Sharif University of Technology
Department of Mechanical Engineering

Connectionist Temporal Classification (CTC)
Social Robotics

Dr. Alireza Taheri
Hossein Ranjbar

May 12, 2024

- **TensorFlow**
- **Keras**
- **PyTorch**

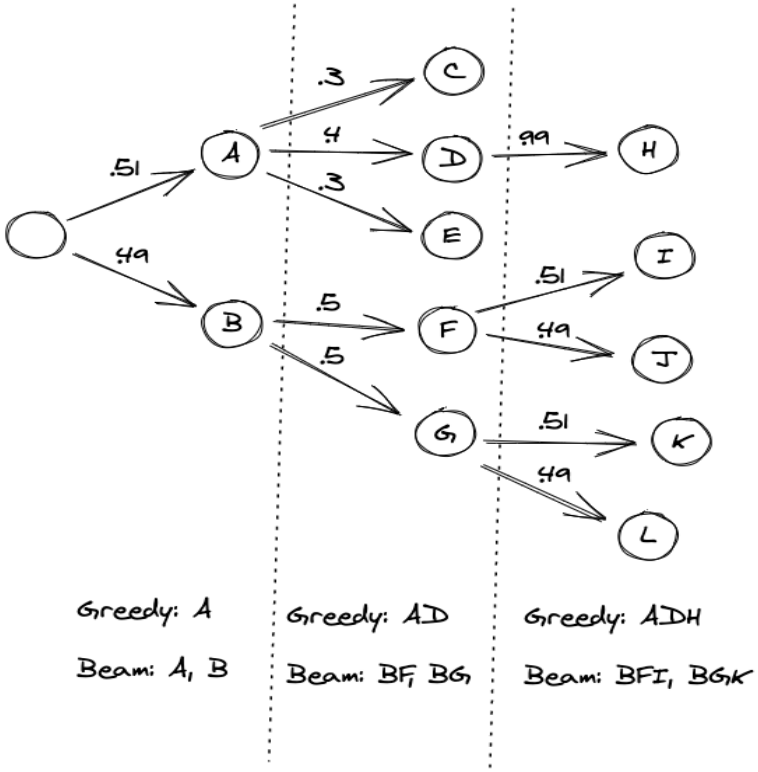
TensorFlow

For training

- tf.nn.ctc_loss(labels, Tensor of shape [batch_size, max_label_seq_length]
  logits, Tensor of shape [frames, batch_size, num_labels]
  label_length, Tensor of shape [batch_size]
  logit_length, Tensor of shape [batch_size]
  logits_time_major, If True (default), [frames, batch_size, num_labels]
  If False, shape is [batch_size, frames, num_labels]
  unique, Unique label indices as computed by ctc_unique_labels(labels)
  blank_index) Set the class index to use for the blank label.
  Default = 0

loss: A 1-D float Tensor of shape [batch_size], containing negative log probabilities.

Beam search and greedy decoder



TensorFlow

For inference

- tf.nn.ctc_beam_search_decoder(

  inputs, 3-D float Tensor, size [max_time, batch_size, num_classes]

  sequence_length, 1-D int32 vector containing sequence lengths[batch_size]

  beam_width, An int scalar >= 0 (beam search beam width)

  top_path) An int scalar >= 0, <= beam_width (controls output size)

Returns: A tuple (decoded, log_probabilities)

TensorFlow

For inference

- tf.nn.ctc_greedy_decoder(

  inputs, 3-D float Tensor, size [max_time, batch_size, num_classes]

  sequence_length, 1-D int32 vector containing sequence lengths[batch_size]

  merge_repeated, Default: True

  blank_index) (Optional). Default: num_classes - 1

Returns: A tuple (decoded, neg_sum_logits)

Keras

for training

- ctc_loss = tf.keras.losses.CTC (
  reduction='sum_over_batch_size',
  name='sparse_categorical_crossentropy')

loss = ctc_loss (y_true, y_pred)

for inference

- tf.keras.ops.ctc_decode(inputs,
  sequence_lengths,
  strategy,
  beam_width,
  top_paths,
  merge_repeated)

"greedy" or "beam_search"

PyTorch

For training

- Torch.nn.CTCLoss(blank, reduction, zeros_infinity)

```
ctc_loss = nn.CTCLoss()
loss = ctc_loss( input,          3-D float Tensor, size [max_time, batch_size, num_classes]
                 target,         Tensor of shape [batch_size, max_label_seq_length]
                 input_lengths,  Tensor of shape [batch_size]
                 target_lengths) Tensor of shape [batch_size]
```

Output: scalar if reduction is 'mean' (default) or 'sum'. If reduction is 'none', output shape will be [batch_size]