۱)

تصویر اول:

۱۳	۱۳	۱۳	۱۳	۱۳	۱۳	۱۳	۲۲	۲۲	۲۲
۱۳	۱۳	۱۳	۱۳	۱۳	۱۳	۱۳	۲۲	۲۲	۲۲
۱۳	۱۳	۱۳	۱۳	۱۳	۱۳	۱۳	۲۲	۲۲	۲۲
۱۳	۱۳	۱۳	۱۳	۱۳	۱۳	۱۳	۲۲	۲۲	۲۲
۱۳	۱۳	۱۳	۱۳	۱۳	۱۳	۱۳	۲۲	۲۲	۲۲
۱۳	۱۳	۱۳	۱۳	۱۳	۱۳	۱۳	۲۲	۲۲	۲۲
۱۳	۱۳	۱۳	۱۳	۱۳	۱۳	۱۳	۲۲	۲۲	۲۲
۱۳	۱۳	۱۳	۱۳	۱۳	۱۳	۱۳	۲۲	۲۲	۲۲
۱۳	۱۳	۱۳	۱۳	۱۳	۱۳	۱۳	۲۲	۲۲	۲۲
۱۳	۱۳	۱۳	۱۳	۱۳	۱۳	۱۳	۲۲	۲۲	۲۲

11111111: 8

01111100: 5

۸	۸	۸	۸	۸	۸	۵	۸
۸	۸	۸	۸	۸	۸	۵	۸
۸	۸	۸	۸	۸	۸	۵	۸
۸	۸	۸	۸	۸	۸	۵	۸
۸	۸	۸	۸	۸	۸	۵	۸
۸	۸	۸	۸	۸	۸	۵	۸
۸	۸	۸	۸	۸	۸	۵	۸
۸	۸	۸	۸	۸	۸	۵	۸

تصویر دوم:

۷۳	۷۳	۷۳	۷۳	۷۳	۷۳	۷۳	۷۳	۷۳	۷۳
۷۳	۷۳	۷۳	۷۳	۷۳	۷۳	۷۳	۷۳	۷۳	۷۳
۷۳	۷۳	۷۳	۷۳	۷۳	۷۳	۷۳	۷۳	۷۳	۷۳
۷۳	۷۳	۷۳	۷۳	۷۳	۷۳	۷۳	۷۳	۷۳	۷۳
۷۳	۷۳	۷۳	۷۳	۷۳	۷۳	۷۳	۷۳	۷۳	۷۳
۸۵	۸۵	۸۵	۸۵	۸۵	۸۵	۸۵	۸۵	۸۵	۸۵
۸۵	۸۵	۸۵	۸۵	۸۵	۸۵	۸۵	۸۵	۸۵	۸۵
۸۵	۸۵	۸۵	۸۵	۸۵	۸۵	۸۵	۸۵	۸۵	۸۵
۸۵	۸۵	۸۵	۸۵	۸۵	۸۵	۸۵	۸۵	۸۵	۸۵
۸۵	۸۵	۸۵	۸۵	۸۵	۸۵	۸۵	۸۵	۸۵	۸۵

11111111: 8,

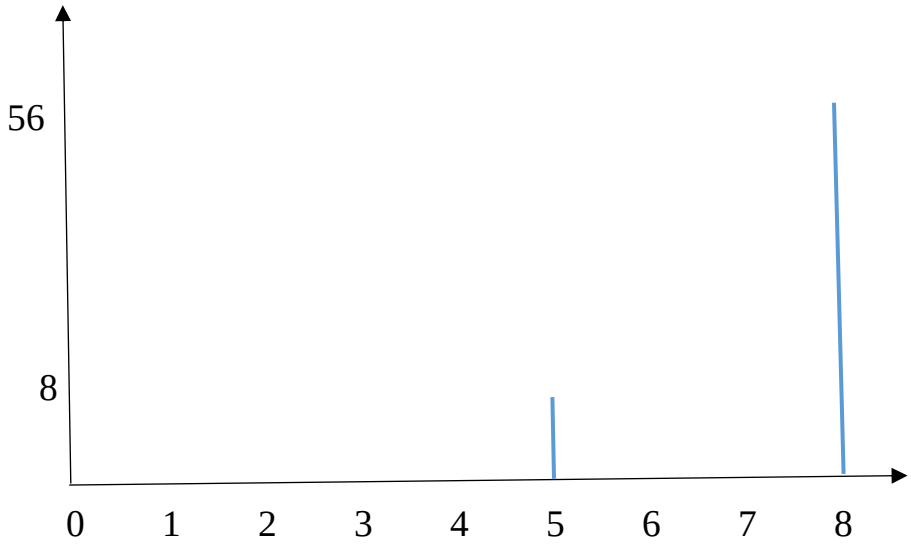
00001110: 3,

00011111: 5

۸	۸	۸	۸	۸	۸	۸	۸
۸	۸	۸	۸	۸	۸	۸	۸
۸	۸	۸	۸	۸	۸	۸	۸
۸	۸	۸	۸	۸	۸	۸	۸
۵	۵	۵	۵	۵	۵	۵	۵
۸	۸	۸	۸	۸	۸	۸	۸
۸	۸	۸	۸	۸	۸	۸	۸
۸	۸	۸	۸	۸	۸	۸	۸

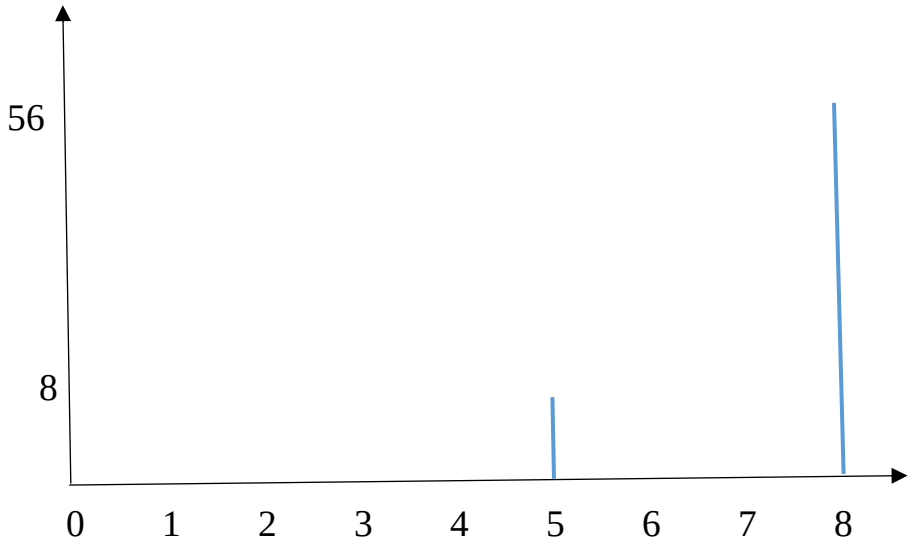
هیستوگرام تصویر اول:

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



هیستوگرام تصویر دوم:

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



الگوهای دودویی محلی LBP_8^1 (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.

۲)

$$\text{Cross entropy} = -\sum_{i}^{M} y_i \log(\hat{y}_i)$$

Where $\hat{y}$ is the predicted value, y is the true value and M is the number of classes

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 $\hat{y}$ 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 $\hat{y}=0$ باشد.

د)

در ابتدا تمام $\hat{y}_i$ها مقدار یکسانی دارند و برای هر i:

$$\hat{y}_i = 1 / c$$

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

$$\text{Cross\_entropy\_loss} = -\log(1/c)$$

ه)

$\hat{y}$	Softmax($\hat{y}$)	y	Cross Entropy Loss
[A, B, C, D]	?	[0, 0, 0, 1]	?
		[0, 0, 1, 0]	?
		[0, 1, 0, 0]	?
		[1, 0, 0, 0]	?

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

c1 = - (0 + 0 + 0 + $\log^4$) = - $\log^4$ = -2

c2 = - (0 + 0 + $\log^0$ + 0) = ∞

c3 = - (0 + $\log^0$ + 0 + 0) = ∞

c4 = - ($\log^2$ + 0 + 0 + 0) = - $\log^2$ = -1

$$softmax(z_i) = \frac{\exp(z_i)}{\sum_j \exp(z_j)}$$

softmax(A) = e^2 / (e^2 + e^0 + e^0 + e^4) = 0.12

softmax(B) = e^0 / (e^2 + e^0 + e^0 + e^4) = 0.015

softmax(C) = e^0 / (e^2 + e^0 + e^0 + e^4) = 0.015

softmax(D) = e^4 / (e^2 + e^0 + e^0 + e^4) = 0.85

softmax = [0.12 0.015 0.015 0.85]

۳)

الف)

```
model = Sequential()
model.add(Input(shape=(500, 7)))
model.add(Conv1D(filters=16,kernel_size=3, activation="relu"))
model.add(MaxPool1D())
model.add(Conv1D(filters=32,kernel_size=5, activation="relu"))
model.add(MaxPool1D())
model.add(Conv1D(filters=64, kernel_size=5, activation="relu"))
model.add(MaxPool1D())
model.add(Flatten())
model.add(Dense(units=128,activation="relu"))
model.add(Dense(units=5, activation="softmax"))
```

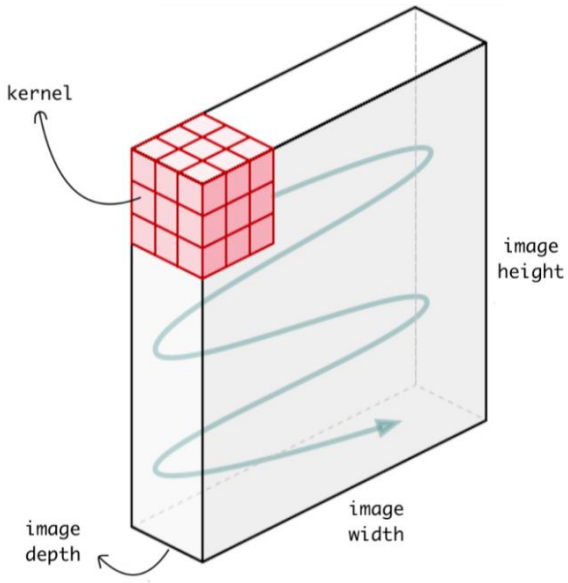
layer	Output_shape	params
conv	(500-3+1), 16 = 498, 16	16 * (3 * 7 + 1) = 352
maxpool	498 / 2, 16 = 249, 16	0
conv	(249-5+1), 16 = 245, 32	32 * (5 * 16 + 1) = 2592
maxpool	245 / 2, 32 = 122, 32	0
conv	(122-5+1), 64 = 118, 64	64 * (5 * 32 + 1) = 10304
maxpool	118 / 2, 64 = 59, 64	0
flatten	3776	0
dense	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.

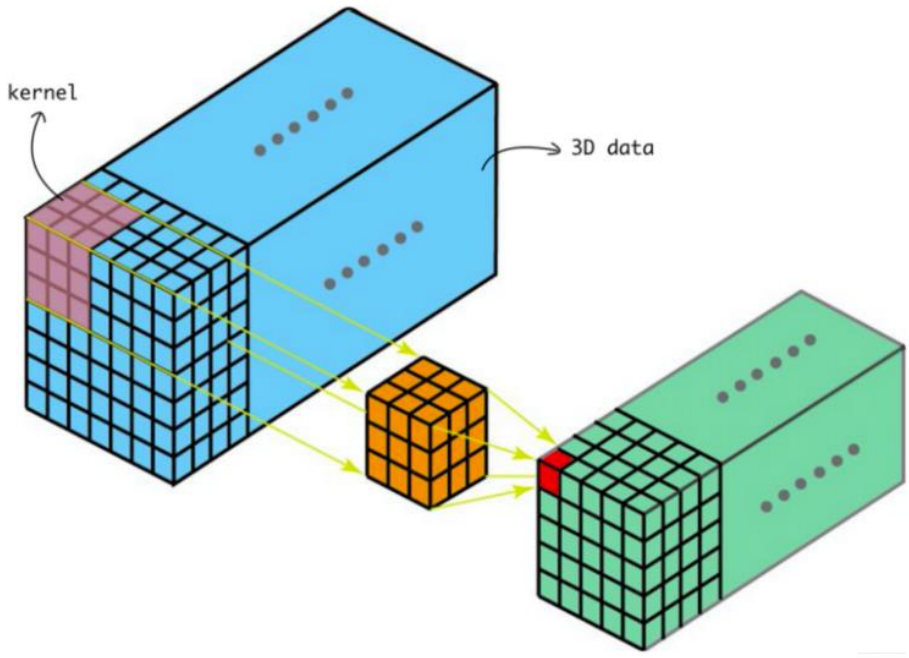
ب)

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 ۲ بعد روی داده ها می‌لغزد که در تصویر زیر نشان داده شده است. داده های ورودی و خروجی کانولوشن دو بعدی، سه بعدی است.



Kernal sliding over the Image

در Conv3D، کرنل در ۳ بعد مانند شکل زیر می‌لغزد.



Kernel sliding on 3D data

داده های ورودی و خروجی کانولوشن ۳ بعدی ۴ 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