# tf.nn.log_poisson_loss

```
log_poisson_loss(
    targets,
    log_input,
    compute_full_loss=False,
    name=None
)
```

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

See the guide: Neural Network > Losses

Computes log Poisson loss given `log_input`.

Gives the log-likelihood loss between the prediction and the target under the assumption that the target has a Poisson distribution. Caveat: By default, this is not the exact loss, but the loss minus a constant term [log(z!)]. That has no effect for optimization, but does not play well with relative loss comparisons. To compute an approximation of the log factorial term, specify compute_full_loss=True to enable Stirling's Approximation.

For brevity, let `c = log(x) = log_input`, `z = targets`. The log Poisson loss is

```
  -log(exp(-x) * (x^z) / z!)
= -log(exp(-x) * (x^z)) + log(z!)
~ -log(exp(-x)) - log(x^z) [+ z * log(z) - z + 0.5 * log(2 * pi * z)]
    [ Note the second term is the Stirling's Approximation for log(z!).
      It is invariant to x and does not affect optimization, though
      important for correct relative loss comparisons. It is only
      computed when compute_full_loss == True. ]
= x - z * log(x) [+ z * log(z) - z + 0.5 * log(2 * pi * z)]
= exp(c) - z * c [+ z * log(z) - z + 0.5 * log(2 * pi * z)]
```

## Args:

- `targets`: A `Tensor` of the same type and shape as `log_input`.
- `log_input`: A `Tensor` of type `float32` or `float64`.
- `compute_full_loss`: whether to compute the full loss. If false, a constant term is dropped in favor of more efficient optimization.
- `name`: A name for the operation (optional).

## Returns:

A `Tensor` of the same shape as `log_input` with the componentwise logistic losses.

## Raises:

- `ValueError`: If `log_input` and `targets` do not have the same shape.

*Last updated November 2, 2017.*