

example 1: cats vs dogs

we get 50%

solutions to improve the model:

- collect more data
- collect more diverse training set.
- train algorithm longer with gradient descent
- try adam instead of gradient descent
- try bigger network
- try smaller network
- try dropout
- add L2 regularization
- change network architecture
 - activation functions
 - # hidden units

chain of assumptions of ML

- 1/ fit training set well on cost function
↓
- 2/ fit dev set well on cost function
↓
- 3/ fit test set well on cost function
↓
- 4/ performs well in real world

the algorithm fails in step 1

- bigger network
- better optimization algorithm

if the algorithm fails on dev set

- regularization
- bigger training set

if the algorithm fails on test set

- get bigger dev set : the algorithm overfits to the dev set

if it fails in real world

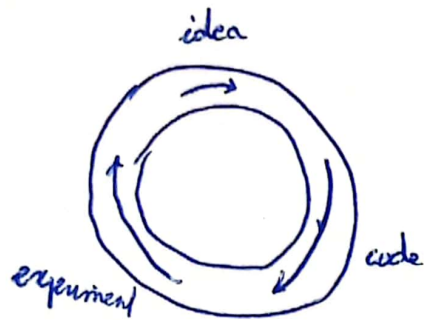
- change dev set
- change the cost function

1/ Choisir une archi simple

- perceptron multi couche - leNet - LSTM
- +
- fonction d'activation par défaut
- +
- normalisation des données

why AI wasn't available:

- limited computational power
- interesting applications such as image recognition require large amounts of data that were not available



- 1 - larger amounts of data allow researchers to try more ideas (better models) but require a longer time to train
 - 2 - improvements in the CPU/GPU hardware enable the discovery of better deep learning algorithms
 - 3 - better algorithms can speed up the iterative process by reducing the necessary computation time
-

the two fundamental assumptions of supervised learning:

1/ you can fit the training set pretty well
(low avoidable bias)

2/ the training set performance generalizes pretty well to the dev/test set
(low variance)

Reducing

human-level error as a proxy for Bayes error.

example: medical image classification

suppose

typical human: 3% error

typical doctor: 1% error

experienced doctor: 0.7% error

team of experienced doctors: 0.5% error

what is human level error?

Bayes error $\leq 0.5\%$

\Rightarrow human level error: 0.5%

human level error: 1% / 0.7% / 0.5%

training error: 5% 5% / 5%

dev error: 6% 6% 6%

The choice doesn't matter: focus on bias

1%	0.7%	0.5%
1%	1%	1%
5%	5%	5%

The choice of human level doesn't matter:

focus on variance

5%	0.7%	0.5%
0.7%	0.7%	0.7%
0.8%	0.8%	0.8%

Here we need to choose 0.5% as a human level

surpassing human level performance

team of humans:	0.5%	0.5%
one human:	1%	1%
training error:	0.6%	0.5%
dev error:	0.8%	0.4%
avoidable bias:	0.1%	
variance:	0.8%	

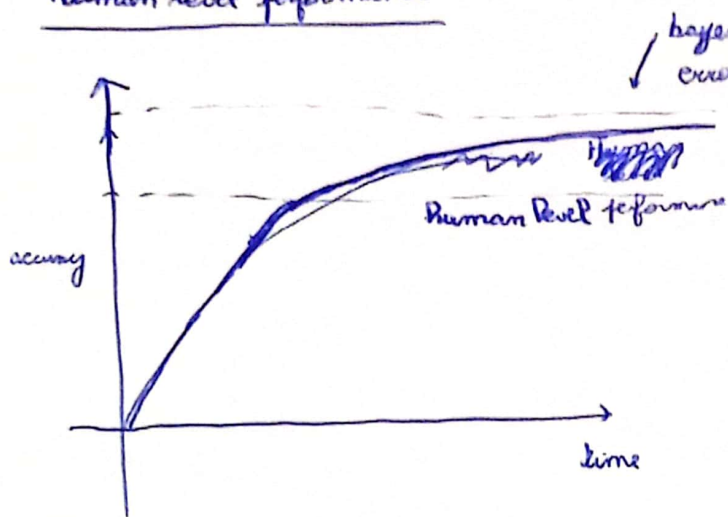
problems where ML outperforms human level performance

<ul style="list-style-type: none"> - online advertising - product recommendations - logistics (predicting transit time) - loan approvals 	} structured data
--	-------------------

\Rightarrow not natural perception tasks

- speech recognition
- some image recognition tasks
- medical tasks

Human-level performance



Bayes optimal error : best possible error

why progress slows down after surpassing human level performance?

1/ human level performance is not so far from Bayes optimal error in some cases.

2/ when we are below human level performance there are many tools to use to surpass ^{it} human level performance but harder to use after surpassing human level performance

So long as HL is worse than humans, you can:

- get labeled data from humans
- gain insight from manual error analysis (manual checking)
- better analysis of bias / variance

cat classification example 1

Humans: 1%

training error: 8%

dev error: 10%

⇒ we want to do better on train

(the algo is not fitting well the train dataset)

⇒ focus on reducing bias

example 2

Humans: 7.5%

training error: 8%

dev error: 10%

⇒ focus on reducing variance

⚠ for computer vision tasks, we think of human level error as a proxy for Bayes error.

1/ ~~the~~ difference between Bayes error and train error:

avoidable bias

⚠ in overfitting, we can achieve ~~less~~ ^{error} ~~error~~ ^{error} than Bayes error

2/ difference between train error and dev error:

variance

development set:

you try different models, you train on train set
use the dev set to pick one model
keep iterating to improve the model performance
until we find the classifier we are happy with
test it on the test set.

example: cat classifier

Region: US - UK - other Europe - South America
India - China - other Asia - Australia

1/ first way:

use US - UK - other Europe - South America:

Dev Set

the others

Test set

⇒ bad idea because Dev and Test come from
different distribution

2/ second way: the best

Randomly shuffle the data into dev and test
sets.

Guideline:

choose a dev set and test set to reflect data you
expect to get in the future and consider important to
do well on

sometimes we need to train the model on
the data that is available and its distribution
may not be the same as the data that will
occur in production.

adding training data that differs from the dev
set may still help the model improve the performance
on the test set.

what matters is dev and test sets have the same
distribution

size of the test set

it must be big enough to give high confidence
in the overall performance of your system

when changing dev and test sets metric

metric error

A: 3% error + shows inappropriate images

B: 5% error + no inappropriate images

metric + dev set: prefer A (lower error)

Re/users: prefer B (no inappropriate images)

⇒ we need to change the metric on the dev set

$$\text{Error}_{\text{initial}} = \frac{1}{m_{\text{dev}}} \sum_{i=1}^{m_{\text{dev}}} \mathbb{1} \{y_{\text{pred}}^{(i)} \neq y^{(i)}\}$$

$$\Downarrow$$
$$\text{Error}_{\text{changed}} = \frac{1}{\sum w^{(i)}} \sum_{i=1}^{m_{\text{dev}}} w^{(i)} \mathbb{1} \{y_{\text{pred}}^{(i)} \neq y^{(i)}\}$$

$$w^{(i)} = \begin{cases} 1 & \text{if } x^{(i)} \text{ is non porn} \\ 10 & \text{if } x^{(i)} \text{ is porn image} \end{cases}$$

algorithm A: 3% on dev set

B: 5% on dev set

But algorithm B is doing better on the production

in dev: high quality images
in prod: low quality images

⇒ change the dev/test set

1. 2. 3. 4. 5. 6. 7. 8. 9. 10. 11. 12. 13. 14. 15. 16. 17. 18. 19. 20. 21. 22. 23. 24. 25. 26. 27. 28. 29. 30. 31. 32. 33. 34. 35. 36. 37. 38. 39. 40. 41. 42. 43. 44. 45. 46. 47. 48. 49. 50. 51. 52. 53. 54. 55. 56. 57. 58. 59. 60. 61. 62. 63. 64. 65. 66. 67. 68. 69. 70. 71. 72. 73. 74. 75. 76. 77. 78. 79. 80. 81. 82. 83. 84. 85. 86. 87. 88. 89. 90. 91. 92. 93. 94. 95. 96. 97. 98. 99. 100. 101. 102. 103. 104. 105. 106. 107. 108. 109. 110. 111. 112. 113. 114. 115. 116. 117. 118. 119. 120. 121. 122. 123. 124. 125. 126. 127. 128. 129. 130. 131. 132. 133. 134. 135. 136. 137. 138. 139. 140. 141. 142. 143. 144. 145. 146. 147. 148. 149. 150. 151. 152. 153. 154. 155. 156. 157. 158. 159. 160. 161. 162. 163. 164. 165. 166. 167. 168. 169. 170. 171. 172. 173. 174. 175. 176. 177. 178. 179. 180. 181. 182. 183. 184. 185. 186. 187. 188. 189. 190. 191. 192. 193. 194. 195. 196. 197. 198. 199. 200. 201. 202. 203. 204. 205. 206. 207. 208. 209. 210. 211. 212. 213. 214. 215. 216. 217. 218. 219. 220. 221. 222. 223. 224. 225. 226. 227. 228. 229. 230. 231. 232. 233. 234. 235. 236. 237. 238. 239. 240. 241. 242. 243. 244. 245. 246. 247. 248. 249. 250. 251. 252. 253. 254. 255. 256. 257. 258. 259. 260. 261. 262. 263. 264. 265. 266. 267. 268. 269. 270. 271. 272. 273. 274. 275. 276. 277. 278. 279. 280. 281. 282. 283. 284. 285. 286. 287. 288. 289. 290. 291. 292. 293. 294. 295. 296. 297. 298. 299. 300. 301. 302. 303. 304. 305. 306. 307. 308. 309. 310. 311. 312. 313. 314. 315. 316. 317. 318. 319. 320. 321. 322. 323. 324. 325. 326. 327. 328. 329. 330. 331. 332. 333. 334. 335. 336. 337. 338. 339. 340. 341. 342. 343. 344. 345. 346. 347. 348. 349. 350. 351. 352. 353. 354. 355. 356. 357. 358. 359. 360. 361. 362. 363. 364. 365. 366. 367. 368. 369. 370. 371. 372. 373. 374. 375. 376. 377. 378. 379. 380. 381. 382. 383. 384. 385. 386. 387. 388. 389. 390. 391. 392. 393. 394. 395. 396. 397. 398. 399. 400. 401. 402. 403. 404. 405. 406. 407. 408. 409. 410. 411. 412. 413. 414. 415. 416. 417. 418. 419. 420. 421. 422. 423. 424. 425. 426. 427. 428. 429. 430. 431. 432. 433. 434. 435. 436. 437. 438. 439. 440. 441. 442. 443. 444. 445. 446. 447. 448. 449. 450. 451. 452. 453. 454. 455. 456. 457. 458. 459. 460. 461. 462. 463. 464. 465. 466. 467. 468. 469. 470. 471. 472. 473. 474. 475. 476. 477. 478. 479. 480. 481. 482. 483. 484. 485. 486. 487. 488. 489. 490. 491. 492. 493. 494. 495. 496. 497. 498. 499. 500. 501. 502. 503. 504. 505. 506. 507. 508. 509. 510. 511. 512. 513. 514. 515. 516. 517. 518. 519. 520. 521. 522. 523. 524. 525. 526. 527. 528. 529. 530. 531. 532. 533. 534. 535. 536. 537. 538. 539. 540. 541. 542. 543. 544. 545. 546. 547. 548. 549. 550. 551. 552. 553. 554. 555. 556. 557. 558. 559. 560. 561. 562. 563. 564. 565. 566. 567. 568. 569. 570. 571. 572. 573. 574. 575. 576. 577. 578. 579. 580. 581. 582. 583. 584. 585. 586. 587. 588. 589. 590. 591. 592. 593. 594. 595. 596. 597. 598. 599. 600. 601. 602. 603. 604. 605. 606. 607. 608. 609. 610. 611. 612. 613. 614. 615. 616. 617. 618. 619. 620. 621. 622. 623. 624. 625. 626. 627. 628. 629. 630. 631. 632. 633. 634. 635. 636. 637. 638. 639. 640. 641. 642. 643. 644. 645. 646. 647. 648. 649. 650. 651. 652. 653. 654. 655. 656. 657. 658. 659. 660. 661. 662. 663. 664. 665. 666. 667. 668. 669. 670. 671. 672. 673. 674. 675. 676. 677. 678. 679. 680. 681. 682. 683. 684. 685. 686. 687. 688. 689. 690. 691. 692. 693. 694. 695. 696. 697. 698. 699. 700. 701. 702. 703. 704. 705. 706. 707. 708. 709. 710. 711. 712. 713. 714. 715. 716. 717. 718. 719. 720. 721. 722. 723. 724. 725. 726. 727. 728. 729. 730. 731. 732. 733. 734. 735. 736. 737. 738. 739. 740. 741. 742. 743. 744. 745. 746. 747. 748. 749. 750. 751. 752. 753. 754. 755. 756. 757. 758. 759. 760. 761. 762. 763. 764. 765. 766. 767. 768. 769. 770. 771. 772. 773. 774. 775. 776. 777. 778. 779. 780. 781. 782. 783. 784. 785. 786. 787. 788. 789. 790. 791. 792. 793. 794. 795. 796. 797. 798. 799. 800. 801. 802. 803. 804. 805. 806. 807. 808. 809. 810. 811. 812. 813. 814. 815. 816. 817. 818. 819. 820. 821. 822. 823. 824. 825. 826. 827. 828. 829. 830. 831. 832. 833. 834. 835. 836. 837. 838. 839. 840. 84

y. ~~NRG~~

Having 3 evaluation metrics makes it harder for you to quickly choose between 2 different algorithms and will slow down the speed with which you team can iterate

⇒ solution:

define which criterion is most important then set thresholds for the other two



stakeholders must define thresholds for satisficing metrics

and must leave the optimizing metric unbounded

addressing data mismatch

1/ carry out ~~the~~ manual error analysis to try to understand the differences between training and dev/test sets

to avoid overfitting the dev set: we should manually examining only the dev set.

2/ make training data more similar or collect more data similar to dev/test sets

i/ collect more data

ii/ artificial data synthesis

⚠ if we have

10000 hours of speech

1 hour of noise

if we repeat the noise 10000 times

⇒ we may overfit to the noise

the synthetic data can help to train the model

to get better performance at the dev set

but shouldn't be added to the dev or test sets because

they don't represent our target in a completely accurate way

choose architecture
(model, data, etc)



$$n_{cv} = 500$$

the algorithm misclassifies 100

manually examine them

machine learning project

How to add data?

- 1/ add more data of everything
- 2/ add more data of the types where error analysis has indicated it might help.
- 3/ data augmentation (images - sounds)
- 4/ create new synthesis examples
- 5/ transfer learning: when we don't have a lot of data

example: you want to predict numbers 0...9

option 1: only train output layers algorithm
train a model to predict ~~with~~ a lot of types of data

(cats - dogs - people) ...) \rightarrow supervised training

take the parameters of the model

apply the model for the bills

(change the parameters of the final layer)

~~adjust~~

\nearrow fine tuning

option 2: train all parameters and the parameters of the hidden layers will be initialized with the old values

steps of transfer learning

1/ download neural network parameters train on a large dataset ~~train~~ with same input type as your application

2/ further train the network on your own data

~~2~~

examples: cat classifier

30% accuracy 10% error

if the algorithm misclassifies dogs
and puts them into cats

⇒ add more dog images

add features specific to dogs

but should you do make your cat classifier
better on dogs?

error analysis:

get ~100 misclassified dev set examples

count up how many are dogs

if 5% are dogs:

⇒ "ceiling"

if 50% are dogs

⇒ more optimistic
to spend time on
dogs

2/ evaluate multiple ideas in parallel

ideas for cat detection algorithm

1/ fix dogs pictures being recognized as cats

2/ fix great cats (Reno - jankos, ...)

being misrecognized

3/ improve performance on blurry images

| Image | idea 1 | idea 2 | blurry |
|---------|--------|--------|--------|
| 1 | ✓ | | |
| 2 | | | ✓ |
| ... | | ✓ | ✓ |
| % total | 8% | 43% | 65% |

cleaning incorrectly labeled data

1/ training set

deep learning algorithms are quite robust to random
errors in the training set

⇒ if the errors are reasonably random, don't pay
attention to them

but they are sensitive to systematic errors

⇒ if the errors are systematic, ~~missed~~ (classify all
while dogs are annotated cats)

⇒ we need to fix it

2/ dev set and test set

in the error analysis, we add a column called
incorrectly labeled

example 1

overall dev set error: 10%

errors due to incorrect labels: 6% of the errors

⇒ 0.6% of all data

errors due to other causes: 3.4% of all data

in this case: errors due to incorrect labels
is a small fraction

example 2

overall error: 2%

errors due to incorrect labels: 30% of the errors

⇒ 0.6% of all data

errors due to other causes: 1.4% of all data
(70% of the errors)

⇒ in this case it is better to fix the incorrect labels

How to avoid leaks

- 1/ apply some process to your dev and test sets to make sure that they come from the same distribution
- 2/ consider examining example your algorithm got right as well as ones it got wrong
(it isn't always done)
- 3/ train and dev/test data may now come from slightly different distributions

we want a mobile app that classifies training and testing on different

we have: mobile pictures distributions

~~we don't want~~ pictures from web-pages

~~we don't want~~ pictures from mobile app (less quality)

→ n

≈ 50 000 mobile pictures

≈ 200 000 web-pictures

we want our algorithm to work on mobile images

we don't want only use the mobile pictures
(small dataset)

option 1: put the 2 datasets together

shuffle them into train/dev/test set

advantages: train/dev/test sets come from the same distribution

disadvantage: the dev set: most of the images will come from web-pictures which is not representative of the user case

⇒ this option is not suitable

option 2:

train: 200 K web + 5 K mobile

dev: 2,5 K mobile pictures

test: 2,5 K mobile pictures

advantage: we aim the objective of the app

advantage: the distributions are different

⇒ this option is suitable

II - bias and variance

cat classifier example:

average humans get 20% error

training error: 1%

dev error: 10%

| | |
|--------------------|----------------------|
| Human error | ↓ bias |
| training error | ↓ variance |
| training dev error | ↓ data mismatch |
| dev error | ↓ overfitting to dev |

① if dev data came from the same distribution as the training set:

⇒ we have a variance problem

② if they don't come from the same distribution

⇒ we can't have this interpretation

(dev set may contain harder examples)

(we may also have variance but not only)



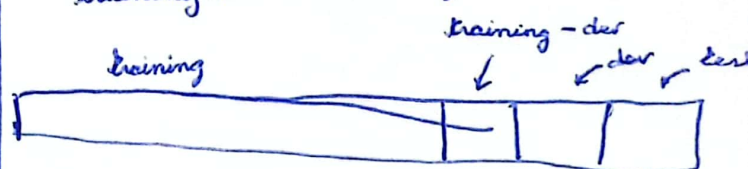
How to determine the cause?

variance or different distributions?

also we define another set:

training - dev set: same distribution as

training set but not used for training



1 - train on training

training error: 1%

training dev error: 9%

dev error: 10%

⇒ Here we can affirm that we have a variance problem.

2 - training error: 1%

training dev error: 1.5%

dev error: 10%

⇒ Here we have a data mismatch error

3 - Human error $\approx 0\%$
Training error: 10%
Training dev error: 11%
dev error: 12%

\Rightarrow bias problem

4 - Human error $\approx 0\%$
Training error: 10%
Training dev error: 11%
dev error: 20%

\Rightarrow bias problem

data mismatch error

5 - Human error: 4%
Training error: 7%
Training dev error: 10%
dev error: 6%
test error: 6%

\Rightarrow Here we can say that training set is much
harder than the test

~~Human error~~