Sequence Modeling With CTC

A PhD student's introduction to Connectionist Temporal Classification, an algorithm used to train deep neural networks when the alignment between the input and output sequences is unknown. Illustrated by examples and other research problems.

Awni Hannun
Baidu Research
awni.hannun@baidu.com

Published on Distill: 2017-10-27
DOI: 10.23915/distill.00008

Introduction

Consider speech recognition. We have a dataset of audio clips and corresponding transcriptions, and we'd like to train the Deep Neural Net to map from audio to text. The lengths of the audio and text are both variable, and we don't know the alignment between them.

Without this alignment, the speech supervision cannot underflow us. We could bootstrap a crude alignment by training a Gaussian Mixture Model on the Deep Speech features, but this is complicated and slow. Instead, we'd like to use a loss function that can take our Deep Net, feed it the corresponding transcriptions and audio, and update the weights to increase the probability of good transcriptions. This tutorial describes one such loss function, called CTC, and shows how it can be used to train Deep Neural Networks for speech recognition.

The problem doesn't just arise in speech recognition. We want to learn when alignment information is not available or is too expensive to obtain. Handwriting recognition, machine translation, and text-to-speech are all examples.

Connectionist Temporal Classification (CTC) is a loss, or cost function for training the alignment between the input and the output. It can be used for speech, handwriting and machine translation.

In this post we'd like to focus on a simple, intuitive version of CTC. We'll start with the forward algorithm, which is the heart of the loss function. We'll then show how we can use the loss function to train a recurrent neural network to transcribe speech. We'll conclude by discussing some of the more advanced ideas, including how CTC can be used to learn the alignment between the input and the output.

The Algorithm

The CTC algorithm can be complicated to fully understand, but it's not too bad to develop the probability it assigns to CTC-style alignments from scratch. We'll start with a simple, but inefficient, version of the algorithm. We'll then show how it can be made more efficient with dynamic programming.

Alignment

The CTC algorithm is deterministic. It takes a sequence of inputs and returns the sum over all the outputs. Inputs are in a particular order, and the outputs must be in the same order. We'll first define what we mean by an alignment between the input and the output, and show how alignments are used to compute the loss function.

For a concrete, but quite ridiculous, example, suppose we are given a 5 timestep input and we want the output to be the phoneme sequence "C", "A", "T". We'll call this output sequence the target. We'll call the length of the target U. In this case, $U = 3$.

We'll denote the set of all possible phonemes as $\mathcal{A}$. In this example, $\mathcal{A}$ is the set of all capital letters. We'll denote the length of $\mathcal{A}$ as $|\mathcal{A}|$. In this case, $|\mathcal{A}| = 26$.

We'll also add a special token to $\mathcal{A}$, the blank token, which we'll denote as $\varnothing$. The blank token $\varnothing$ is not a phoneme, but it can be inserted anywhere in the output to produce a valid sequence. We'll call this extended output vocabulary $\mathcal{A}'$. In this case, $|\mathcal{A}'| = 27$.

We'll denote the length of the input T. In this case, $T = 5$. We'll call the set of all length T sequences over $\mathcal{A}'$ the set of all length T paths, $\mathcal{A}'^T$. In this case, $|\mathcal{A}'^T| = 27^5 = 14{,}348{,}907$.

We'll define a function, $\mathcal{B}$, which maps from $\mathcal{A}'^T$ to $\mathcal{A}^{\leq T}$. We'll define $\mathcal{B}$ as the function that first removes all repeated characters, then removes all blanks. For example:

$$\mathcal{B}(\text{"C A A - T"}) = \mathcal{B}(\text{"C - A A T T"}) = \text{"C A T"}$$

We'll use $\mathcal{B}$ to define the probability of a target l given an input x. We'll define the probability of a path π given x as $p(\pi \mid x)$. We'll define the probability of a target l given x as $p(l \mid x)$. We'll define $\mathcal{B}^{-1}(l)$ as the set of all length T paths such that $\mathcal{B}(\pi) = l$. Then we can express the probability of l given x as the sum of the probabilities of all paths in $\mathcal{B}^{-1}(l)$:

$$p(l \mid x) = \sum_{\pi \in \mathcal{B}^{-1}(l)} p(\pi \mid x)$$

We'll now show how we can compute $p(\pi \mid x)$ and $\mathcal{B}^{-1}(l)$, which will let us compute $p(l \mid x)$.

The Algorithm

The CTC algorithm can now be phrased for any l and x for which $\mathcal{B}^{-1}(l)$ is not empty. For training, we probably want to maximize the log probability of the correct labeling, so we'll actually be interested in the gradient of $p(l \mid x)$ with respect to x. We'll show how to compute the forward pass, and then how to compute the gradient.

Inference

The CTC algorithm is also used for inference. Given a trained model, we want to guess a labeling for an input x. We'll show how to do this using a simple heuristic and a more time-consuming beam search.

Extensions

The CTC algorithm is quite flexible. We'll conclude by discussing how it can be extended to work for any sequence to sequence problem, including machine translation and text-to-speech.

Notation

In this post, we will consider a network that takes in T timesteps of input and outputs a distribution over $|\mathcal{A}'|$ characters at each timestep. We'll denote the output at time t as y^t. We'll denote the kth element of y^t as y^t_k. We'll denote the probability of character k at time t as $p(k, t \mid x) = y^t_k$.

In this post we'll be interested in paths, which are length T sequences over $\mathcal{A}'$. We'll denote a path as π. We'll denote the tth character in π as π_t. We'll define the probability of π given x as $p(\pi \mid x) = \prod_{t=1}^{T} y^t_{\pi_t}$.

We'll define a function, $\mathcal{B}$, which maps from $\mathcal{A}'^T$ to $\mathcal{A}^{\leq T}$. We'll define $\mathcal{B}$ as the function that first removes all repeated characters, then removes all blanks. For example:

$$\mathcal{B}(\text{"C A A - T"}) = \mathcal{B}(\text{"C - A A T T"}) = \text{"C A T"}$$

We'll use $\mathcal{B}$ to define the probability of a target l given an input x. We'll define the probability of a path π given x as $p(\pi \mid x)$. We'll define the probability of a target l given x as $p(l \mid x)$. We'll define $\mathcal{B}^{-1}(l)$ as the set of all length T paths such that $\mathcal{B}(\pi) = l$. Then we can express the probability of l given x as the sum of the probabilities of all paths in $\mathcal{B}^{-1}(l)$:

$$p(l \mid x) = \sum_{\pi \in \mathcal{B}^{-1}(l)} p(\pi \mid x)$$

We'll now show how we can compute $p(\pi \mid x)$ and $\mathcal{B}^{-1}(l)$, which will let us compute $p(l \mid x)$.

Alignment Properties

In order to compute $\mathcal{B}^{-1}(l)$, we need to understand how alignments work. We'll propose a mapping between paths in $\mathcal{B}^{-1}(l)$ and labelings l, and use it to argue that all paths in $\mathcal{B}^{-1}(l)$ are equally probable. We'll then show how this means we can find $p(l \mid x)$ if we know how many paths in $\mathcal{B}^{-1}(l)$ lead to each labeling. Lastly, we'll show how we can count the number of paths in $\mathcal{B}^{-1}(l)$ for any l.

Conditional Independence

Since the output at time t is independent of the input given the network parameters, we'll also assume that the probability of a path π is independent of x given π. We'll assume that $p(\pi \mid x) = \prod_{t=1}^{T} y^t_{\pi_t}$.

We'll also assume that the probability of π is independent of the inputs given the previous outputs. We'll assume that $p(\pi) = \prod_{t=1}^{T} p(\pi_t \mid \pi_1, \ldots, \pi_{t-1})$.

These assumptions imply that the probability of a path π is the product of the probabilities of each character in π:

$$p(\pi \mid x) = p(\pi) = \prod_{t=1}^{T} p(\pi_t)$$

Alignment Invariance

The loss function we'll define depends on the alignment between the input and the output. We'll show that many paths in $\mathcal{B}^{-1}(l)$ correspond to the same labeling l. We'll propose a heuristic that assigns probabilities to each of these groups of paths, and show that it's exact for a uniform distribution over paths. We'll show that the probability we want to compute, $p(l \mid x)$, is proportional to the probability we assign to l under this heuristic. This means that if we can compute this heuristic, we can find $p(l \mid x)$. We'll now propose a heuristic that we will show how to compute in the next section.

Let's say that we're given a length T input and we want to know the probability we assign to a labeling l. We'll propose a heuristic that says that the probability we assign to l is proportional to the probability we assign to the most probable path in $\mathcal{B}^{-1}(l)$. We'll now show that this heuristic is exact when we assign uniform probabilities to all length T paths.

Let's say that we assign uniform probabilities to all length T paths. We'll call this probability $q(\pi)$. We'll define $q(\pi) = \frac{1}{|\mathcal{A}'|^T}$. We'll define $Q = |\mathcal{A}'|^T$, so $q(\pi) = \frac{1}{Q}$.

We'll define $\pi^*(l)$ as the most probable path in $\mathcal{B}^{-1}(l)$:

$$\pi^*(l) = \arg\max_{\pi \in \mathcal{B}^{-1}(l)} p(\pi \mid x)$$

We'll define our heuristic, $h(l)$, as the probability we assign to $\pi^*(l)$:

$$h(l) = p(\pi^*(l) \mid x)$$

We'll now show that $h(l) \propto p(l \mid x)$. We'll start by noting that $p(l \mid x) = \sum_{\pi \in \mathcal{B}^{-1}(l)} p(\pi \mid x)$. We'll now show that $p(\pi \mid x) \propto h(l)$ for all $\pi \in \mathcal{B}^{-1}(l)$. We'll start by noting that $p(\pi \mid x) = p(\pi)$. We'll now show that $p(\pi) = \frac{1}{Q}$. We'll now show that $\frac{1}{Q} \propto h(l)$. We'll now show that $h(l) = \frac{1}{Q}$. We'll now show that $p(\pi \mid x) \propto h(l)$. We'll now show that $p(l \mid x) \propto h(l)$.

Counting Paths

We'll now show how we can find $h(l)$. We'll define $n(l)$ as the number of length T paths in $\mathcal{B}^{-1}(l)$. We'll now show that $h(l) = \frac{1}{Q} n(l)$. We'll now show how we can find $n(l)$.

We'll define l' as the labeling we get when we add a blank to the beginning and end of l. We'll now show that $n(l) = n(l') \cdot |l'|$. We'll now show how we can find $n(l')$.

We'll define $n(s, t)$ as the number of paths in $\mathcal{B}^{-1}(l_{1:s})$ such that $\mathcal{B}(\pi_{1:t}) = l_{1:s}$. We'll now show how we can find $n(s, t)$.

We'll now show that $n(s, t) = n(s, t-1) + [l'_s \neq \varnothing] \cdot n(s-1, t-1) + [l'_s \neq l'_{s-2}] \cdot n(s-2, t-1)$.

We'll now show that $n(|l'|, T) + n(|l'|-1, T)$ gives us $n(l)$. We'll now show how we can find $n(l)$.

We'll now show that $n(l) = n(l') \cdot |l'|$. We'll now show that $h(l) = \frac{1}{Q} n(l)$. We'll now show that $p(\pi \mid x) \propto h(l)$. We'll now show that $p(l \mid x) \propto h(l)$.

Forward Algorithm

We'll now show how we can use dynamic programming to compute $n(l)$. We'll define the forward variable $\alpha(s, t)$ as the sum of the probabilities of all paths in $\mathcal{B}^{-1}(l_{1:s})$ such that $\mathcal{B}(\pi_{1:t}) = l_{1:s}$. We'll show that we can compute $\alpha(s, t)$ recursively. We'll show that $n(l) = \alpha(|l'|, T) + \alpha(|l'|-1, T)$. We'll show how we can find $\alpha(|l'|, T)$ and $\alpha(|l'|-1, T)$.

Initialization

We'll now show how we can find $\alpha(1, 1)$ and $\alpha(2, 1)$. We'll show that $\alpha(s, 1) = 0$ for $s > 2$.

Recursion

We'll now show how we can find $\alpha(s, t)$ given we've already found $\alpha(s, t-1)$, $\alpha(s-1, t-1)$, and $\alpha(s-2, t-1)$. We'll show that $\alpha(s, t) = \alpha(s, t-1) + [l'_s \neq \varnothing] \cdot \alpha(s-1, t-1) + [l'_s \neq l'_{s-2}] \cdot \alpha(s-2, t-1)$.

Termination

We'll now show how we can find $n(l)$ given we've found $\alpha(|l'|, T)$ and $\alpha(|l'|-1, T)$. We'll show that $n(l) = \alpha(|l'|, T) + \alpha(|l'|-1, T)$.

Backward Algorithm

We'll now show how we can use dynamic programming to compute $p(\pi \mid x)$. We'll define the backward variable $\beta(s, t)$ as the sum of the probabilities of all paths such that $\mathcal{B}(\pi_{t:T}) = l_{s:|l'|}$. We'll show that we can compute $\beta(s, t)$ recursively. We'll show that $p(\pi \mid x) = \frac{\alpha(s, t) \beta(s, t)}{y^t_{\pi_t}}$. We'll show how we can find $\beta(1, 1)$ and $\beta(2, 1)$. We'll show that $\beta(s, T) = 0$ for $s < |l'| - 1$.

Recursion

We'll now show how we can find $\beta(s, t)$ given we've already found $\beta(s, t+1)$, $\beta(s+1, t+1)$, and $\beta(s+2, t+1)$. We'll show that $\beta(s, t) = \beta(s, t+1) + [l'_s \neq \varnothing] \cdot \beta(s+1, t+1) + [l'_s \neq l'_{s+2}] \cdot \beta(s+2, t+1)$.

Gradient

We'll now show how we can compute $p(\pi \mid x)$ given we've found $\alpha(s, t)$ and $\beta(s, t)$. We'll show that $p(\pi \mid x) = \frac{\alpha(s, t) \beta(s, t)}{y^t_{\pi_t}}$.

We'll now show how we can find $p(l \mid x)$ given we've found $\alpha(s, t)$ and $\beta(s, t)$. We'll show that $p(l \mid x) = \sum_{s=1}^{|l'|} \alpha(s, t) \beta(s, t)$.

We'll now show how we can find $\frac{\partial p(l \mid x)}{\partial y^t_k}$. We'll show that $\frac{\partial p(l \mid x)}{\partial y^t_k} = \frac{1}{(y^t_k)^2} \sum_{s \in lab(l, k)} \alpha(s, t) \beta(s, t)$.

Beam Search

We'll now show how we can use a beam search to find a good guess for the most probable labeling for an input x. We'll show how we can use the forward variables to prune the search. We'll show how we can use a language model to score the labelings we consider.

Extensions

We'll now show how we can use CTC to train a sequence to sequence model, including speech recognition and machine translation. We'll show how we can use a language model to score the labelings we consider during training. We'll show how we can use CTC to learn the alignment between the input and the output.

Conclusion

We've now shown how to compute the CTC loss function, and how to compute the gradient with respect to the network outputs. We've also shown how to use CTC to decode a trained network, and how to incorporate a language model into the decoding. We've also shown how CTC can be used to train models for speech recognition, handwriting recognition, and machine translation. We've also shown how CTC can be used to learn the alignment between the input and the output.

Appendix

Derivation of $\frac{\partial p(l \mid x)}{\partial y^t_k}$

We'll now show how we can find $\frac{\partial p(l \mid x)}{\partial y^t_k}$. We'll show that $\frac{\partial p(l \mid x)}{\partial y^t_k} = \frac{1}{(y^t_k)^2} \sum_{s \in lab(l, k)} \alpha(s, t) \beta(s, t)$.

Footnotes

1. Speech recognition is the task of mapping from audio to text. The inputs to the algorithm are a sequence of frequency spectrum features, and the outputs are a sequence of characters. ↩
2. Machine translation is the task of mapping from text in one language to text in another language. The inputs to the algorithm are a sequence of word or character vectors, and the outputs are a sequence of word or character vectors. ↩
3. Text-to-speech is the task of mapping from text to audio. The inputs to the algorithm are a sequence of word or character vectors, and the outputs are a sequence of frequency spectrum features. ↩
4. Since $\mathcal{B}$ maps from $\mathcal{A}'^T$ to $\mathcal{A}^{\leq T}$, $\mathcal{B}^{-1}$ maps from $\mathcal{A}^{\leq T}$ to $\mathcal{A}'^T$. ↩
5. Since $\mathcal{B}$ maps from $\mathcal{A}'^T$ to $\mathcal{A}^{\leq T}$, $\mathcal{B}^{-1}$ maps from $\mathcal{A}^{\leq T}$ to $\mathcal{A}'^T$. ↩
6. Since $\mathcal{B}$ maps from $\mathcal{A}'^T$ to $\mathcal{A}^{\leq T}$, $\mathcal{B}^{-1}$ maps from $\mathcal{A}^{\leq T}$ to $\mathcal{A}'^T$. ↩
7. Since $\mathcal{B}$ maps from $\mathcal{A}'^T$ to $\mathcal{A}^{\leq T}$, $\mathcal{B}^{-1}$ maps from $\mathcal{A}^{\leq T}$ to $\mathcal{A}'^T$. ↩
8. Since $\mathcal{B}$ maps from $\mathcal{A}'^T$ to $\mathcal{A}^{\leq T}$, $\mathcal{B}^{-1}$ maps from $\mathcal{A}^{\leq T}$ to $\mathcal{A}'^T$. ↩
9. Since $\mathcal{B}$ maps from $\mathcal{A}'^T$ to $\mathcal{A}^{\leq T}$, $\mathcal{B}^{-1}$ maps from $\mathcal{A}^{\leq T}$ to $\mathcal{A}'^T$. ↩
10. Since $\mathcal{B}$ maps from $\mathcal{A}'^T$ to $\mathcal{A}^{\leq T}$, $\mathcal{B}^{-1}$ maps from $\mathcal{A}^{\leq T}$ to $\mathcal{A}'^T$. ↩
11. Since $\mathcal{B}$ maps from $\mathcal{A}'^T$ to $\mathcal{A}^{\leq T}$, $\mathcal{B}^{-1}$ maps from $\mathcal{A}^{\leq T}$ to $\mathcal{A}'^T$. ↩
12. Since $\mathcal{B}$ maps from $\mathcal{A}'^T$ to $\mathcal{A}^{\leq T}$, $\mathcal{B}^{-1}$ maps from $\mathcal{A}^{\leq T}$ to $\mathcal{A}'^T$. ↩
13. Since $\mathcal{B}$ maps from $\mathcal{A}'^T$ to $\mathcal{A}^{\leq T}$, $\mathcal{B}^{-1}$ maps from $\mathcal{A}^{\leq T}$ to $\mathcal{A}'^T$. ↩
14. Since $\mathcal{B}$ maps from $\mathcal{A}'^T$ to $\mathcal{A}^{\leq T}$, $\mathcal{B}^{-1}$ maps from $\mathcal{A}^{\leq T}$ to $\mathcal{A}'^T$. ↩
15. Since $\mathcal{B}$ maps from $\mathcal{A}'^T$ to $\mathcal{A}^{\leq T}$, $\mathcal{B}^{-1}$ maps from $\mathcal{A}^{\leq T}$ to $\mathcal{A}'^T$. ↩
16. Since $\mathcal{B}$ maps from $\mathcal{A}'^T$ to $\mathcal{A}^{\leq T}$, $\mathcal{B}^{-1}$ maps from $\mathcal{A}^{\leq T}$ to $\mathcal{A}'^T$. ↩
17. Since $\mathcal{B}$ maps from $\mathcal{A}'^T$ to $\mathcal{A}^{\leq T}$, $\mathcal{B}^{-1}$ maps from $\mathcal{A}^{\leq T}$ to $\mathcal{A}'^T$. ↩
18. Since $\mathcal{B}$ maps from $\mathcal{A}'^T$ to $\mathcal{A}^{\leq T}$, $\mathcal{B}^{-1}$ maps from $\mathcal{A}^{\leq T}$ to $\mathcal{A}'^T$. ↩
19. Since $\mathcal{B}$ maps from $\mathcal{A}'^T$ to $\mathcal{A}^{\leq T}$, $\mathcal{B}^{-1}$ maps from $\mathcal{A}^{\leq T}$ to $\mathcal{A}'^T$. ↩
20. Since $\mathcal{B}$ maps from $\mathcal{A}'^T$ to $\mathcal{A}^{\leq T}$, $\mathcal{B}^{-1}$ maps from $\mathcal{A}^{\leq T}$ to $\mathcal{A}'^T$. ↩
21. Since $\mathcal{B}$ maps from $\mathcal{A}'^T$ to $\mathcal{A}^{\leq T}$, $\mathcal{B}^{-1}$ maps from $\mathcal{A}^{\leq T}$ to $\mathcal{A}'^T$. ↩
22. Since $\mathcal{B}$ maps from $\mathcal{A}'^T$ to $\mathcal{A}^{\leq T}$, $\mathcal{B}^{-1}$ maps from $\mathcal{A}^{\leq T}$ to $\mathcal{A}'^T$. ↩
23. Since $\mathcal{B}$ maps from $\mathcal{A}'^T$ to $\mathcal{A}^{\leq T}$, $\mathcal{B}^{-1}$ maps from $\mathcal{A}^{\leq T}$ to $\mathcal{A}'^T$. ↩
24. Since $\mathcal{B}$ maps from $\mathcal{A}'^T$ to $\mathcal{A}^{\leq T}$, $\mathcal{B}^{-1}$ maps from $\mathcal{A}^{\leq T}$ to $\mathcal{A}'^T$. ↩
25. Since $\mathcal{B}$ maps from $\mathcal{A}'^T$ to $\mathcal{A}^{\leq T}$, $\mathcal{B}^{-1}$ maps from $\mathcal{A}^{\leq T}$ to $\mathcal{A}'^T$. ↩
26. Since $\mathcal{B}$ maps from $\mathcal{A}'^T$ to $\mathcal{A}^{\leq T}$, $\mathcal{B}^{-1}$ maps from $\mathcal{A}^{\leq T}$ to $\mathcal{A}'^T$. ↩
27. Since $\mathcal{B}$ maps from $\mathcal{A}'^T$ to $\mathcal{A}^{\leq T}$, $\mathcal{B}^{-1}$ maps from $\mathcal{A}^{\leq T}$ to $\mathcal{A}'^T$. ↩
28. Since $\mathcal{B}$ maps from $\mathcal{A}'^T$ to $\mathcal{A}^{\leq T}$, $\mathcal{B}^{-1}$ maps from $\mathcal{A}^{\leq T}$ to $\mathcal{A}'^T$. ↩
29. Since $\mathcal{B}$ maps from $\mathcal{A}'^T$ to $\mathcal{A}^{\leq T}$, $\mathcal{B}^{-1}$ maps from $\mathcal{A}^{\leq T}$ to $\mathcal{A}'^T$. ↩
30. Since $\mathcal{B}$ maps from $\mathcal{A}'^T$ to $\mathcal{A}^{\leq T}$, $\mathcal{B}^{-1}$ maps from $\mathcal{A}^{\leq T}$ to $\mathcal{A}'^T$. ↩
31. Since $\mathcal{B}$ maps from $\mathcal{A}'^T$ to $\mathcal{A}^{\leq T}$, $\mathcal{B}^{-1}$ maps from $\mathcal{A}^{\leq T}$ to $\mathcal{A}'^T$. ↩
32. Since $\mathcal{B}$ maps from $\mathcal{A}'^T$ to $\mathcal{A}^{\leq T}$, $\mathcal{B}^{-1}$ maps from $\mathcal{A}^{\leq T}$ to $\mathcal{A}'^T$. ↩
33. Since $\mathcal{B}$ maps from $\mathcal{A}'^T$ to $\mathcal{A}^{\leq T}$, $\mathcal{B}^{-1}$ maps from $\mathcal{A}^{\leq T}$ to $\mathcal{A}'^T$. ↩
34. Since $\mathcal{B}$ maps from $\mathcal{A}'^T$ to $\mathcal{A}^{\leq T}$, $\mathcal{B}^{-1}$ maps from $\mathcal{A}^{\leq T}$ to $\mathcal{A}'^T$. ↩
35. Since $\mathcal{B}$ maps from $\mathcal{A}'^T$ to $\mathcal{A}^{\leq T}$, $\mathcal{B}^{-1}$ maps from $\mathcal{A}^{\leq T}$ to $\mathcal{A}'^T$. ↩
36. Since $\mathcal{B}$ maps from $\mathcal{A}'^T$ to $\mathcal{A}^{\leq T}$, $\mathcal{B}^{-1}$ maps from $\mathcal{A}^{\leq T}$ to $\mathcal{A}'^T$. ↩
37. Since $\mathcal{B}$ maps from $\mathcal{A}'^T$ to $\mathcal{A}^{\leq T}$, $\mathcal{B}^{-1}$ maps from $\mathcal{A}^{\leq T}$ to $\mathcal{A}'^T$. ↩
38. Since $\mathcal{B}$ maps from $\mathcal{A}'^T$ to $\mathcal{A}^{\leq T}$, $\mathcal{B}^{-1}$ maps from $\mathcal{A}^{\leq T}$ to $\mathcal{A}'^T$. ↩
39. Since $\mathcal{B}$ maps from $\mathcal{A}'^T$ to $\mathcal{A}^{\leq T}$, $\mathcal{B}^{-1}$ maps from $\mathcal{A}^{\leq T}$ to $\mathcal{A}'^T$. ↩
40. Since $\mathcal{B}$ maps from $\mathcal{A}'^T$ to $\mathcal{A}^{\leq T}$, $\mathcal{B}^{-1}$ maps from $\mathcal{A}^{\leq T}$ to $\mathcal{A}'^T$. ↩
41. Since $\mathcal{B}$ maps from $\mathcal{A}'^T$ to $\mathcal{A}^{\leq T}$, $\mathcal{B}^{-1}$ maps from $\mathcal{A}^{\leq T}$ to $\mathcal{A}'^T$. ↩
42. Since $\mathcal{B}$ maps from $\mathcal{A}'^T$ to $\mathcal{A}^{\leq T}$, $\mathcal{B}^{-1}$ maps from $\mathcal{A}^{\leq T}$ to $\mathcal{A}'^T$. ↩
43. Since $\mathcal{B}$ maps from $\mathcal{A}'^T$ to $\mathcal{A}^{\leq T}$, $\mathcal{B}^{-1}$ maps from $\mathcal{A}^{\leq T}$ to $\mathcal{A}'^T$. ↩
44. Since $\mathcal{B}$ maps from $\mathcal{A}'^T$ to $\mathcal{A}^{\leq T}$, $\mathcal{B}^{-1}$ maps from $\mathcal{A}^{\leq T}$ to $\mathcal{A}'^T$. ↩
45. Since $\mathcal{B}$ maps from $\mathcal{A}'^T$ to $\mathcal{A}^{\leq T}$, $\mathcal{B}^{-1}$ maps from $\mathcal{A}^{\leq T}$ to $\mathcal{A}'^T$. ↩
46. Since $\mathcal{B}$ maps from $\mathcal{A}'^T$ to $\mathcal{A}^{\leq T}$, $\mathcal{B}^{-1}$ maps from $\mathcal{A}^{\leq T}$ to $\mathcal{A}'^T$. ↩
47. Since $\mathcal{B}$ maps from $\mathcal{A}'^T$ to $\mathcal{A}^{\leq T}$, $\mathcal{B}^{-1}$ maps from $\mathcal{A}^{\leq T}$ to $\mathcal{A}'^T$. ↩
48. Since $\mathcal{B}$ maps from $\mathcal{A}'^T$ to $\mathcal{A}^{\leq T}$, $\mathcal{B}^{-1}$ maps from $\mathcal{A}^{\leq T}$ to $\mathcal{A}'^T$. ↩
49. Since $\mathcal{B}$ maps from $\mathcal{A}'^T$ to $\mathcal{A}^{\leq T}$, $\mathcal{B}^{-1}$ maps from $\mathcal{A}^{\leq T}$ to $\mathcal{A}'^T$. ↩
50. Since $\mathcal{B}$ maps from $\mathcal{A}'^T$ to $\mathcal{A}^{\leq T}$, $\mathcal{B}^{-1}$ maps from $\mathcal{A}^{\leq T}$ to $\mathcal{A}'^T$. ↩
51. Since $\mathcal{B}$ maps from $\mathcal{A}'^T$ to $\mathcal{A}^{\leq T}$, $\mathcal{B}^{-1}$ maps from $\mathcal{A}^{\leq T}$ to $\mathcal{A}'^T$. ↩
52. Since $\mathcal{B}$ maps from $\mathcal{A}'^T$ to $\mathcal{A}^{\leq T}$, $\mathcal{B}^{-1}$ maps from $\mathcal{A}^{\leq T}$ to $\mathcal{A}'^T$. ↩
53. Since $\mathcal{B}$ maps from $\mathcal{A}'^T$ to $\mathcal{A}^{\leq T}$, $\mathcal{B}^{-1}$ maps from $\mathcal{A}^{\leq T}$ to $\mathcal{A}'^T$. ↩
54. Since $\mathcal{B}$ maps from $\mathcal{A}'^T$ to $\mathcal{A}^{\leq T}$, $\mathcal{B}^{-1}$ maps from $\mathcal{A}^{\leq T}$ to $\mathcal{A}'^T$. ↩
55. Since $\mathcal{B}$ maps from $\mathcal{A}'^T$ to $\mathcal{A}^{\leq T}$, $\mathcal{B}^{-1}$ maps from $\mathcal{A}^{\leq T}$ to $\mathcal{A}'^T$. ↩
56. Since $\mathcal{B}$ maps from $\mathcal{A}'^T$ to $\mathcal{A}^{\leq T}$, $\mathcal{B}^{-1}$ maps from $\mathcal{A}^{\leq T}$ to $\mathcal{A}'^T$. ↩
57. Since $\mathcal{B}$ maps from $\mathcal{A}'^T$ to $\mathcal{A}^{\leq T}$, $\mathcal{B}^{-1}$ maps from $\mathcal{A}^{\leq T}$ to $\mathcal{A}'^T$. ↩
58. Since $\mathcal{B}$ maps from $\mathcal{A}'^T$ to $\mathcal{A}^{\leq T}$, $\mathcal{B}^{-1}$ maps from $\mathcal{A}^{\leq T}$ to $\mathcal{A}'^T$. ↩
59. Since $\mathcal{B}$ maps from $\mathcal{A}'^T$ to $\mathcal{A}^{\leq T}$, $\mathcal{B}^{-1}$ maps from $\mathcal{A}^{\leq T}$ to $\mathcal{A}'^T$. ↩
60. Since $\mathcal{B}$ maps from $\mathcal{A}'^T$ to $\mathcal{A}^{\leq T}$, $\mathcal{B}^{-1}$ maps from $\mathcal{A}^{\leq T}$ to $\mathcal{A}'^T$. ↩
61. Since $\mathcal{B}$ maps from $\mathcal{A}'^T$ to $\mathcal{A}^{\leq T}$, $\mathcal{B}^{-1}$ maps from $\mathcal{A}^{\leq T}$ to $\mathcal{A}'^T$. ↩
62. Since $\mathcal{B}$ maps from $\mathcal{A}'^T$ to $\mathcal{A}^{\leq T}$, $\mathcal{B}^{-1}$ maps from $\mathcal{A}^{\leq T}$ to $\mathcal{A}'^T$. ↩
63. Since $\mathcal{B}$ maps from $\mathcal{A}'^T$ to $\mathcal{A}^{\leq T}$, $\mathcal{B}^{-1}$ maps from $\mathcal{A}^{\leq T}$ to $\mathcal{A}'^T$. ↩
64. Since $\mathcal{B}$ maps from $\mathcal{A}'^T$ to $\mathcal{A}^{\leq T}$, $\mathcal{B}^{-1}$ maps from $\mathcal{A}^{\leq T}$ to $\mathcal{A}'^T$. ↩
65. Since $\mathcal{B}$ maps from $\mathcal{A}'^T$ to $\mathcal{A}^{\leq T}$, $\mathcal{B}^{-1}$ maps from $\mathcal{A}^{\leq T}$ to $\mathcal{A}'^T$. ↩
66. Since $\mathcal{B}$ maps from $\mathcal{A}'^T$ to $\mathcal{A}^{\leq T}$, $\mathcal{B}^{-1}$ maps from $\mathcal{A}^{\leq T}$ to $\mathcal{A}'^T$. ↩
67. Since $\mathcal{B}$ maps from $\mathcal{A}'^T$ to $\mathcal{A}^{\leq T}$, $\mathcal{B}^{-1}$ maps from $\mathcal{A}^{\leq T}$ to $\mathcal{A}'^T$. ↩
68. Since $\mathcal{B}$ maps from $\mathcal{A}'^T$ to $\mathcal{A}^{\leq T}$, $\mathcal{B}^{-1}$ maps from $\mathcal{A}^{\leq T}$ to $\mathcal{A}'^T$. ↩
69. Since $\mathcal{B}$ maps from $\mathcal{A}'^T$ to $\mathcal{A}^{\leq T}$, $\mathcal{B}^{-1}$ maps from $\mathcal{A}^{\leq T}$ to $\mathcal{A}'^T$. ↩
70. Since $\mathcal{B}$ maps from $\mathcal{A}'^T$ to $\mathcal{A}^{\leq T}$, $\mathcal{B}^{-1}$ maps from $\mathcal{A}^{\leq T}$ to $\mathcal{A}'^T$. ↩
71. Since $\mathcal{B}$ maps from $\mathcal{A}'^T$ to $\mathcal{A}^{\leq T}$, $\mathcal{B}^{-1}$ maps from $\mathcal{A}^{\leq T}$ to $\mathcal{A}'^T$. ↩
72. Since $\mathcal{B}$ maps from $\mathcal{A}'^T$ to $\mathcal{A}^{\leq T}$, $\mathcal{B}^{-1}$ maps from $\mathcal{A}^{\leq T}$ to $\mathcal{A}'^T$. ↩
73. Since $\mathcal{B}$ maps from $\mathcal{A}'^T$ to $\mathcal{A}^{\leq T}$, $\mathcal{B}^{-1}$ maps from $\mathcal{A}^{\leq T}$ to $\mathcal{A}'^T$. ↩
74. Since $\mathcal{B}$ maps from $\mathcal{A}'^T$ to $\mathcal{A}^{\leq T}$, $\mathcal{B}^{-1}$ maps from $\mathcal{A}^{\leq T}$ to $\mathcal{A}'^T$. ↩
75. Since $\mathcal{B}$ maps from $\mathcal{A}'^T$ to $\mathcal{A}^{\leq T}$, $\mathcal{B}^{-1}$ maps from $\mathcal{A}^{\leq T}$ to $\mathcal{A}'^T$. ↩
76. Since $\mathcal{B}$ maps from $\mathcal{A}'^T$ to $\mathcal{A}^{\leq T}$, $\mathcal{B}^{-1}$ maps from $\mathcal{A}^{\leq T}$ to $\mathcal{A}'^T$. ↩
77. Since $\mathcal{B}$ maps from $\mathcal{A}'^T$ to $\mathcal{A}^{\leq T}$, $\mathcal{B}^{-1}$ maps from $\mathcal{A}^{\leq T}$ to $\mathcal{A}'^T$. ↩
78. Since $\mathcal{B}$ maps from $\mathcal{A}'^T$ to $\mathcal{A}^{\leq T}$, $\mathcal{B}^{-1}$ maps from $\mathcal{A}^{\leq T}$ to $\mathcal{A}'^T$. ↩
79. Since $\mathcal{B}$ maps from $\mathcal{A}'^T$ to $\mathcal{A}^{\leq T}$, $\mathcal{B}^{-1}$ maps from $\mathcal{A}^{\leq T}$ to $\mathcal{A}'^T$. ↩
80. Since $\mathcal{B}$ maps from $\mathcal{A}'^T$ to $\mathcal{A}^{\leq T}$, $\mathcal{B}^{-1}$ maps from $\mathcal{A}^{\leq T}$ to $\mathcal{A}'^T$. ↩
81. Since $\mathcal{B}$ maps from $\mathcal{A}'^T$ to $\mathcal{A}^{\leq T}$, $\mathcal{B}^{-1}$ maps from $\mathcal{A}^{\leq T}$ to $\mathcal{A}'^T$. ↩
82. Since $\mathcal{B}$ maps from $\mathcal{A}'^T$ to $\mathcal{A}^{\leq T}$, $\mathcal{B}^{-1}$ maps from $\mathcal{A}^{\leq T}$ to $\mathcal{A}'^T$. ↩
83. Since $\mathcal{B}$ maps from $\mathcal{A}'^T$ to $\mathcal{A}^{\leq T}$, $\mathcal{B}^{-1}$ maps from $\mathcal{A}^{\leq T}$ to $\mathcal{A}'^T$. ↩
84. Since $\mathcal{B}$ maps from $\mathcal{A}'^T$ to $\mathcal{A}^{\leq T}$, $\mathcal{B}^{-1}$ maps from $\mathcal{A}^{\leq T}$ to $\mathcal{A}'^T$. ↩
85. Since $\mathcal{B}$ maps from $\mathcal{A}'^T$ to $\mathcal{A}^{\leq T}$, $\mathcal{B}^{-1}$ maps from $\mathcal{A}^{\leq T}$ to $\mathcal{A}'^T$. ↩
86. Since $\mathcal{B}$ maps from $\mathcal{A}'^T$ to $\mathcal{A}^{\leq T}$, $\mathcal{B}^{-1}$ maps from $\mathcal{A}^{\leq T}$ to $\mathcal{A}'^T$. ↩
87. Since $\mathcal{B}$ maps from $\mathcal{A}'^T$ to $\mathcal{A}^{\leq T}$, $\mathcal{B}^{-1}$ maps from $\mathcal{A}^{\leq T}$ to $\mathcal{A}'