miliardy dolarů: téměř všechny plynuly do praktických aplikací mělkých přístupů k strojovému učení. V roce 2015 to bylo již více než 5 miliard dolarů a v roce 2017 závratných 16 miliard dolarů. Během těchto několika let vznikly stovky startupů, které se snažily využít humbuk kolem hlubokého učení. Velké technologické společnosti, jako je *Google*, *Amazon* a *Microsoft*, mezitím investovaly do interních výzkumných oddělení částky, které by s největší pravděpodobností objem rizikového kapitálu převýšily.

Strojové učení, a zejména pak hluboké učení, se stalo centrem produktové strategie těchto technologických gigantů. Koncem roku 2015 generální ředitel *Google* Sundar Pichai uvedl: „Strojové učení je jádrem, transformačním způsobem, kterým přehodnocujeme, jak všechno děláme. Smysluplně jej aplikujeme na všechny naše produkty, ať už je to vyhledávání, reklamy, *YouTube* nebo *Play*. Prozatím jsme na počátku, ale postupně uvidíte, jak systematicky používáme strojové učení ve všech těchto oblastech.“[10]

[10] Sundar Pichai, Prezentace výsledků společnosti *Alphabet*, 22. října 2015.

V důsledku této vlny investic se počet lidí pracujících na hlubokém učení za méně než deset let zvýšil z několika set na desítky tisíc a pokrok v oblasti výzkumu dosáhl frenetického tempa. V současné době neexistují náznaky, že se tento trend brzy zpomalí.

1.3.5 Demokratizace hlubokého učení

Jedním z klíčových faktorů, který způsobil příliv nových tváří do hlubokého učení, byla demokratizace nástrojů používaných v této oblasti. V raných dobách vyžadovalo hluboké učení významné odborné znalosti C++ a CUDA, kterými disponovalo jen málo lidí.

V současné době postačují k pokročilému výzkumu hlubokého učení základní znalosti skriptování v jazyce *Python*. Přispěl k tomu především vývoj dnes již neexistujícího frameworku *Theano* a poté frameworku *TensorFlow* – dvou symbolických frameworků pro manipulaci s tenzory v jazyce *Python* podporujícími autodiferenciaci, což výrazně zjednodušuje implementaci nových modelů. Dalším stimulem byl vznik uživatelsky přívětivých frameworků jako *Keras*, díky nimž je hluboké učení stejně snadné jako manipulace s kostkami LEGO.

Keras se po vydání na začátku roku 2015 rychle stal řešením pro hluboké učení pro velké množství nových začínajících firem, postgraduálních studentů a výzkumných pracovníků, kteří se této oblasti věnovali.

1.3.6 Vydrží to?

Je na hlubokých neuronových sítích něco zvláštního, co z nich dělá ten „správný“ přístup, do nějž by společnosti mohly investovat a výzkumníci houfně následovat? Nebo je hluboké učení jen bláznivý nápad, který dlouho nevydrží? Budeme hluboké neuronové sítě používat i za 20 let?

Hluboké učení má několik vlastností, které ho opravňují k tomu, abychom ho považovali v oblasti umělé inteligence za revoluci, a nepochybně s námi zůstane. Za dvacet let možná nebudeme používat neuronové sítě, ale to, co budeme používat, bude přímo navazovat na moderní hluboké učení a jeho základní koncepty. Tyto důležité vlastnosti lze obecně roztřídit do tří kategorií:

- *Jednoduchost* – Hluboké učení odstraňuje potřebu inženýrství rysů. Nahrazuje složité, křehké posloupnosti zpracování jednoduchými end-to-end trénovatelnými modely,[11] které jsou typicky postaveny pomocí pouhých pěti nebo šesti tenzorových operací.
- *Škálovatelnost* – Hluboké učení je velmi přístupné paralelizaci na GPU nebo TPU, takže může plně využít Moorova zákona. Modely hlubokého učení jsou navíc

[11] *Pozn. překladatele:* Termínem *end-to-end trénovatelný model* se označují modely, které mohou být trénovány na datech, jež jsou ve stejném formátu jako data, která budou použita pro rozhodování za provozu.

školeny iterací přes malé dávky dat, což jim umožňuje natrénování na datových sadách libovolné velikosti. (Jediným úzkým hrdlem je množství dostupného paralelního výpočetního výkonu, které se však díky Moorovu zákonu rychle zvětšuje.)

- *Všestrannost a opakovatelnost* – Na rozdíl od mnoha předchozích přístupů k strojovému učení mohou být modely s hlubokým učením trénovány pomocí dodatečných dat, aniž by je bylo nutné restartovat. To je činí životaschopnými pro nepřetržité online učení – což je důležitá vlastnost velkých výrobních modelů.

  Modely natrénované hlubokým učením jsou víceúčelové a opakovaně použitelné: například je možné vzít model pro hluboké učení natrénovaný pro klasifikaci obrazu a přemístit ho do datovodu (pipeline) pro zpracování obrazu. Umožňuje to reinvestovat předchozí práci do stále složitějších a výkonnějších modelů. Také to činí hluboké učení použitelným i pro poměrně malé soubory dat.

Hluboké učení je v centru pozornosti teprve několik let a my jsme dosud neurčili celý rozsah toho, co může udělat. Každý další měsíc se dozvídáme o nových případech použití a technických zlepšeních, která ruší předchozí omezení.

V návaznosti na vědeckou revoluci pokrok obecně postupuje podle sigmoidy,[12] která začíná obdobím rychlého pokroku a ve chvíli, kdy výzkumníci narazí na tvrdá omezení a další vylepšení už jsou marginální, přejde do fáze stabilizace.

Když jsem v roce 2016 psal první vydání této knihy, předpovídal jsem, že hluboké učení se stále nachází v první polovině této sigmoidy a že v následujících několika letech dojde k mnohem většímu transformačnímu pokroku. To se v praxi ukázalo jako pravdivé, protože v letech 2017 a 2018 se objevily modely hlubokého učení založené na *Transformátoru* pro zpracování přirozeného jazyka, které v této oblasti znamenaly revoluci.

Hluboké učení také přináší stabilní pokrok v oblasti počítačového vidění a rozpoznávání řeči. Dnes (v roce 2021) se zdá, že hluboké učení vstoupilo do druhé poloviny této sigmoidy. V příštích letech bychom měli stále očekávat významný pokrok, ale pravděpodobně je již počáteční fáze explozivního pokroku za námi.

Dnes jsem nesmírně nadšený z nasazení technologie hlubokého učení na všechny problémy, které dokáže vyřešit – seznam je nekonečný. Hluboké učení je stále ještě revolucí ve vývoji a bude trvat mnoho let, než se jeho potenciál projeví naplno.

[12] *Pozn. překladatele:* Sigmoida = křivka ve tvaru protaženého písmene S – viz obrázek 4.3 na straně 6 nebo https://en.wikipedia.org/wiki/Sigmoid_function.

Kapitola 2
Než začneme: matematické stavební bloky neuronových sítí

Tato kapitola probírá:

- První příklad neuronové sítě.
- Tenzory a tenzorové operace.
- Jak se neuronové sítě učí prostřednictvím zpětného šíření a gradientního sestupu.

Pochopení hlubokého učení vyžaduje znalost mnoha jednoduchých matematických pojmů: *tenzory*, *tenzorové operace*, *diferenciace*, *gradientní sestup* a dalších. V této kapitole bude naším cílem seznámit vás s těmito pojmy, abyste o nich získali přehled a aniž bychom museli zabíhat do technických detailů. Zejména se budeme vyhýbat matematickému zápisu, jenž může být pro ty, kteří nemají dostatečné matematické základy, složitý (a navíc není pro správné pochopení nezbytný). Nejpřesnějším a nejjednoznačnějším popisem matematické operace je její spustitelný kód.

Abychom vám poskytli dostatečný kontext pro představení tenzorů a gradientního sestupu, začneme kapitolu praktickým příkladem neuronové sítě. Poté postupně probereme každý nový zavedený pojem. Mějte na paměti, že tyto pojmy budou pro pochopení praktických příkladů v následujících kapitolách nezbytné!

Po přečtení této kapitoly budete chápat princip toho, jak fungují neuronové sítě, a budete se moci přesunout k praktickým aplikacím, kterým se budeme věnovat ve třetí kapitole.

2.1 První pohled na neuronovou síť

Podívejme se na konkrétní příklad neuronové sítě, která se pomocí frameworku *Keras* v jazyce *Python* učí klasifikovat ručně psané číslice. Pokud ještě nemáte s frameworkem *Keras* nebo podobnými frameworky zkušenosti, nebudete zřejmě v tomto prvním příkladu hned vše chápat. To je v pořádku.

V další kapitole každý prvek příkladu prozkoumáme a podrobně vysvětlíme. Takže se nebojte, pokud vám budou některé kroky připadat náhodné nebo záhadné. Někde začít musíme.

Problém, který se zde pokusíme vyřešit, je třídění obrázků ručně psaných číslic (28 × 28 pixelů) ve stupních šedi do 10 kategorií (0 až 9). Použijeme databázi MNIST, která je v komunitě strojového učení klasikou. Jedná se o soubor 60 000 trénovacích plus 10 000 testovacích obrazů, sestavený v osmdesátých letech Národním institutem pro standardy a technologie (National Institute of Standards and Technology – *NIST*)[13]. Databázi MNIST můžeme považovat za takový „*Hello world*" hlubokého učení – používá se k validaci, že algoritmy pracují podle očekávání. Při dalším studiu se s ní budete pravidelně setkávat ve vědeckých článcích, blogových příspěvcích apod. Několik ukázek si můžete prohlédnout na obrázku 2.1.

Poznámka překladatele
Při strojovém učení se kategorie v problému s klasifikací označuje jako *třída* (*class*). Datové body se pak nazývají *příklady*, *vzorky* (*samples*) nebo *záznamy* (*records*). Informace o příslušnosti příkladu do dané třídy se nazývá *označení třídy* nebo *štítek* (*label*).

Obrázek 2.1:
Číslice z databáze MINST

Nemusíte se hned pokoušet reprodukovat tento příklad na svém počítači. Pokud to ale chcete zkusit, musíte nejprve zprovoznit *Keras* podle návodu v kapitole 3.[14]

Datová sada MNIST je v systému *Keras* předinstalována ve formě sady čtyř polí *NumPy*.

---

[13] *Pozn. překladatele:* Podrobněji se o této databázi můžete dočíst ve Wikipedii na adrese https://en.wikipedia.org/wiki/MNIST_database.

[14] *Pozn. překladatele:* Jestli netoužíte po úvodním seznámení s frameworky a jejich historií, ale chcete hned skočit na pasáž o nastavování prostředí, přejděte rovnou na podkapitolu *3.4 Nastavení pracovního prostoru pro hluboké učení* na straně 6.

Výpis 2.1: *Načtení sady MINSET v* Keras

```
>>> from tensorflow.keras.datasets import mnist
>>> (train_images, train_labels), (test_images, test_labels) = mnist.load_data()
Downloading data from https://storage.googleapis.com/tensorflow/tf-keras-
datasets/mnist.npz
11490434/11490434 [==============================] - 1s 0us/step
>>>
```

Trénovací sadu (*training set*), z níž se model učí, tvoří `train_images` a `train_labels`. Model pak bude testován na *testovací sadě* (*test set*) `test_images` a `test_labels`.[15] Obrazy jsou kódovány jako *NumPy* pole, označení tříd jsou reprezentována poli číslic od 0 do 9. Obrázky a označení tříd si vzájemně odpovídají.

Podívejme se na trénovací data:

```
>>> train_images.shape
(60000, 28, 28)
>>> len(train_labels)
60000
>>> train_labels
array([5, 0, 4, ..., 5, 6, 8], dtype=uint8)
```

A nyní na testovací data:

```
>>> test_images.shape
(10000, 28, 28)
>>> len(test_labels)
10000
>>> test_labels
array([7, 2, 1, ..., 4, 5, 6], dtype=uint8)
```

Pracovní postup bude následující: Nejprve napojíme neuronovou síť na trénovací data `train_images` a `train_labels`. Síť se naučí sdružit obrázky s odpovídajícími označeními tříd. Nakonec požádáme síť, aby vytvořila predikce pro `test_images`, a ověříme, zda se tyto predikce shodují s odpovídajícími označeními tříd z `test_labels`.

Vytvoříme nyní síť. Nezapomínejte, že se od vás prozatím neočekává, že v tomto příkladu ihned všechno pochopíte.

Výpis 2.2: *Architektura modelu*

```
>>> from tensorflow import keras
>>> from tensorflow.keras import layers
>>> model = keras.Sequential([
...     layers.Dense(512, activation="relu"),
...     layers.Dense(10, activation="softmax")
... ])
```

Základním stavebním blokem neuronových sítí je *vrstva* (*layer*) zpracovávající modul, který můžete považovat za datový filtr. Některá data do nich vstoupí a vystoupí v nějaké užitečnější formě. Konkrétně: vrstvy ze vstupujících dat, která jsou do nich vkládána,

[15] *Pozn. překladatele:* Někdy hovoříme o trénovací, resp. testovací sadě.