یادگیری عمیق

مدرس: محمدرضا محمدی

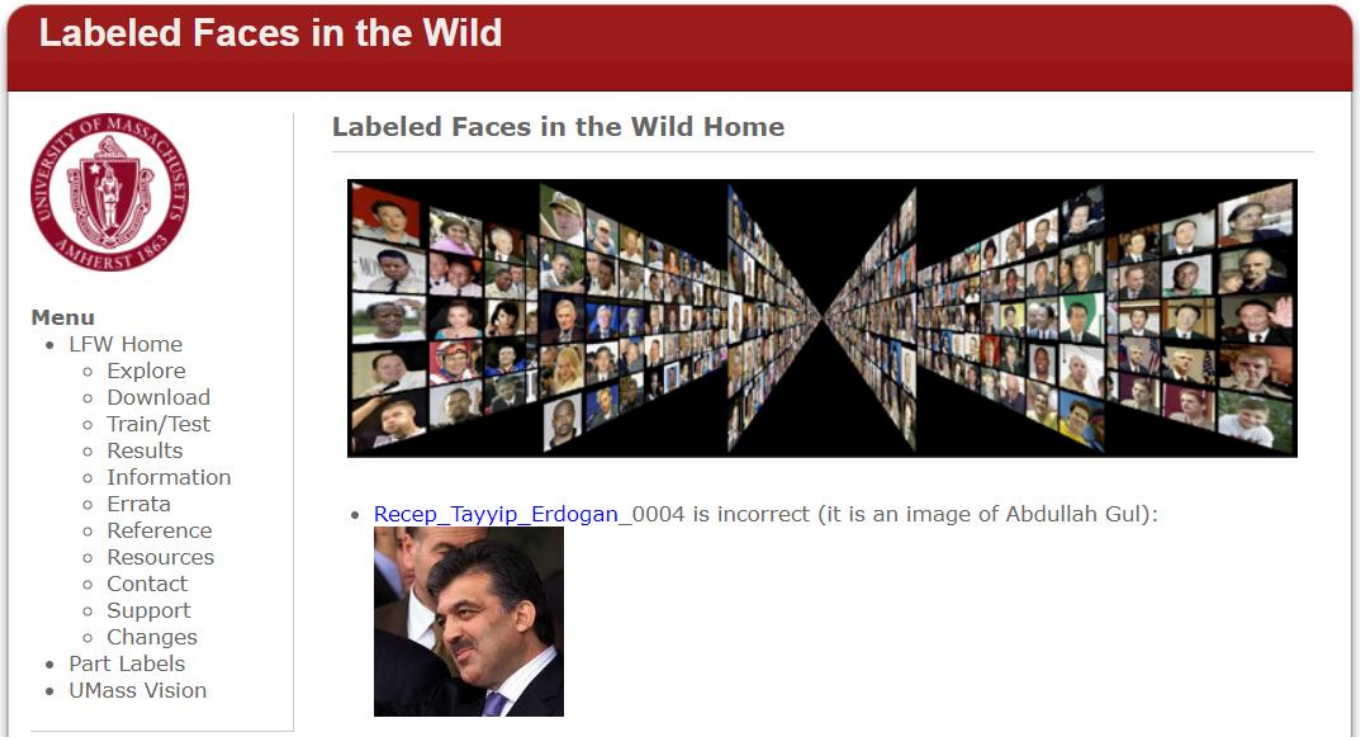
زمستان ۱۴۰۱

منظم‌سازی

Regularization

افزودن نویز در خروجی مطلوب

- اکثر مجموعه‌های داده مقداری اشتباه در برچسب‌های y دارند
- ماکزیمم کردن $\log p(y|x)$ زمانیکه y اشتباه است می‌تواند خیلی مضر باشد



افزودن نویز در خروجی مطلوب

- اکثر مجموعه‌های داده مقداری اشتباه در برچسب‌های y دارند
- ماکزیمم کردن $\log p(y|x)$ زمانیکه y اشتباه است می‌تواند خیلی مضر باشد
- می‌توانیم فرض کنیم برچسب موجود در مجموعه داده با احتمال $1-\epsilon$ درست است که ϵ یک عدد کوچک است
- با استفاده از Label Smoothing، بجای آنکه خروجی مطلوب برای دسته‌بند را مقادیر سخت ۰ و ۱ قرار دهیم، از مقادیر نرم شده $\epsilon/(k-1)$ و $1-\epsilon$ استفاده می‌کنیم

1.0	0.0	0.0

0.90	0.05	0.05

- از این مقادیر در تابع ضرر cross entropy استاندارد استفاده می‌کنیم

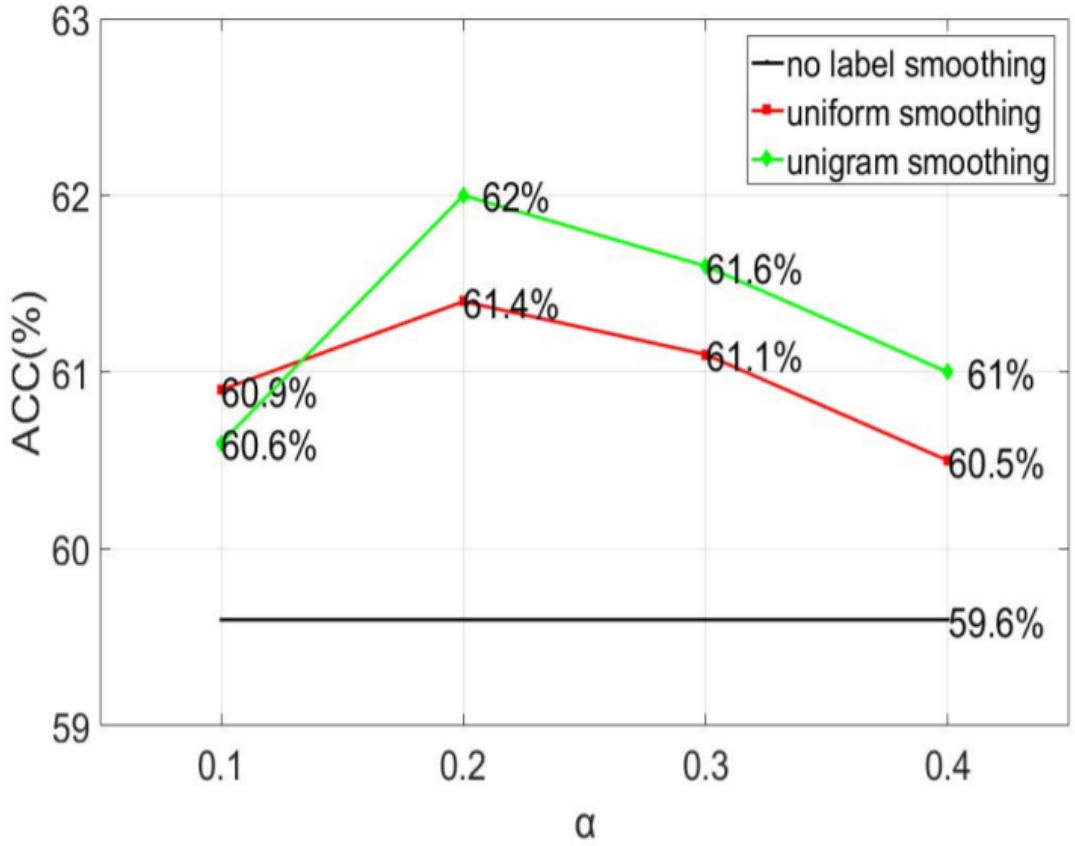
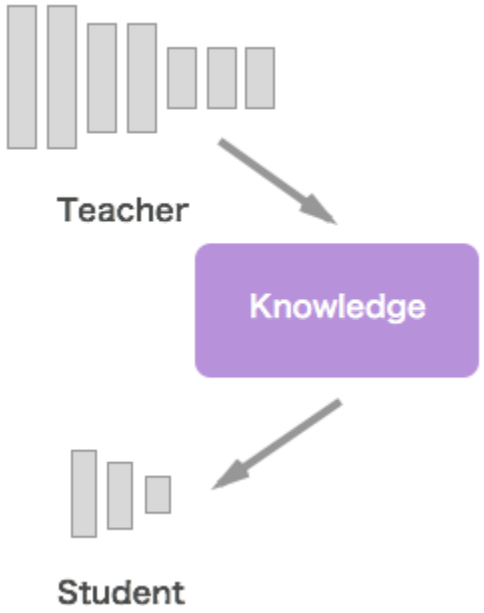


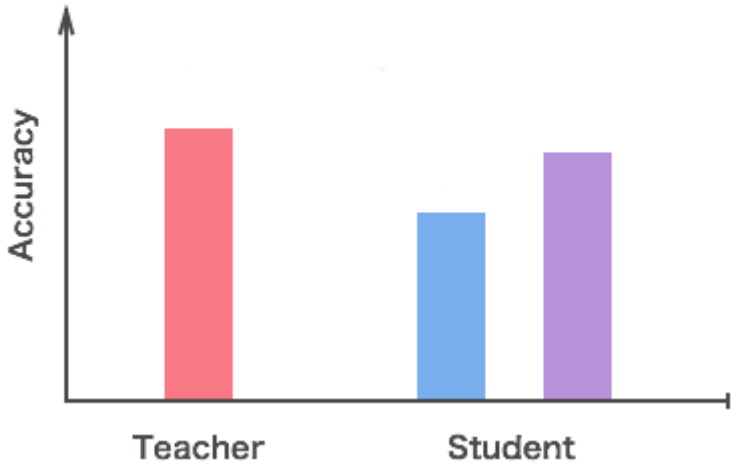
Figure 5: *The performance of different label smoothing methods.*

Huang, Jian, et al. "Learning Utterance-Level Representations with Label Smoothing for Speech Emotion Recognition." *INTERSPEECH*. 2020.

تقطیر دانش (Knowledge Distillation)

- در یادگیری ماشین، تقطیر دانش به فرآیندی گفته می‌شود که دانش از یک مدل بزرگ‌تر (معلم) به یک مدل کوچک‌تر (دانش‌آموز) منتقل می‌شود
  - یکی از کاربردهای آن توسعه مدل‌های سریع با دقت مناسب است

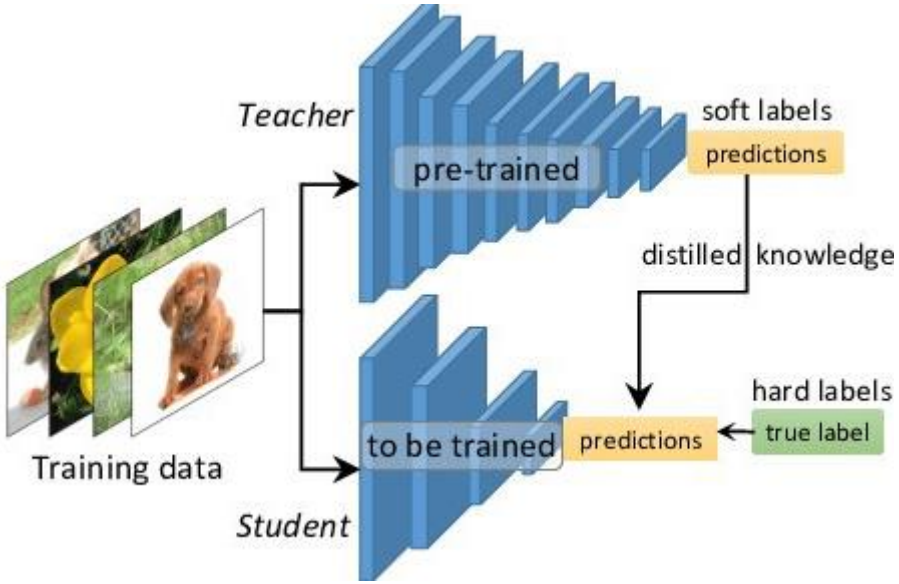




https://medium.com/neuralmachine/knowledge-distillation-dc241d7c2322

تقطیر دانش

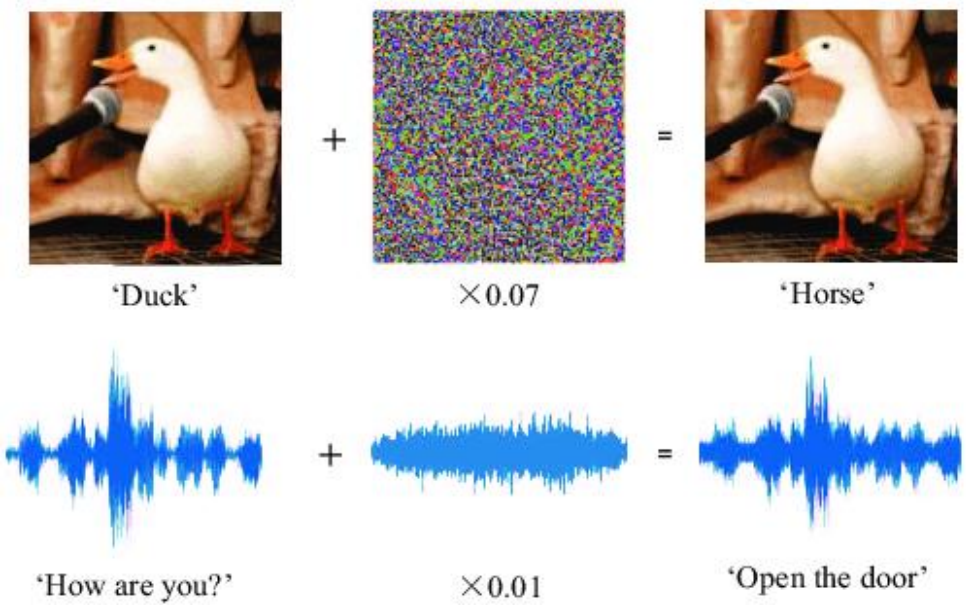
- روش‌های مختلفی برای تقطیر دانش پیشنهاد شده است که یکی از آنها تقطیر در سطح پاسخ یا خروجی شبکه است
- از پیش‌بینی شبکه معلم برای نظارت بر پاسخ شبکه دانش‌آموز به عنوان برچسب‌های نرم استفاده می‌شود

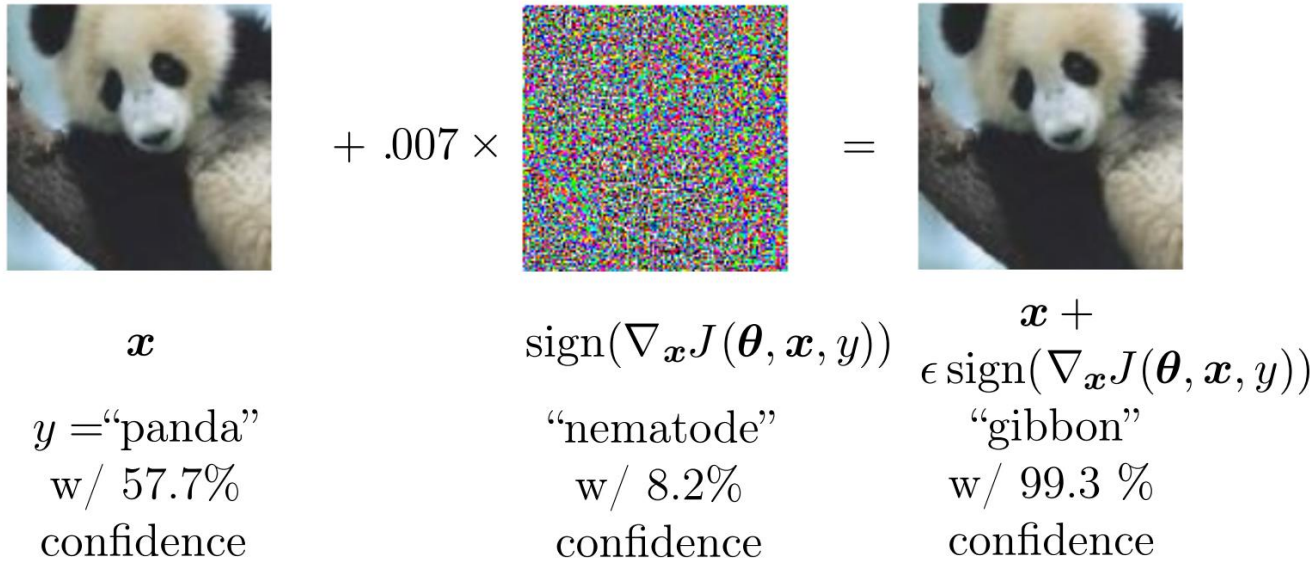


https://towardsdatascience.com/knowledge-distillation-simplified-dd4973dbc764

آموزش تخاصمی (Adversarial Training)

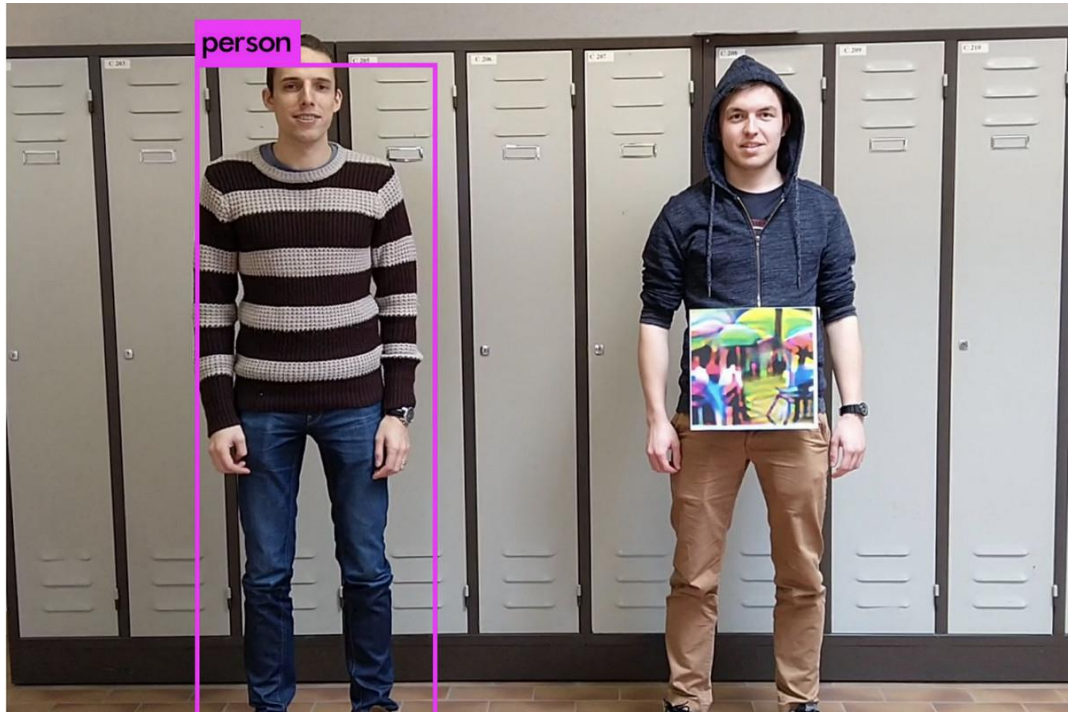
- در بسیاری از موارد، شبکه‌های عمیق توانسته‌اند به کارآیی در حد انسان دست بیابند
- به منظور بررسی سطح درک یک شبکه از مسئله مورد نظر، می‌توانیم نمونه‌هایی را بررسی کنیم که مدل آنها را نادرست دسته‌بندی کرده است





آموزش تخاصمی

- می‌توانیم نرخ خطا روی مجموعه تست را با استفاده از یادگیری تخاصمی کاهش دهیم
  - آموزش بر روی نمونه‌هایی از مجموعه آموزشی که به صورت تخاصمی ساخته شده‌اند

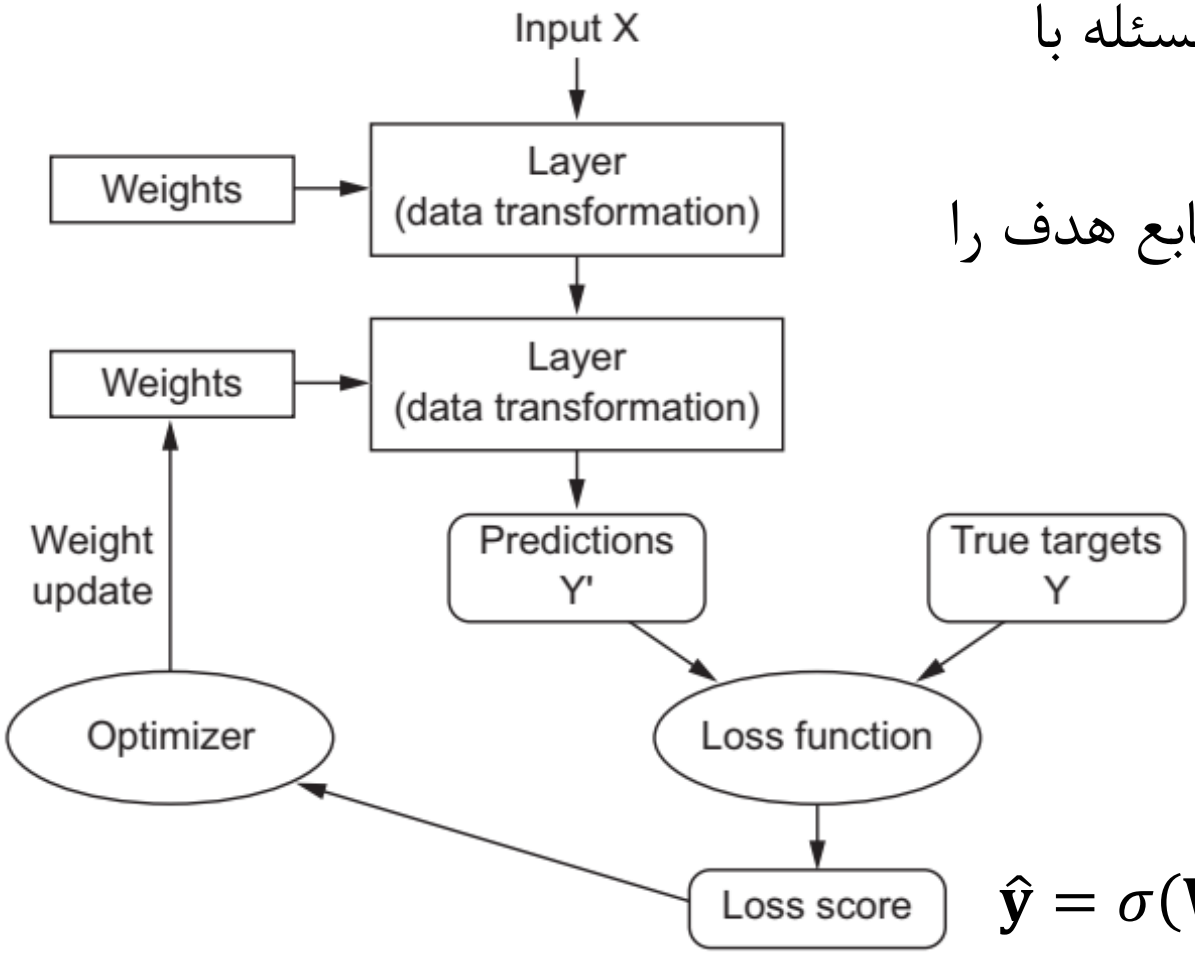


الگوریتم‌های بهینه‌سازی

Optimization Algorithms

بهینه‌سازی

- در یادگیری ماشین، به دنبال مدل مناسبی برای حل مسئله با مقداری محدودی از داده هستیم
- هدف از بهینه‌سازی دستیابی به پارامترهایی است که تابع هدف را به بهترین مقدار خود برساند



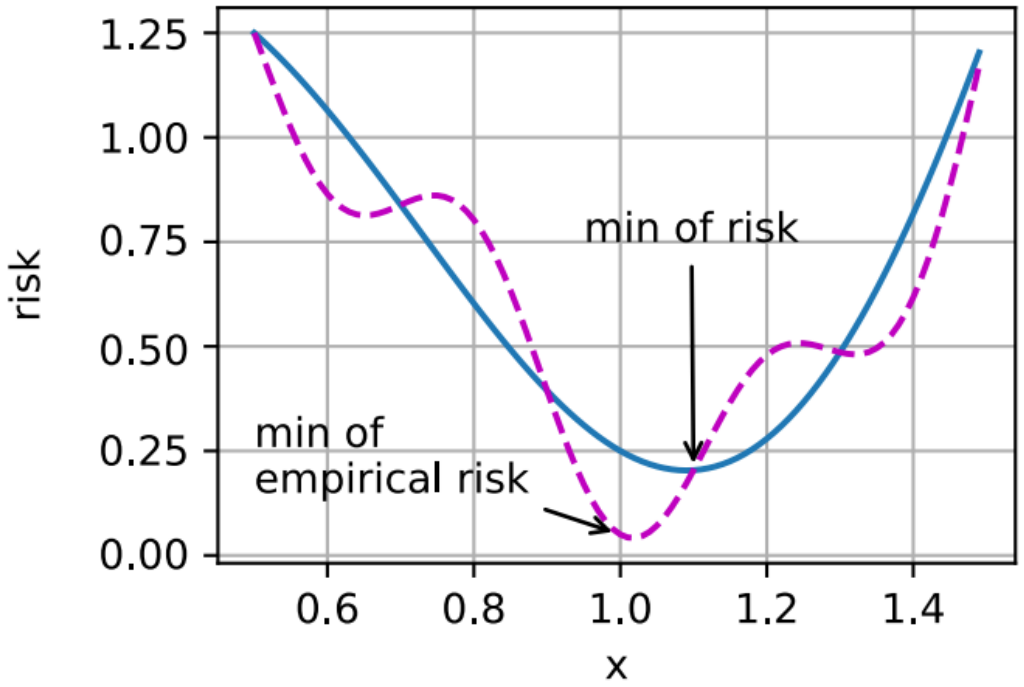
$$\hat{\mathbf{y}} = \sigma(\mathbf{W}_2\sigma(\mathbf{W}_1\mathbf{x} + \mathbf{b}_1) + \mathbf{b}_2)$$

خطای تجربی

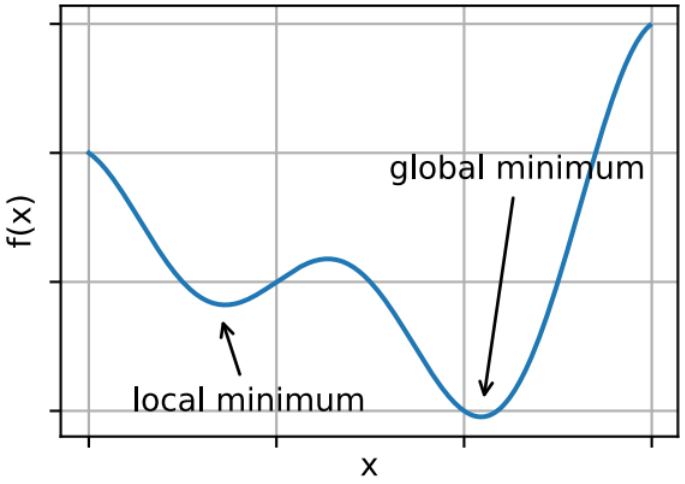
- خطای تجربی برابر با میانگین تابع ضرر برای داده‌های آموزشی است
- خطا برابر با امید ریاضی تابع ضرر است

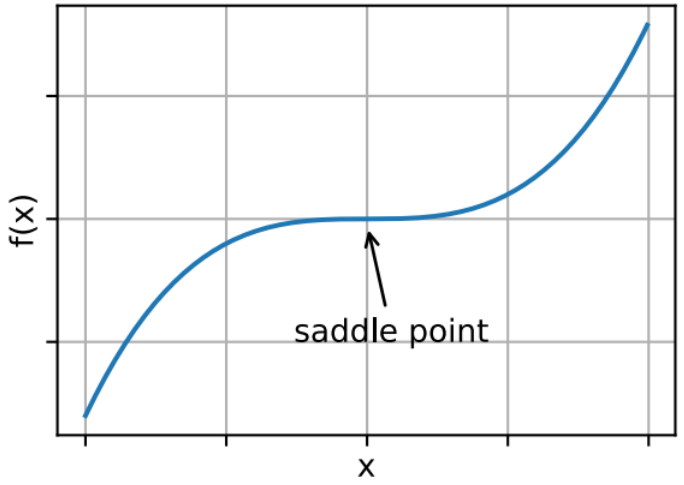
$$\min_f \frac{1}{n}\sum_{i=1}^{n} l(f(\mathbf{x}_i), y_i)$$

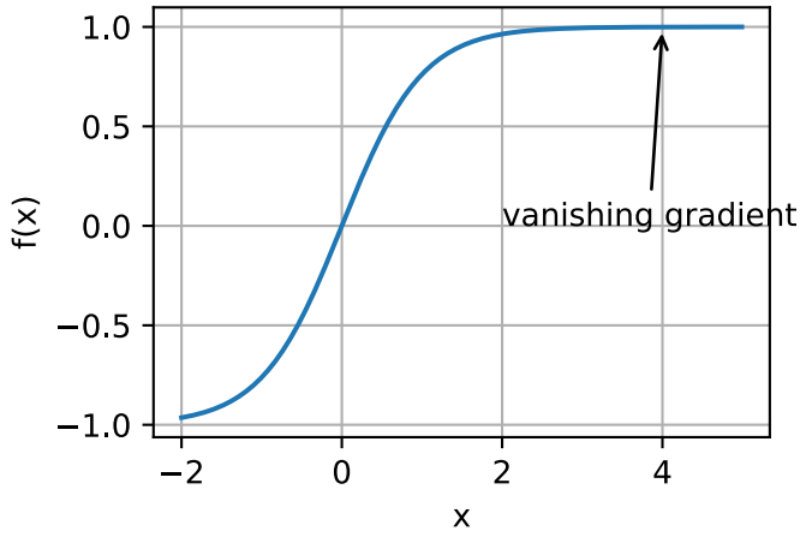
$$E_{p(\mathbf{x},y)}[l(f(\mathbf{x}), y)] = \iint l(f(\mathbf{x}), y)p(\mathbf{x}, y)d\mathbf{x}dy$$



چالش‌های بهینه‌سازی در یادگیری عمیق







- بهینه محلّی
  - امکان همگرا شدن به بهینه محلّی در روش‌های مبتنی گرادیان زیاد است
  - حتی در نقاط زینی هم گرادیان صفر است
- محوشدگی گرادیان
  - برخی توابع باعث می‌شوند گرادیان نزدیک به صفر شود و بهینه‌سازی ممکن است بسیار کند شود

رویکرد ۱: جستجوی تصادفی

```
# assume X_train is the data where each column is an example (e.g. 3073 x 50,000)
# assume Y_train are the labels (e.g. 1D array of 50,000)
# assume the function L evaluates the loss function

bestloss = float("inf") # Python assigns the highest possible float value
for num in range(1000):
  W = np.random.randn(10, 3073) * 0.0001 # generate random parameters
  loss = L(X_train, Y_train, W) # get the loss over the entire training set
  if loss < bestloss: # keep track of the best solution
    bestloss = loss
    bestW = W
  print 'in attempt %d the loss was %f, best %f' % (num, loss, bestloss)

# prints:
# in attempt 0 the loss was 9.401632, best 9.401632
# in attempt 1 the loss was 8.959668, best 8.959668
# in attempt 2 the loss was 9.044034, best 8.959668
# in attempt 3 the loss was 9.278948, best 8.959668
# in attempt 4 the loss was 8.857370, best 8.857370
# in attempt 5 the loss was 8.943151, best 8.857370
# in attempt 6 the loss was 8.605604, best 8.605604
# ... (trunctated: continues for 1000 lines)
```

- پاسخ بسیار ضعیف است!
- دقت نهایی تنها ۱۵.۵٪ است
- نتایج روش‌های جدید بالای ۹۹٪ است

```
# Assume X_test is [3073 x 10000], Y_test [10000 x 1]
scores = Wbest.dot(Xte_cols) # 10 x 10000, the class scores for all test examples
# find the index with max score in each column (the predicted class)
Yte_predict = np.argmax(scores, axis = 0)
# and calculate accuracy (fraction of predictions that are correct)
np.mean(Yte_predict == Yte)
# returns 0.1555
```

رویکرد ۲: جستجوی محلی تصادفی

- در هر مرحله بهترین وزن‌ها را ذخیره می‌کنیم و جستجو را با یک گام محدود در اطراف آن انجام می‌دهیم
- با استفاده از این تغییر ساده، دقت بر روی داده‌های آزمون به ۲۱.۴٪ افزایش می‌یابد
- همچنان دقت خیلی پائین است!

```
W = np.random.randn(10, 3073) * 0.001 # generate random starting W
bestloss = float("inf")
for i in range(1000):
  step_size = 0.0001
  Wtry = W + np.random.randn(10, 3073) * step_size
  loss = L(Xtr_cols, Ytr, Wtry)
  if loss < bestloss:
    W = Wtry
    bestloss = loss
  print 'iter %d loss is %f' % (i, bestloss)
```

رویکرد ۳: حرکت در مسیر شیب

- تقریب مرتبه اول یک تابع یک‌بعدی به صورت زیر تعریف می‌شود:

$$f(x+\epsilon) = f(x) + \epsilon f'(x) + \mathcal{O}(\epsilon^2)$$

- اگر $\epsilon = -\eta f'(x)$ با $\eta > 0$ انتخاب شود

$$f\left(x - \eta f'(x)\right) = f(x) - \eta f'^2(x) + \mathcal{O}\left(\eta^2 f'^2(x)\right)$$

- اگر $f'(x) \neq 0$ باشد و η کوچک باشد:

$$f\left(x - \eta f'(x)\right) \lesssim f(x)$$

$$x \leftarrow x - \eta f'(x)$$