# tf.contrib.seq2seq.LuongMonotonicAttention

Contents



## Class **LuongMonotonicAttention**

Defined in `tensorflow/contrib/seq2seq/python/ops/attention_wrapper.py`.

Monotonic attention mechanism with Luong-style energy function.

This type of attention encorces a monotonic constraint on the attention distributions; that is once the model attends to a given point in the memory it can't attend to any prior points at subsequence output timesteps. It achieves this by using the _monotonic_probability_fn instead of softmax to construct its attention distributions. Otherwise, it is equivalent to LuongAttention. This approach is proposed in

Colin Raffel, Minh-Thang Luong, Peter J. Liu, Ron J. Weiss, Douglas Eck, "Online and Linear-Time Attention by Enforcing Monotonic Alignments." ICML 2017. https://arxiv.org/abs/1704.00784

## Properties

### `alignments_size`

### `batch_size`

### `keys`

### `memory_layer`

### `query_layer`

### `values`

## Methods

### `__init__`

```
__init__(
    num_units,
    memory,
    memory_sequence_length=None,
    scale=False,
    score_mask_value=float('-inf'),
    sigmoid_noise=0.0,
    sigmoid_noise_seed=None,
    score_bias_init=0.0,
    mode='parallel',
    name='LuongMonotonicAttention'
)
```

Construct the Attention mechanism.

Args:

- `num_units`: The depth of the query mechanism.
- `memory`: The memory to query; usually the output of an RNN encoder. This tensor should be shaped `[batch_size, max_time, ...]`. memory_sequence_length (optional): Sequence lengths for the batch entries in memory. If provided, the memory tensor rows are masked with zeros for values past the respective sequence lengths.
- `scale`: Python boolean. Whether to scale the energy term.
- `score_mask_value`: (optional): The mask value for score before passing into `probability_fn`. The default is -inf. Only used if `memory_sequence_length` is not None.
- `sigmoid_noise`: Standard deviation of pre-sigmoid noise. See the docstring for `_monotonic_probability_fn` for more information.
- `sigmoid_noise_seed`: (optional) Random seed for pre-sigmoid noise.
- `score_bias_init`: Initial value for score bias scalar. It's recommended to initialize this to a negative value when the length of the memory is large.
- `mode`: How to compute the attention distribution. Must be one of 'recursive', 'parallel', or 'hard'. See the docstring for `tf.contrib.seq2seq.monotonic_attention` for more information.
- `name`: Name to use when creating ops.

## `__call__`

```
__call__(
    query,
    previous_alignments
)
```

Score the query based on the keys and values.

Args:

- `query`: Tensor of dtype matching `self.values` and shape `[batch_size, query_depth]`.
- `previous_alignments`: Tensor of dtype matching `self.values` and shape `[batch_size, alignments_size]` (`alignments_size` is memory's `max_time`).

Returns:

- `alignments`: Tensor of dtype matching `self.values` and shape `[batch_size, alignments_size]` (`alignments_size` is memory's `max_time`).

## initial_alignments

```
initial_alignments(
    batch_size,
    dtype
)
```

Creates the initial alignment values for the monotonic attentions.

Initializes to dirac distributions, i.e. [1, 0, 0, ...memory length..., 0] for all entries in the batch.

### Args:

- `batch_size`: **`int32`** scalar, the batch_size.
- `dtype`: The **`dtype`**.

### Returns:

A **`dtype`** tensor shaped **`[batch_size, alignments_size]`** (**`alignments_size`** is the values' **`max_time`**).

---



*Last updated November 2, 2017.*