tf.add_n

```
add_n(
    inputs,
    name=None
)
```

Defined in `tensorflow/python/ops/math_ops.py`.

See the guide: Math > Basic Math Functions

Adds all input tensors element-wise.

Args:

- **`inputs`**: A list of `Tensor` objects, each with same shape and type.
- **`name`**: A name for the operation (optional).

Returns:

A `Tensor` of same shape and type as the elements of `inputs`.

Raises:

- **`ValueError`**: If `inputs` don't all have same shape and dtype or the shape cannot be inferred.

tf.abs

```
abs(
    x,
    name=None
)
```

Defined in `tensorflow/python/ops/math_ops.py`.

See the guide: Math > Basic Math Functions

Computes the absolute value of a tensor.

Given a tensor `x` of complex numbers, this operation returns a tensor of type `float32` or `float64` that is the absolute value of each element in `x`. All elements in `x` must be complex numbers of the form *[Math Processing Error]a+bj*. The absolute value is computed as *[Math Processing Error]a2+b2*. For example:

```
x = tf.constant([[-2.25 + 4.75j], [-3.25 + 5.75j]])
tf.abs(x)  # [5.25594902, 6.60492229]
```

Args:

- **`x`**: A `Tensor` or `SparseTensor` of type `float32`, `float64`, `int32`, `int64`, `complex64` or `complex128`.
- **`name`**: A name for the operation (optional).

Returns:

A `Tensor` or `SparseTensor` the same size and type as `x` with absolute values. Note, for `complex64` or `complex128' input, the returned`Tensor`will be of type`float32`or`float64`, respectively.

tf.negative

```
negative(
    x,
    name=None
)
```

Defined in `tensorflow/python/ops/math_ops.py`.

See the guide: Math > Basic Math Functions

Computes numerical negative value element-wise.

I.e., (y = -x).

Args:

- **`x`**: A `Tensor` or `SparseTensor`. Must be one of the following types: `half`, `float32`, `float64`, `int32`, `int64`, `complex64`, `complex128`.
- **`name`**: A name for the operation (optional).

Returns:

A `Tensor` or `SparseTensor`, respectively. Has the same type as `x`.

tf.sign

```
sign(
    x,
    name=None
)
```

Defined in `tensorflow/python/ops/math_ops.py`.

See the guide: Math > Basic Math Functions

Returns an element-wise indication of the sign of a number.

`y = sign(x) = -1` if `x < 0`; 0 if `x == 0` or `tf.is_nan(x)`; 1 if `x > 0`.

Zero is returned for NaN inputs.

For complex numbers, `y = sign(x) = x / |x|` if `x != 0`, otherwise `y = 0`.

Args:

- **`x`**: A `Tensor` or `SparseTensor`. Must be one of the following types: `half`, `float32`, `float64`, `int32`, `int64`, `complex64`, `complex128`.
- **`name`**: A name for the operation (optional).

Returns:

A `Tensor` or `SparseTensor`, respectively. Has the same type as `x`.

numpy compatibility

Equivalent to numpy.sign except for the behavior for input values of NaN.

tf.reciprocal

```
reciprocal(
    x,
    name=None
)
```

Defined in `tensorflow/python/ops/gen_math_ops.py`.

See the guide: Math > Basic Math Functions

Computes the reciprocal of x element-wise.

I.e., y=1/x.

Args:

- **`x`**: A `Tensor`. Must be one of the following types: `half`, `float32`, `float64`, `int32`, `int64`, `complex64`, `complex128`.
- **`name`**: A name for the operation (optional).

Returns:

A `Tensor`. Has the same type as `x`.

tf.square

```
square(
    x,
    name=None
)
```

Defined in `tensorflow/python/ops/math_ops.py`.

See the guide: Math > Basic Math Functions

Computes square of x element-wise.

I.e., y=x∗x=x2.

Args:

- **`x`**: A `Tensor` or `SparseTensor`. Must be one of the following types: `half`, `float32`, `float64`, `int32`, `int64`, `complex64`, `complex128`.
- **`name`**: A name for the operation (optional).

Returns:

A `Tensor` or `SparseTensor`. Has the same type as `x`.

tf.round

```
round(
    x,
    name=None
)
```

Defined in `tensorflow/python/ops/math_ops.py`.

See the guide: Math > Basic Math Functions

Rounds the values of a tensor to the nearest integer, element-wise.

Rounds half to even. Also known as bankers rounding. If you want to round according to the current system rounding mode use tf::cint. For example:

```
x = tf.constant([0.9, 2.5, 2.3, 1.5, -4.5])
tf.round(x)  # [ 1.0, 2.0, 2.0, 2.0, -4.0 ]
```

Args:

- **`x`**: A `Tensor` of type `float32` or `float64`.
- **`name`**: A name for the operation (optional).

Returns:

A `Tensor` of same shape and type as `x`.

tf.sqrt

```
sqrt(
    x,
    name=None
)
```

Defined in `tensorflow/python/ops/math_ops.py`.

See the guide: Math > Basic Math Functions

Computes square root of x element-wise.

I.e., y=x=x1/2.

Args:

- **`x`**: A `Tensor` or `SparseTensor`. Must be one of the following types: `half`, `float32`, `float64`, `complex64`, `complex128`.
- **`name`**: A name for the operation (optional).

Returns:

A `Tensor` or `SparseTensor`, respectively. Has the same type as `x`.

tf.rsqrt

```
rsqrt(
    x,
    name=None
)
```

Defined in `tensorflow/python/ops/gen_math_ops.py`.

See the guide: Math > Basic Math Functions

Computes reciprocal of square root of x element-wise.

I.e., y=1/x.

Args:

- **`x`**: A `Tensor`. Must be one of the following types: `half`, `float32`, `float64`, `complex64`, `complex128`.
- **`name`**: A name for the operation (optional).

Returns:

A `Tensor`. Has the same type as `x`.

tf.pow

```
pow(
    x,
    y,
    name=None
)
```

Defined in `tensorflow/python/ops/math_ops.py`.

See the guide: Math > Basic Math Functions

Computes the power of one value to another.

Given a tensor `x` and a tensor `y`, this operation computes *[Math Processing Error]xy* for corresponding elements in `x` and `y`. For example:

```
x = tf.constant([[2, 2], [3, 3]])
y = tf.constant([[8, 16], [2, 3]])
tf.pow(x, y)  # [[256, 65536], [9, 27]]
```

Args:

- **`x`**: A `Tensor` of type `float32`, `float64`, `int32`, `int64`, `complex64`, or `complex128`.
- **`y`**: A `Tensor` of type `float32`, `float64`, `int32`, `int64`, `complex64`, or `complex128`.
- **`name`**: A name for the operation (optional).

Returns:

A `Tensor`.

tf.exp

```
exp(
    x,
    name=None
)
```

Defined in `tensorflow/python/ops/gen_math_ops.py`.

See the guide: Math > Basic Math Functions

Computes exponential of x element-wise. y=ex.

Args:

- **`x`**: A `Tensor`. Must be one of the following types: `half`, `float32`, `float64`, `complex64`, `complex128`.
- **`name`**: A name for the operation (optional).

Returns:

A `Tensor`. Has the same type as `x`.

tf.expm1

```
expm1(
    x,
    name=None
)
```

Defined in `tensorflow/python/ops/gen_math_ops.py`.

See the guide: Math > Basic Math Functions

Computes exponential of x - 1 element-wise.

I.e., \(y = (\exp x) - 1\).

Args:

- **`x`**: A `Tensor`. Must be one of the following types: `half`, `float32`, `float64`, `complex64`, `complex128`.
- **`name`**: A name for the operation (optional).

Returns:

A `Tensor`. Has the same type as `x`.

tf.log

```
log(
    x,
    name=None
)
```

Defined in `tensorflow/python/ops/gen_math_ops.py`.

See the guide: Math > Basic Math Functions

Computes natural logarithm of x element-wise.

I.e., y=log$_e$x.

Args:

- **`x`**: A `Tensor`. Must be one of the following types: `half`, `float32`, `float64`, `complex64`, `complex128`.
- **`name`**: A name for the operation (optional).

Returns:

A `Tensor`. Has the same type as `x`.

tf.log1p

```
log1p(
    x,
    name=None
)
```

Defined in `tensorflow/python/ops/gen_math_ops.py`.

See the guide: Math > Basic Math Functions

Computes natural logarithm of (1 + x) element-wise.

I.e., y=log$_e$(1+x).

Args:

- **`x`**: A `Tensor`. Must be one of the following types: `half`, `float32`, `float64`, `complex64`, `complex128`.
- **`name`**: A name for the operation (optional).

Returns:

A `Tensor`. Has the same type as `x`.

tf.ceil

```
ceil(
    x,
    name=None
)
```

Defined in `tensorflow/python/ops/gen_math_ops.py`.

See the guide: Math > Basic Math Functions

Returns element-wise smallest integer in not less than x.

Args:

- **`x`**: A `Tensor`. Must be one of the following types: `half`, `float32`, `float64`.
- **`name`**: A name for the operation (optional).

Returns:

A `Tensor`. Has the same type as `x`.

tf.floor

```
floor(
    x,
    name=None
)
```

Defined in `tensorflow/python/ops/gen_math_ops.py`.

See the guide: Math > Basic Math Functions

Returns element-wise largest integer not greater than x.

Args:

- **`x`**: A `Tensor`. Must be one of the following types: `half`, `float32`, `float64`.
- **`name`**: A name for the operation (optional).

Returns:

A `Tensor`. Has the same type as `x`.

tf.maximum

```
maximum(
    x,
    y,
    name=None
)
```

Defined in `tensorflow/python/ops/gen_math_ops.py`.

See the guide: Math > Basic Math Functions

Returns the max of x and y (i.e. x > y ? x : y) element-wise.

NOTE: `Maximum` supports broadcasting. More about broadcasting here

Args:

- **`x`**: A `Tensor`. Must be one of the following types: `half`, `float32`, `float64`, `int32`, `int64`.
- **`y`**: A `Tensor`. Must have the same type as `x`.
- **`name`**: A name for the operation (optional).

Returns:

A `Tensor`. Has the same type as `x`.

tf.minimum

```
minimum(
    x,
    y,
    name=None
)
```

Defined in `tensorflow/python/ops/gen_math_ops.py`.

See the guide: Math > Basic Math Functions

Returns the min of x and y (i.e. x < y ? x : y) element-wise.

NOTE: `Minimum` supports broadcasting. More about broadcasting here

Args:

- **`x`**: A `Tensor`. Must be one of the following types: `half`, `float32`, `float64`, `int32`, `int64`.
- **`y`**: A `Tensor`. Must have the same type as `x`.
- **`name`**: A name for the operation (optional).

Returns:

A `Tensor`. Has the same type as `x`.

tf.cos

```
cos(
    x,
    name=None
)
```

Defined in `tensorflow/python/ops/gen_math_ops.py`.

See the guide: Math > Basic Math Functions

Computes cos of x element-wise.

Args:

- **`x`**: A `Tensor`. Must be one of the following types: `half`, `float32`, `float64`, `complex64`, `complex128`.
- **`name`**: A name for the operation (optional).

Returns:

A `Tensor`. Has the same type as `x`.

tf.sin

```
sin(
    x,
    name=None
)
```

Defined in `tensorflow/python/ops/gen_math_ops.py`.

See the guide: Math > Basic Math Functions

Computes sin of x element-wise.

Args:

- **`x`**: A `Tensor`. Must be one of the following types: `half`, `float32`, `float64`, `complex64`, `complex128`.
- **`name`**: A name for the operation (optional).

Returns:

A `Tensor`. Has the same type as `x`.

tf.lbeta

```
lbeta(
    x,
    name='lbeta'
)
```

Defined in `tensorflow/python/ops/special_math_ops.py`.

See the guide: Math > Basic Math Functions

Computes ln(|Beta(x)|), reducing along the last dimension.

Given one-dimensional `z = [z_0,...,z_{K-1}]`, we define

$$Beta(z)=\prod_j Gamma(z_j)/Gamma(\sum_j z_j)$$

And for `n + 1` dimensional `x` with shape `[N1, ..., Nn, K]`, we define

$$lbeta(x)[i1,...,in]=Log(|Beta(x[i1,...,in,:])|)$$

.

In other words, the last dimension is treated as the `z` vector.

Note that if `z = [u, v]`, then $Beta(z)=int_0^1 t^{u-1}(1-t)^{v-1}dt$, which defines the traditional bivariate beta function.

If the last dimension is empty, we follow the convention that the sum over the empty set is zero, and the product is one.

Args:

- **`x`**: A rank `n + 1` `Tensor`, `n >= 0` with type `float`, or `double`.
- **`name`**: A name for the operation (optional).

Returns:

The logarithm of |Beta(x)| reducing along the last dimension.

tf.tan

```
tan(
    x,
    name=None
)
```

Defined in `tensorflow/python/ops/gen_math_ops.py`.

See the guide: Math > Basic Math Functions

Computes tan of x element-wise.

Args:

- **`x`**: A `Tensor`. Must be one of the following types: `half`, `float32`, `float64`, `int32`, `int64`, `complex64`, `complex128`.
- **`name`**: A name for the operation (optional).

Returns:

A `Tensor`. Has the same type as `x`.

tf.acos

```
acos(
    x,
    name=None
)
```

Defined in `tensorflow/python/ops/gen_math_ops.py`.

See the guide: Math > Basic Math Functions

Computes acos of x element-wise.

Args:

- **`x`**: A `Tensor`. Must be one of the following types: `half`, `float32`, `float64`, `int32`, `int64`, `complex64`, `complex128`.
- **`name`**: A name for the operation (optional).

Returns:

A `Tensor`. Has the same type as `x`.

tf.asin

```
asin(
    x,
    name=None
)
```

Defined in `tensorflow/python/ops/gen_math_ops.py`.

See the guide: Math > Basic Math Functions

Computes asin of x element-wise.

Args:

- **`x`**: A `Tensor`. Must be one of the following types: `half`, `float32`, `float64`, `int32`, `int64`, `complex64`, `complex128`.
- **`name`**: A name for the operation (optional).

Returns:

A `Tensor`. Has the same type as `x`.

tf.atan

```
atan(
    x,
    name=None
)
```

Defined in `tensorflow/python/ops/gen_math_ops.py`.

See the guide: Math > Basic Math Functions

Computes atan of x element-wise.

Args:

- **`x`**: A `Tensor`. Must be one of the following types: `half`, `float32`, `float64`, `int32`, `int64`, `complex64`, `complex128`.
- **`name`**: A name for the operation (optional).

Returns:

A `Tensor`. Has the same type as `x`.