مدرس: دکتر فدایی و دکتر یعقوب‌زاده
طراح: آرش هاتفی - حمید خدادادی - سینا نگارنده

Dataloader

از این کلاس جهت آماده‌سازی داده‌های ورودی استفاده میشود. این کلاس در Constructor خود، برداری از داده‌ها (data)، برچسب‌های متناظر (labels)، تعداد کلاس‌ها (n_classes)، اندازه‌ی batch موردنظر (batch_size) و بُر خوردن یا نخوردن داده‌ها (shuffle) را میگیرد.

Args	Description
data	داده‌ها
labels	برچسب‌های متناظر
n_classes	تعداد کلاس‌ها
batch_size	اندازه‌ی batch
shuffle	بُر خوردن یا نخوردن داده‌ها

Methods

---

onehot

از برچسب‌های ورودی، بردارهای onehot می‌سازد. این تابع برچسب‌ها و تعداد کلاس‌ها را به عنوان ورودی دریافت می‌نماید.

Args	Description
labels	برچسب‌ها
n_classes	تعداد کلاس‌ها

Returns	Description
onehot_vectors	بردارهای برچسب‌های onehot شده

shuffle_dataset

همزمان داده‌ها و برچسب‌هایشان را بُر می‌زند.

Args	Description
labels	برچسب‌های داده‌ها

__iter__

برای گرفتن batchها استفاده می‌شود.

Returns	Description
Batch	با توجه به batch_size، batch را باز می‌گرداند

Activation Functions

قالب کلی برای توابع فعالساز مورد استفاده در شبکه‌های عصبی.

تنها برای LeakyRelu داریم:

Args	Description
negative_slope	مقدار پارامتر آلفا برای شیب منفی

Methods

---

val

مقدار تابع را به ازای یک ورودی خاص محاسبه می‌کند.

Args	Description
matrix	بردار ورودی تابع فعالساز

Returns	Description
val	مقدار تابع

---

derivative

مشتق تابع را به ازای یک ورودی خاص محاسبه می‌کند.

Args	Description
matrix	بردار ورودی مشتق تابع فعالساز

Returns	Description
derivative_val	مقدار مشتق تابع

سایر متدهای موجود جهت آسانتر کردن کاربری کلاس‌ها است.

__call__

متد موجود جهت آسان‌تر کردن کاربری کلاس است.

Args	Description
matrix	بردار ورودی تابع فعالساز

Returns	Description
val	مقدار تابع

نکته:

```
identical(x)
```

زمانی که instance را call می‌کنید به صورت خودکار متد "__call__" از کلاس call می‌شود که __val را باز می‌گرداند. به عبارت دیگر از کد بالا به جای کد پایین باید استفاده کنیم.

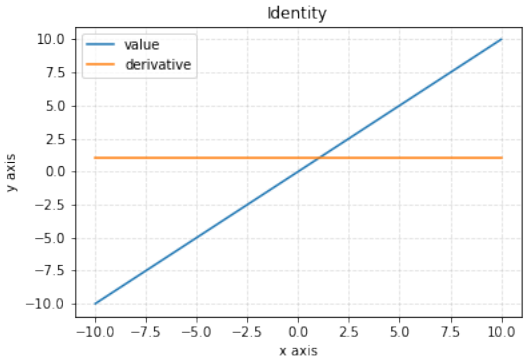
```
identical.val(x)
```

در ادامه به بررسی هر یک از توابع فعالساز می‌پردازیم. تابع فعالساز Softmax در Cross Entropy توضیح داده شده است.

Identity

از این تابع برای مدل سازی در زمانی که قصد استفاده از توابع فعال ساز را در یک لایه را نداریم، استفاده می‌کنیم.

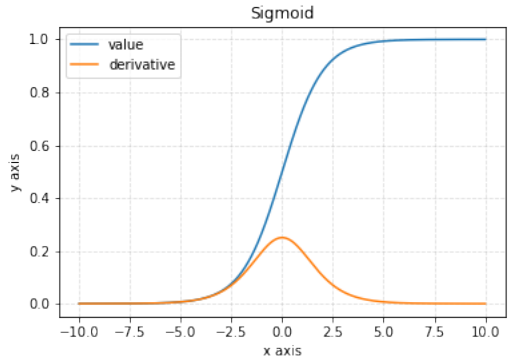
Plot



Function	Derivative	Range
$f(x) = x$	$f'(x) = 1$	$(-\infty, \infty)$

Sigmoid

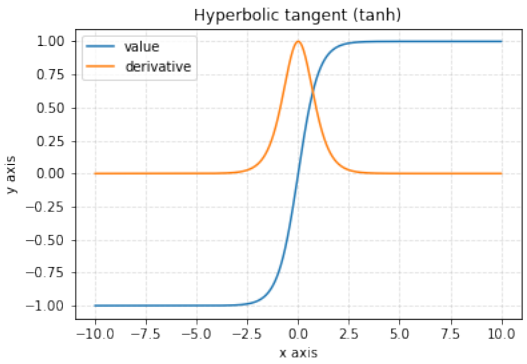
Plot



Function	Derivative	Range
$f(x) = \frac{1}{1+e^{-x}}$	$f'(x) = f(x)(1-f(x))$	(0,1)

Hyperbolic tangent (tanh)

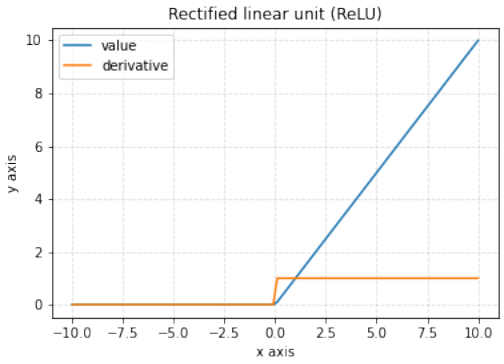
Plot



Function	Derivative	Range
$f(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}}$	$f'(x) = 1 - f(x)^2$	$(-1,1)$

Rectified linear unit (ReLU)

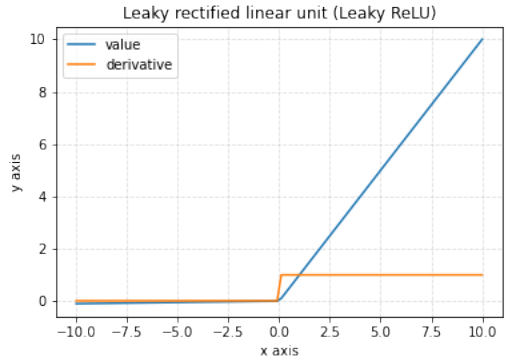
Plot



Function	Derivative	Range
$f(x) = \begin{cases} 0 & if\ x \leq 0 \\ x & if\ x > 0 \end{cases}$	$f'(x) = \begin{cases} 0 & if\ x < 0 \\ 1 & if\ x > 0 \\ undefined & if\ x = 0 \end{cases}$	$[0,\infty)$

Leaky rectified linear unit (Leaky ReLU)

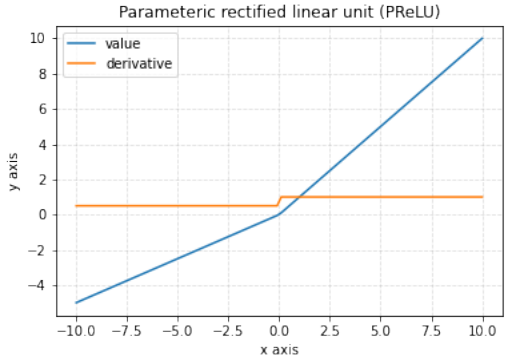
Plot



Function	Derivative	Range
$f(x) = \begin{cases} 0.01 \times x & if\ x < 0 \\ x & if\ x \geq 0 \end{cases}$	$f'(x) = \begin{cases} 0.01 & if\ x < 0 \\ 1 & if\ x \geq 0 \end{cases}$	$(-\infty, \infty)$

Parametric rectified linear unit (PReLU)

Plot



Function	Derivative	Range
$f(x) = \begin{cases} \alpha \times x & if\ x < 0 \\ x & if\ x \geq 0 \end{cases}$	$f'(x) = \begin{cases} \alpha & if\ x < 0 \\ 1 & if\ x \geq 0 \end{cases}$	$(-\infty, \infty)$

CrossEntropy

از این کلاس برای پیاده‌سازی تابع زیان (Loss function) استفاده می‌کنیم.

Args	Description
None	

Methods

---

val

مقدار تابع را به ازای یک ورودی خاص محاسبه می‌کند.

Args	Description
true_val	برچسب‌های خروجی شبکه
expected_val	برچسب‌های مجموعه داده‌ها

Returns	Description
cross_entropy_value	مقدار تابع

---

derivative

مشتق تابع را به ازای یک ورودی خاص محاسبه می‌کند.

Args	Description
true_val	برچسب‌های خروجی شبکه
expected_val	برچسب‌های مجموعه داده‌ها

Returns	Description
cross_entropy_derivative	مقدار مشتق تابع

سایر متدهای موجود جهت آسانتر کردن کاربری کلاس‌ها است.

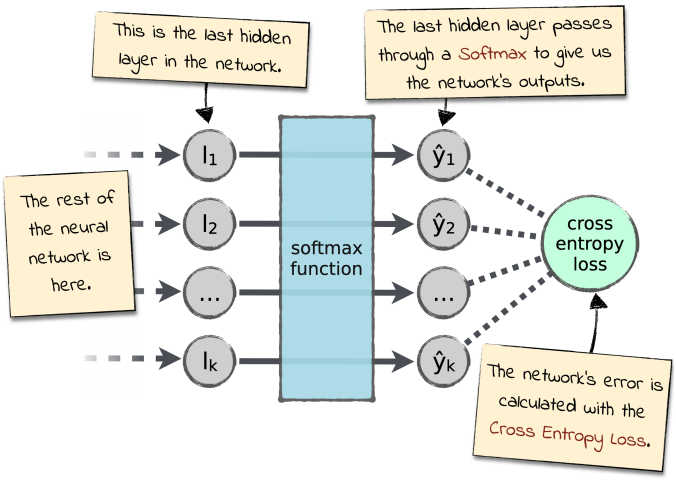
__call__

متد موجود جهت آسان‌تر کردن کاربری کلاس است.

Args	Description
true_val	برچسب‌های خروجی شبکه
expected_val	برچسب‌های مجموعه داده‌ها

Returns	Description
cross_entropy_value	مقدار خروجی تابع خطا

تابع هزینه (Cost function) یا تابع زیان (Loss function) در علم آمار و بهینه‌سازی تابعی است که مقدار ضرر را در یک پیشامد و در واقع میزان خطا در هر بار اجرای شبکه‌ی عصبی را برای داده‌های آموزشی نمایش می‌دهد. در یک مسئله بهینه‌سازی هدف مینیمم‌سازی تابع هزینه است و این کار معمولاً با الگوریتم‌های تخمینی انجام می‌شود. تابع زیان، معیاری برای سنجش مناسب بودن مدل از نظر قابلیت و توانایی در پیشگویی مقدارهای جدید است. یکی از روش‌های معمول برای پیدا کردن کمینه تابع زیان، استفاده از مشتق و الگوریتم «گرادیان کاهشی» (Gradient Descent) است. انتخاب تابع زیان مناسب به عوامل متعددی نظیر وجود نقاط یا داده‌های پرت، نوع الگوریتم یادگیری ماشین، هزینه زمانی اجرای الگوریتم و سادگی محاسبه مشتق و ... بستگی دارد.



تابع هزینه‌ی Cross Entropy در درون خود باید شامل تابع فعالساز Softmax نیز باشد. این به این معنا است که لازم نیست که در هنگام استفاده از این تابع هزینه، لایه‌ی آخر شبکه دارای تابع هزینه‌ی Softmax باشد. همچنین، مشتق این دو تابع نیز باید به صورت یکجا گرفته شود و استفاده شود. علت این شیوه‌ی پیاده‌سازی به ساده‌تر شدن فرم مشتق حاصل از قرار گرفتن این دو تابع در پشت‌هم و افزایش پایداری محاسبات شبکه مربوط است.

با توجه به توضیحات داده شده، باید خروجی لایه آخر شبکه (بدون تابع فعالساز) را به عنوان ورودی به Softmax دهید و از خروجی آن به عنوان $\hat{y}$ استفاده کنید.

برای پیاده‌سازی می‌توانید از عبارات زیر کمک بگیرید.

Softmax

Value

$$Softmax(x) = \frac{e^{x_i}}{\sum_{j=1}^{J} e^{x_j}} \quad for\ i = 1, \ldots, J$$

توجه: برای جلوگیری از overflow در محاسبه‌ی Softmax به ازای ورودی‌های بزرگ، از نسخه‌ی پایدار این تابع (Stable Softmax) استفاده کنید. (در نسخه‌ی پایدار قبل از محاسبه‌ی Softmax برای یک بردار، همه‌ی عناصر بردار را از یک مقدار ثابت کم می‌کنیم تا ماکسیمم درایه‌ی بردار کاهش یابد و overflow اتفاق نیفتد. این مقدار ثابت می‌تواند بزرگترین درایه‌ی بردار یا هر مقدار دیگری باشد.)

Cross Entropy

Value

$$CrossEntropy(y, \hat{y}) = -\sum_{i} y_i \times log(\hat{y}_i)$$

Derivative

$$CrossEntropy(y, \hat{y})' = \hat{y} - y$$

برای مطالعات بیشتر درباره نحوه محاسبه مشتق Cross Entropy می‌توانید به لینک مراجعه کنید.

Layer

از این کلاس جهت ایجاد کردن هر یک از لایه‌های شبکه‌ی عصبی استفاده می‌شود.

تابع فعالساز لایه (activation) به صورت پیشفرض identical (یا بدون activation function) است.

شیوه‌ی وزن‌دهی اولیه (شامل 2 متد وزن دهی uniform یا normal) و پارامترهای مرتبط با آن. (شیوه‌ی وزن‌دهی مطلوب مخاطب به وسیله‌ی یک رشته در ورودی مشخص می‌شود.)

Args	Description
input_size	اندازه‌ی بردار ورودی به لایه
output_size	اندازه‌ی بردار خروجی از لایه
activation	تابع فعالساز لایه - به صورت پیشفرض identical (بدون activation function) است.
initial_weight	شیوه‌ی وزن‌دهی اولیه

Methods

---

forward

به ازای یک ورودی خاص، خروجی لایه را محاسبه می‌کند.

در هر بار صدا شدن این متد، مقادیر ورودی لایه، ورودی تابع فعالساز، مشتق تابع فعالساز نسبت به ورودی و خروجی لایه ذخیره‌سازی می‌شوند تا از آنها در فرایند backpropagation جهت آپدیت کردن وزنهای شبکه استفاده شود.

Args	Description
layer_input	ورودی لایه

Returns	Description
self.__last_activation_output	خروجی لایه

update_weights

به‌روزرسانی وزن‌های لایه را با توجه به جریان گرادیان به‌روزرسانی می‌کند.

این تابع در ورودی خود گرادیان محاسبه شده را از لایه‌های بعدی و learning Rate را دریافت می‌نماید.

این تابع شامل آپدیت کردن بایاس هم می‌باشد.

Args	Description
backprop_tensor	گرادیان محاسبه شده در لایه بعدی
lr	مقدار learning Rate

FeedForwardNN

از این کلاس برای پیاده‌سازی شبکه‌های feed forward با معماری دلخواه استفاده می‌نماییم.

این کلاس در constructor خود سایز ورودی شبکه را دریافت مینماید.

Args	Description
input_shape	سایز ورودی شبکه

این کلاس دارای دو دسته متد است: از متدهای دسته‌ی اول برای ساختن شبکه و از متدهای دسته‌ی دوم برای آموزش شبکه استفاده میشود.

Methods

متدهای دسته‌ی اول به شرح زیر هستند:

---

add_layer

جهت تشکیل معماری شبکه عصبی، از متد add_layer استفاده می‌شود. از این متد می‌توان برای اضافه کردن یک لایه‌ی جدید به انتهای شبکه استفاده کرد.

این متد، پارامترهای موردنیاز جهت ساخت لایه‌ی جدید ازجمله تعداد نورون‌ها، تابع فعالساز و .. را به عنوان آرگومان ورودی دریافت می‌کند.

Args	Description
n_neurons	تعداد نورون‌ها
activation	تابع فعالساز
initial_weight	شیوه‌ی وزن‌دهی اولیه

---

set_training_param

پارامترهای آموزش (از جمله Loss Function و Leanring Rate) را تعیین می‌کند.

Args	Description
loss	تابع خطا
lr	مقدار Leanring Rate

forward

به ازای یک ورودی دلخواه، برای محاسبه‌ی خروجی شبکه ایجاد شده استفاده می‌شود.

Args	Description
network_input	ورودی شبکه

Returns	Description
network_output	خروجی شبکه

متدهای دسته‌ی دوم به شرح زیرند:

fit

شبکه را آموزش می‌دهد.

Args	Description
epochs	تعداد ایپاک‌های آموزش
trainloader	dataloader داده‌های آموزش
testloader	dataloader داده‌های تست (در صورت تمایل)
print_results	درصورتی که print_result فعال باشد، بعد از هر epoch آموزش، مقادیر دقت شبکه چاپ می‌شود.

Returns	Description
log	یک log از فرایند آموزش شبکه در قالب یک dictionary

__train

شبکه را بر روی یک dataloader از مجموعه داده‌های train، (به طول یک اپاک) آموزش می‌دهد.

Args	Description
trainloader	یک dataloader از مجموعه داده‌های train

Returns	Description
np.mean(bach_accuracies)	میانگین مقدارهای batch_accuracy
np.mean(batch_losses)	میانگین مقدارهای batch_average_loss

__test

شبکه را بر روی یک dataloader از مجموعه داده‌های test، (به طول یک اپاک) تست می‌کند.

Args	Description
testloader	یک dataloader از مجموعه داده‌های test

Returns	Description
np.mean(bach_accuracies)	میانگین مقدارهای batch_accuracy
np.mean(batch_losses)	میانگین مقدارهای batch_average_loss

__train_on_batch

شبکه را روی داده‌های یک batch از مجموعه داده‌های train، آموزش می‌دهد.

Args	Description
x_batch	داده‌های یک batch از مجوعه داده train
y_batch	برچسب‌های یک batch از مجوعه داده train

Returns	Description
(batch_accuracy, batch_average_loss)	یک tuple از مقدار batch_accuracy و batch_average_loss

__test_on_batch

شبکه را بر روی داده‌های یک batch از مجموعه داده‌های test، تست می‌کند.

Args	Description
x_batch	داده‌های یک batch از مجوعه داده test
y_batch	برچسب‌های یک batch از مجوعه داده test

Returns	Description
(batch_accuracy, batch_average_loss)	یک tuple از مقدار batch_accuracy و batch_average_loss

__get_labels

برای گرفتن برچسب بردارهای خروجی استفاده می‌شود.

Args	Description
outputs	بردارهای خروجی شبکه

Returns	Description
labels	برچسب‌های تولید شده از بردارهای خروجی شبکه

__compute_accuracy

accuracy شبکه با توجه به یک خروجی واقعی و مقدار مورد انتظار آن در خروجی محاسبه می‌کند.

Args	Description
output	خروجی واقعی
expected_output	مقدار مورد انتظار

Returns	Description
accuracy	مقدار accuracy شبکه

__update_weights

وزن لایه‌های شبکه را به توجه به ورودی و خروجی ذخیره شده در آنها آپدیت می‌کند.

این متد، مقدار واقعی شبکه به ازای یک batch از مجموعه داده آموزش و مقدار مورد انتظار در خروجی را دریافت می‌کند.

Args	Description
output	برچسب‌های خروجی شبکه
y_train	برچسب‌های یک batch از مجوعه داده train

Returns	Description
None	

منابع:

https://en.wikipedia.org/wiki/Activation_function

https://levelup.gitconnected.com/killer-combo-softmax-and-cross-entropy-5907442f60ba