Building HMMs with OpenFst

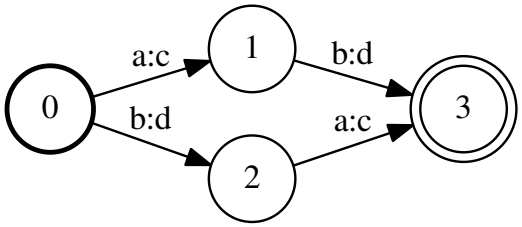
Peter Bell

January 28, 2023

1 Introduction

In the Labs we will use OpenFst (`http://openfst.org`) to build and manipulate finite-state transducer based representations of HMMs. Weighted finite state transducers (WFSTs) are used as the basis of many modern ASR systems, including by the state-the-art Kaldi toolkit.

WFST-based ASR will be covered in detail in later lectures, but for now, all that you need to know is that WFSTs are a graph-based formalism consisting of states and (directed) arcs between them. Arcs have an input label, an output label and optionally, a weight (or cost). The WFST structure *transduces* a sequence of input labels to a sequence of output labels (and optionally, applies a cost to this).

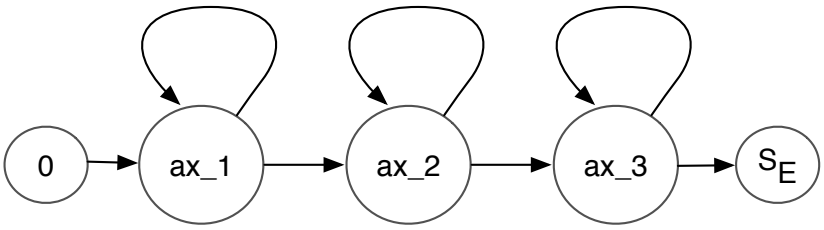


A very simple transducer mapping the string "ab" to "cd" and the string "ba" to "dc". The initial state is shown in bold; the final state is shown with a double circle.

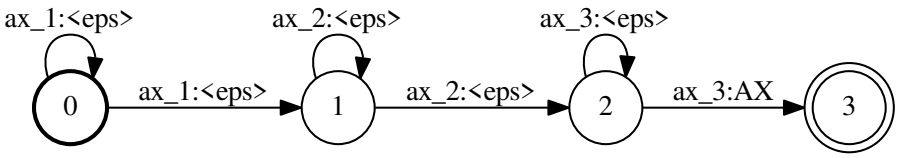
2 The HMM in WFST form

Some small changes are needed to represent the HMM as an FST. It turns out to be easier to think of the HMM emitting states as being on the arcs. The input labels can be used to denote the state ID, whilst the output labels can be used to encode labels to be output by the recogniser, such as words or phones. Note that the epsilon symbol, ε (or `<eps>`), is conventionally used to indicate an empty input/output symbol, meaning that no symbol was consumed/produced by that arc.

This example shows how the 3-state HMM for the phone "AX", containing emitting states that we will label `ax_1`, `ax_2` and `ax_3` can be represented in WFST form. (The non-emitting initial and final states are denoted by 0 and S_E).



Conventional HMM for phone "AX" with three emitting states



WFST representation of the HMM above. Note the output label "AX"