1)

تصویر اول:

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 13 | 13 | 13 | 13 | 13 | 13 | 13 | 22 | 22 | 22 |
| 13 | 13 | 13 | 13 | 13 | 13 | 13 | 22 | 22 | 22 |
| 13 | 13 | 13 | 13 | 13 | 13 | 13 | 22 | 22 | 22 |
| 13 | 13 | 13 | 13 | 13 | 13 | 13 | 22 | 22 | 22 |
| 13 | 13 | 13 | 13 | 13 | 13 | 13 | 22 | 22 | 22 |
| 13 | 13 | 13 | 13 | 13 | 13 | 13 | 22 | 22 | 22 |
| 13 | 13 | 13 | 13 | 13 | 13 | 13 | 22 | 22 | 22 |
| 13 | 13 | 13 | 13 | 13 | 13 | 13 | 22 | 22 | 22 |
| 13 | 13 | 13 | 13 | 13 | 13 | 13 | 22 | 22 | 22 |
| 13 | 13 | 13 | 13 | 13 | 13 | 13 | 22 | 22 | 22 |

11111111: 8

01111100: 5

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 8 | 8 | 8 | 8 | 8 | 8 | 5 | 8 |
| 8 | 8 | 8 | 8 | 8 | 8 | 5 | 8 |
| 8 | 8 | 8 | 8 | 8 | 8 | 5 | 8 |
| 8 | 8 | 8 | 8 | 8 | 8 | 5 | 8 |
| 8 | 8 | 8 | 8 | 8 | 8 | 5 | 8 |
| 8 | 8 | 8 | 8 | 8 | 8 | 5 | 8 |
| 8 | 8 | 8 | 8 | 8 | 8 | 5 | 8 |
| 8 | 8 | 8 | 8 | 8 | 8 | 5 | 8 |

تصویر دوم:

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 73 | 73 | 73 | 73 | 73 | 73 | 73 | 73 | 73 | 73 |
| 73 | 73 | 73 | 73 | 73 | 73 | 73 | 73 | 73 | 73 |
| 73 | 73 | 73 | 73 | 73 | 73 | 73 | 73 | 73 | 73 |
| 73 | 73 | 73 | 73 | 73 | 73 | 73 | 73 | 73 | 73 |
| 73 | 73 | 73 | 73 | 73 | 73 | 73 | 73 | 73 | 73 |
| 85 | 85 | 85 | 85 | 85 | 85 | 85 | 85 | 85 | 85 |
| 85 | 85 | 85 | 85 | 85 | 85 | 85 | 85 | 85 | 85 |
| 85 | 85 | 85 | 85 | 85 | 85 | 85 | 85 | 85 | 85 |
| 85 | 85 | 85 | 85 | 85 | 85 | 85 | 85 | 85 | 85 |
| 85 | 85 | 85 | 85 | 85 | 85 | 85 | 85 | 85 | 85 |

11111111: 8,

00001110: 3,

00011111: 5

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 8 | 8 | 8 | 8 | 8 | 8 | 8 | 8 |
| 8 | 8 | 8 | 8 | 8 | 8 | 8 | 8 |
| 8 | 8 | 8 | 8 | 8 | 8 | 8 | 8 |
| 8 | 8 | 8 | 8 | 8 | 8 | 8 | 8 |
| 5 | 5 | 5 | 5 | 5 | 5 | 5 | 5 |
| 8 | 8 | 8 | 8 | 8 | 8 | 8 | 8 |
| 8 | 8 | 8 | 8 | 8 | 8 | 8 | 8 |
| 8 | 8 | 8 | 8 | 8 | 8 | 8 | 8 |

هیستوگرام تصویر اول:

0:0, 1:0, 2:0, 3:0, 4:0, 5:8, 6:0, 7:0, 8:56

56

8

0 1 2 3 4 5 6 7 8

هیستوگرام تصویر دوم:

0:0, 1:0, 2:0, 3:0, 4:0, 5:8, 6:0, 7:0, 8:56

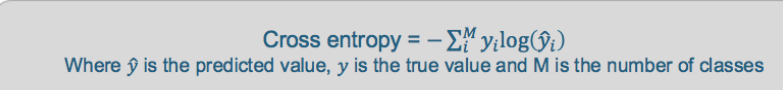
56

8

0 1 2 3 4 5 6 7 8

الگوهای دودویی محلی (نسخه یکنواخت و مستقل از چرخش( برای دو تصویر فرق می‌کند اما همانطور که مشاهده می‌کنید هیستوگرام هر دو یکی است زیرا هیستوگرام فقط به تعداد اهمیت می‌دهد و به اطلاعات مکانی پیکسل‌ها اهمیتی نمی‌دهد.

2)



الف) برای مسائل دسته‌بندی که چندین کلاس داریم اما هر کلاس فقط یک برچسب دارد. مثلا می‌خواهیم اسم یک خودرو را پیشبینی کنیم. چندین دسته وجود دارد اما هر خودرو فقط می‌تواند به یک دسته تعلق داشته باشد. البته باید این توضیح را اضافه کرد که از binary cross entropy که یک حالت خاص از cross entropy است برای دسته‌بندی باینری و همچنین زمانی که در دسته‌بندی چند کلاس داریم و هر نمونه می‌تواند چندین برچسب داشته باشد استفاده می‌شود.

ب) کمترین مقدار این تابع ضرر صفر است و مربوط به زمانی است که کلاس درست پیشبینی شده باشد یعنی y و ŷ کلاس مربوطه هر دو یک باشد.

ج) بیشترین مقدار ∞ است و مربوط به زمانی است که کلاس مربوطه اشتباه پیشبینی شده شده باشد یعنی y=1 و ŷ=0 باشد.

د)

در ابتدا تمام ŷiها مقدار یکسانی دارند و برای هر i:

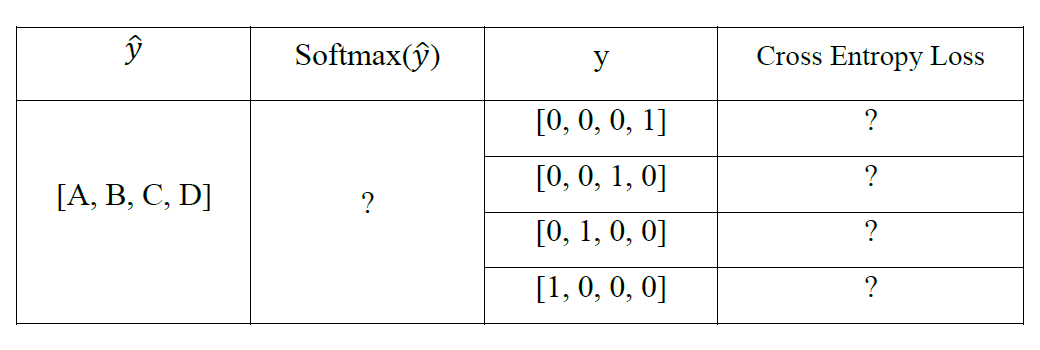
ŷi = 1 / c

در محاسبه خطای cross entropy فقط یک yi مقدار یک دارد بنابراین مقدار خطا برابر است با:

Cross\_entropy\_loss = -log(1/c)

ه)





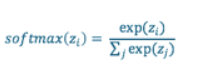
A = 2, B = 0, C = 0, D = 4

c1 = - (0 + 0 + 0 + log4) = - log4 = -2

c2 = - (0 + 0 + log0 + 0) = ∞

c3 = - (0 + log0 + 0 + 0) = ∞

c4 = - (log2 + 0 + 0 + 0) = - log2 = -1



softmax(A) = e2 / (e2 + e0 + e0 + e4) = 0.12

softmax(B) = e0 / (e2 + e0 + e0 + e4) = 0.015

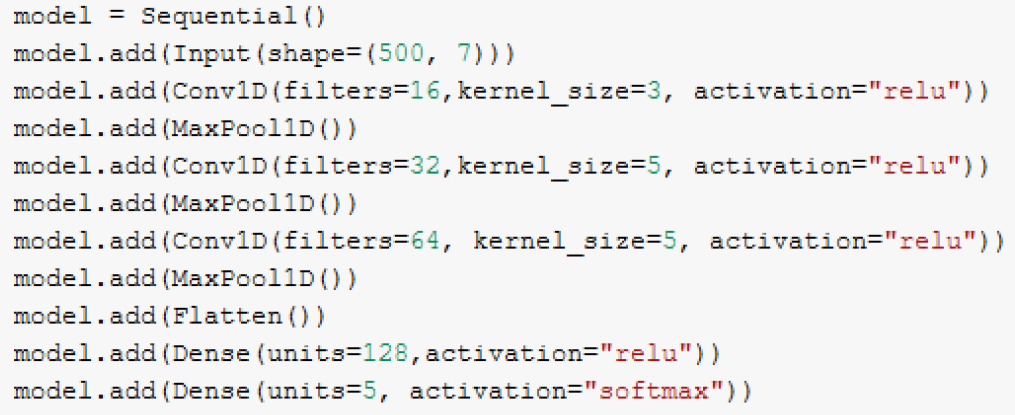
softmax(C) = e0 / (e2 + e0 + e0 + e4) = 0.015

softmax(D) = e4 / (e2 + e0 + e0 + e4) = 0.85

softmax = [0.12 0.015 0.015 0.85]

3)

الف)



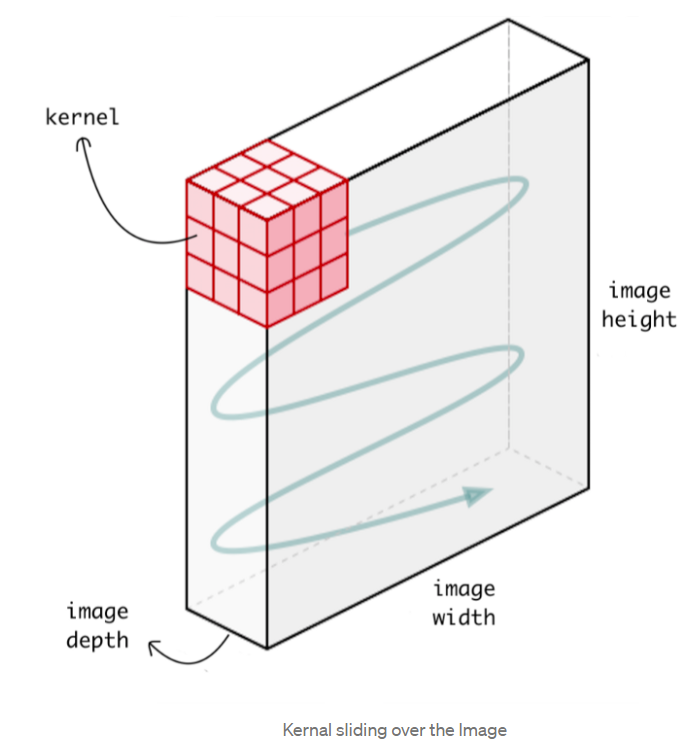
|  |  |  |
| --- | --- | --- |
| layer | Output\_shape | params |
| conv | (500-3+1), 16 = 498, 16 | 16 \* (3 \* 7 + 1) = 352 |
| maxpool | 498 / 2, 16 = 249, 16 | 0 |
| conv | (249-5+1), 16 = 245, 32 | 32 \* (5 \* 16 + 1) = 2592 |
| maxpool | 245 / 2, 32 = 122, 32 | 0 |
| conv | (122-5+1), 64 = 118, 64 | 64 \* (5 \* 32 + 1) = 10304 |
| maxpool | 118 / 2, 64 = 59, 64 | 0 |
| flatten | 3776 | 0 |
| dense | 128 | 128 \* (3776 + 1) = 483456 |
| dense | 5 | 5 \* (128 + 1) = 645 |

تعداد کل پارامترها = 352 + 2592 + 10304 + 483456 + 645 = 497354

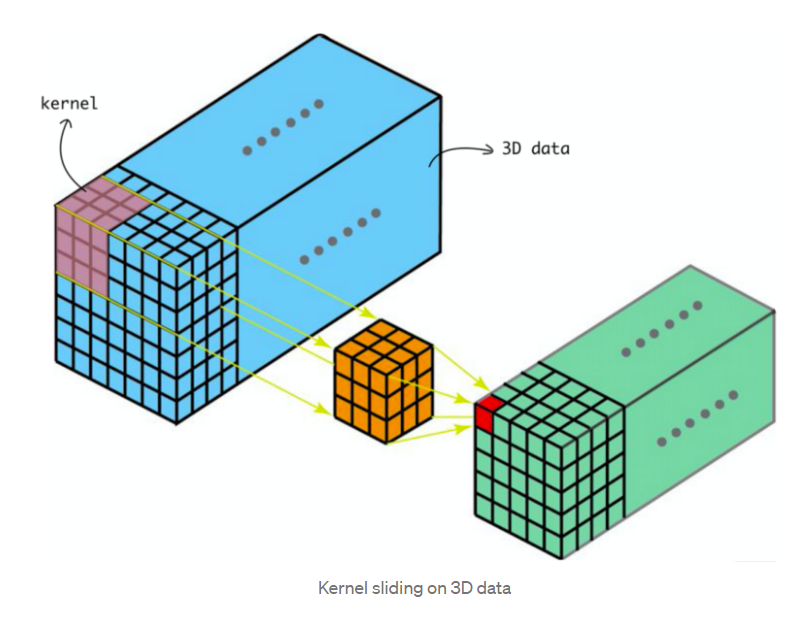
در maxpooling مقدار پیش‌فرض پد valid هست بنابراین وقتی output\_shape[0] فرد هست و maxpooling اعمال می‌شود حاصل به سمت پایین round می‌شود.

ب)

Conv2D برای تصاویر استفاده می‌شود. روش کانولوشن مورد استفاده برای این لایه اصطلاحاً کانولوشن روی حجم نامیده می‌شود. این بدان معناست که شما یک تصویر دو بعدی دارید که شامل چندین کانال، به عنوان مثال RGB است. به آن کانولوشن دو بعدی می‌گویند زیرا کرنل(فیلتر) در امتداد 2 بعد روی داده ها می‌لغزد که در تصویر زیر نشان داده شده است. داده های ورودی و خروجی کانولوشن دو بعدی، سه بعدی است.



در Conv3D، کرنل در 3 بعد مانند شکل زیر می‌لغزد.



داده های ورودی و خروجی کانولوشن 3 بعدی 4 بعدی است. بیشتر در داده های تصویر سه بعدی (MRI، سی تی اسکن) استفاده می شود. Conv3D همچنین برای ویدیوهایی که برای هر بازه زمانی یک فریم وجود دارد، نیز استفاده می‌شود

<https://xzz201920.medium.com/conv1d-conv2d-and-conv3d-8a59182c4d6>

<https://datascience.stackexchange.com/questions/51470/what-are-the-differences-between-convolutional1d-convolutional2d-and-convoluti>