Weave Convolution:

The purpose of weave convolution is to grant larger contextual information to convolution without increasing the size of the filters. Instead of increasing the size of the filters, we will use two separate passes of filters. The first set of filters will be used as local information, and the second will be used as peripheral information. Then the results of each filter will be “woven” together to create a larger image which will then be convolved on again.

The inspiration for this idea came from the observation that humans base the local feature of an image with it’s surroundings.

For example, a sharp change in contrast in an image can be interpreted as an edge, but

**Weave Convolution:**

**Inputs:**

Image of Shape (*C, N, N*); *image*

Filter Size: *filter\_size* (must be odd)

Number of Filters; *n\_filters*

**Do:**

*pad* 🡨 (*filter\_size – 1) / 2*

*p\_image* 🡨**2DZeroPad(***image,* *pad***)**

*p\_image* of size (*C, N + pad, N + pad)*

*conv\_loc* 🡨 **2DConvolve** *p\_image* with *n\_filters* of size (*filter\_size, filter\_size)* with stride 1

*conv\_loc* of size (*n\_filters, N, N*)

*conv\_per* 🡨 **2DConvolve** *p\_image* with *n\_filters* of size (*filter\_size, filter\_size)* with stride 1

*conv\_loc* of size (*n\_filters, N, N*)

e\_conv\_loc 🡨 **ZeroExplode** *conv\_loc* by 2 \* *pad*

*w\_conv\_per* 🡨 **ArrayWeave** *conv\_per* by 2 \* *pad*

*conv\_total* 🡨 e\_conv\_loc + *w\_conv\_per*

*o\_image* 🡨 **2DConvolve** *conv\_total* with *n\_filters* of size (*filter\_size, filter\_size)* with stride

*filter\_size*

*o\_image* of size (*n\_filters,* *N,* *N*)

**return** *o\_image*

**ZeroExplode:**

**Inputs:**

Image of Shape (*C, N, N*); *image*

Distance: *distance*

**Do:**

*e\_image* 🡨 zero array of size (*C, distance\**(*N-1) –* distance, *, distance\**(*N-1) –* distance)

**for** *i\_pos*  in **{**0, 1, …, *N-1*}:

**for** *j\_pos* in **{**0, 1, …, *N-1*}:

*e\_image*[: , (*distance+*1)\**i\_pos*, (*distance+*1)\**j\_pos*] 🡨 *image*[: , *i\_pos, j\_pos*]

**return** *e\_image*

**ArrayWeave:**

**Inputs:**

Image of Shape (*C, N, N*); *image*

Distance: *distance*

**Do:**

*e\_distance* 🡨 2 \* *distance* + 2

*w\_image* 🡨 zero array of size (*C, distance\**(*N-1) –* distance, *, distance\**(*N-1) –* distance)

**for** *i\_pos*  in **{**0, 1, …, *N-1*}:

**for** *j\_pos* in **{**0, 1, …, *N-1*}:

**for** *i\_change* in {-e*\_distance, 0, e\_distance*}:

**for** *j\_change* in {-e*\_distance, 0, e\_distance*}:

*new\_x* 🡨 *i\_pos*\*(1+*distance) + i\_change*

*new\_y* 🡨 *j\_pos*\*(1+*distance) + j\_change*

**if** *w\_image*[*new\_x, new\_y*] exists **and** (*i\_change, j\_change) !=* ***0***:

*w\_image*[: , *new\_x, new\_y*] 🡨 *image*[: , *i\_pos, j\_pos*]

**return** *w\_image*