Keyword search intro

CSS Capstone

Mark Simmons

September 30 2025

Section 1: Project background

Big picture

- Capstone project: multilingual Keyword Search (KWS) benchmark
- This lecture:
  1. What is ASR
  2. What is KWS
  3. How to handle sequences with ML

Automatic speech recognition (ASR)

- Transcribing speech with machine learning
- Map audio X to words W
- Used in
  - Voice chatbots
  - Meeting transcription
  - Caption generation

ASR: Why is it useful?

- Transcribing audio is slow
- Chatbots and smart devices need to recognize voice commands
- Auto-captioning video/audio content makes it more accessible

Keyword search (KWS): What is it?

- Intuitively: Ctrl+F for audio
- Detect if keyword W is present in audio X
- Used in:
  - Wake-word recognition
    - “Hey Siri”
  - Routing robocalls
    - Detect if caller says “customer service”, “returns”, etc.
  - Querying a long untranscribed recording
    - Search lecture for all mentions of “gradient descent”

KWS: Why is it useful?

- Tasks like wake-word recognition don't require a full-blown ASR system
- KWS is more lightweight
- An ASR system might not have a certain word in it's vocabulary
  - E.g. an ASR system transcribing LoTR won't recognize words like "Nazgul", "Sauron", "Gondor"
  - We can build a LoTR-specific KWS system to recognize these terms!

KWS benchmark

- What is a benchmark?
  - A dataset for a common ML task
  - Provides baseline model and evaluation results
  - Allows for fair comparison across ML labs
- E.g. LibriSpeech is a common ASR benchmark based on audiobook recordings in English (Panayotov et al. 2015)
- There is no current **multilingual benchmark** for KWS

How to create a benchmark?

- Create or use an open-source dataset
- Define the task and evaluation metrics
- Establish a baseline model & method
- Make accessible to public

How to make a benchmark for KWS?

- An ASR dataset can be repurposed for KWS
  - Define keywords as subset of vocabulary of ASR dataset
  - Each sentence becomes a positive or negative record for a given keyword
    - E.g. for keyword “potatoes” the sentence “I flew to Ireland to learn how to plant potatoes” is a **positive record**
    - The sentence “Right now I’m stuck in AP&M learning about keyword search” is a **negative record** for “potatoes”
  - Evaluate model on ability to spot keywords and avoid false alarms
- Good news: plenty of multi-lingual ASR benchmarks already exist!

Capstone goals

1. Understand ASR and KWS and basic methods for implementing them
2. Transform an existing multilingual ASR dataset into a KWS dataset
3. Implement basic multilingual and/or language-specific KWS models
4. Release the new benchmark to the public

Section 2: Intro to ASR

Classification in ML

- Think of classification as a voting ballot
  - E.g. audio classification: “Do we think waveform X is a /ʃ/ or a /x/?”
  - Define some algorithm that distributes “votes” among the candidate phonemes
- We could do ASR by naively scaling this up:
  - Divide audio into windows
  - Classify each window
  - Build sentence from successive classifications

Naive windowing approach

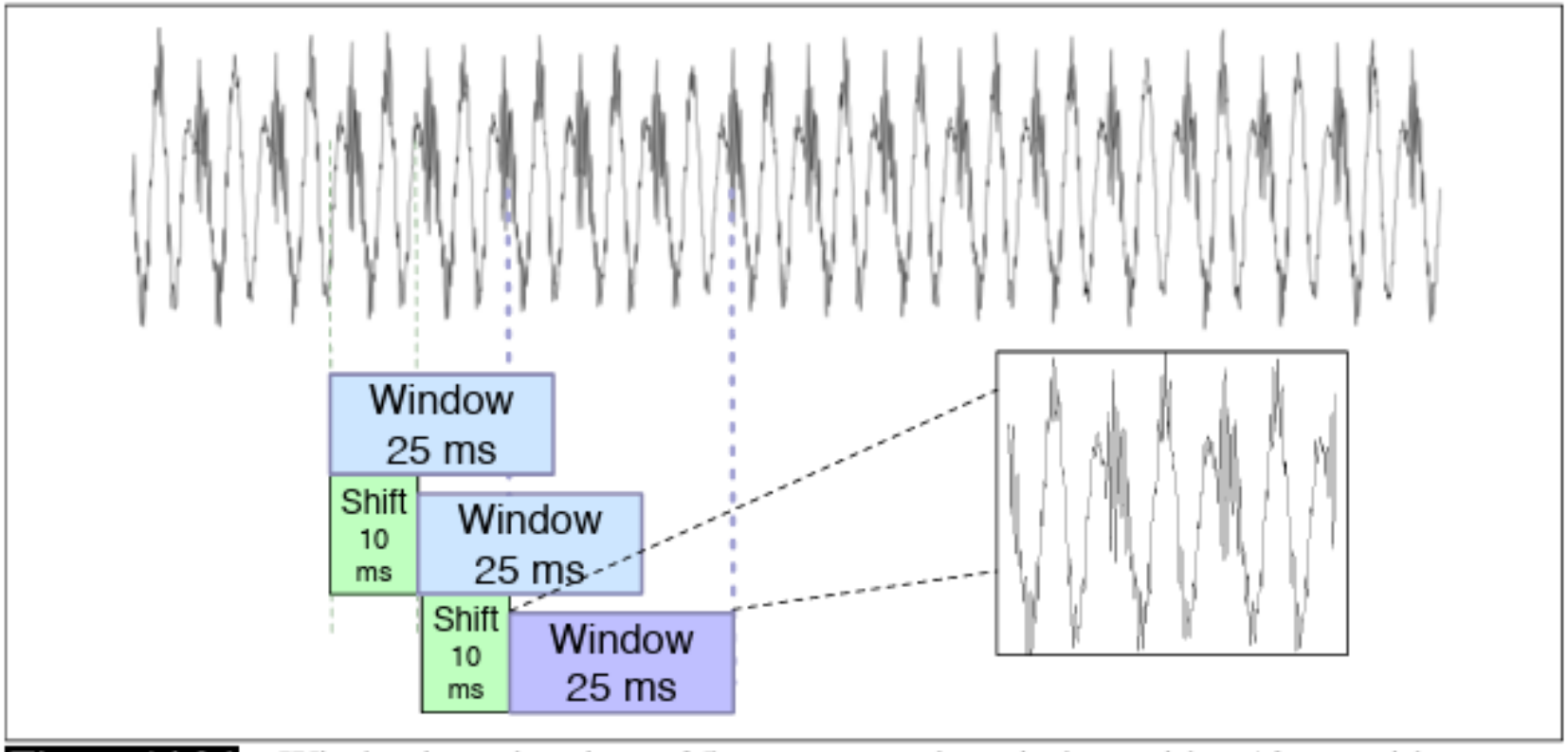
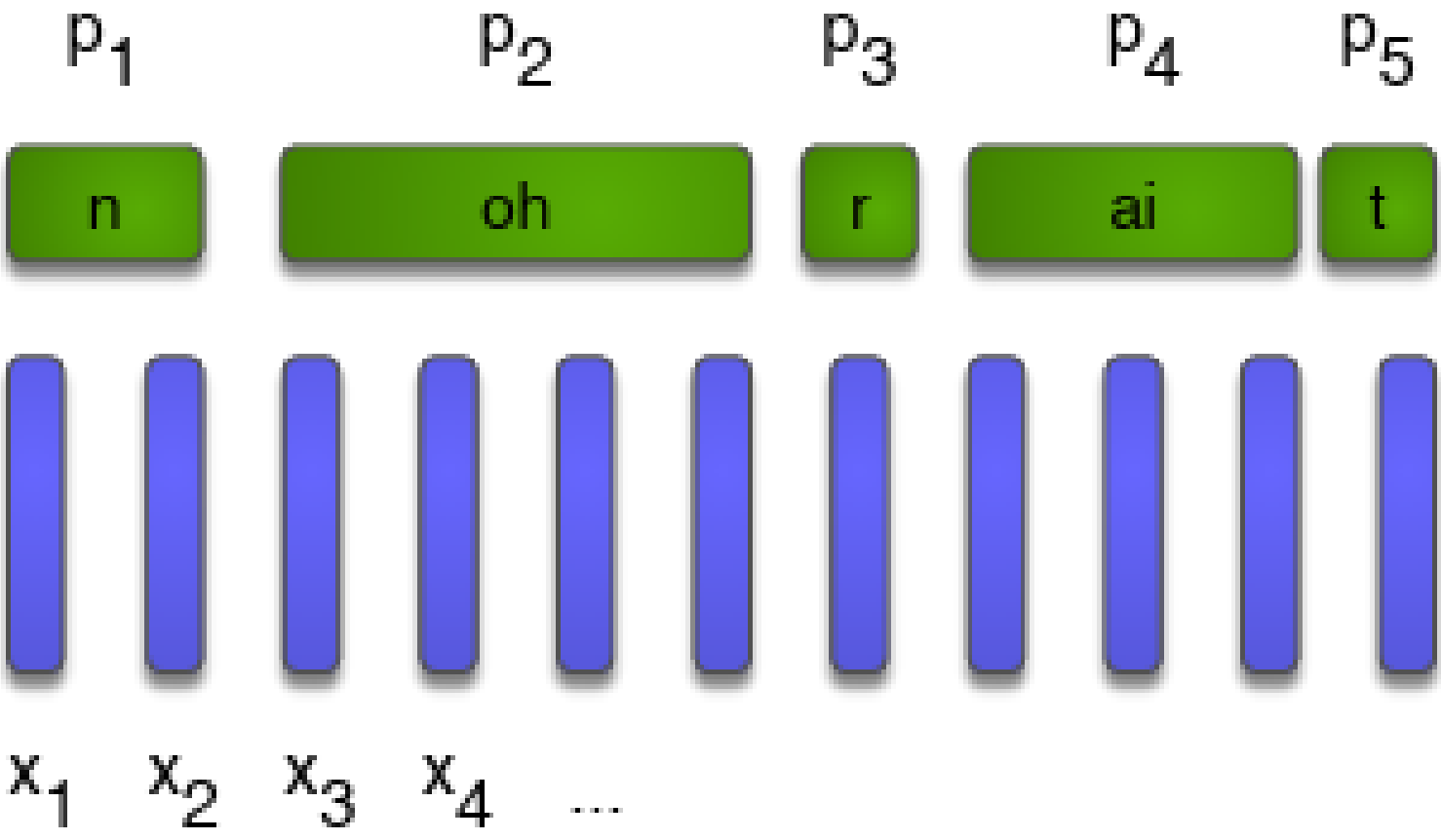


Figure 14.26 Windowing, showing a 25 ms rectangular window with a 10ms stride.

(Jurafsky and Martin 2025)

Naive windowing approach cont.



(Peter Bell, Edinburgh ASR Course Lecture 1, 13 Jan 2025)

What's wrong with this approach

- Handles speech as series of *independent classification decisions*
- No contextual information used!
  - Is this silent interval a pause or the closure of a /k/?
  - Is this unaspirated [p] a /p/ as in "span" or a /b/ as in "ban"?
- How can we fix this?

Sequence modeling in ML

General intuition: our decision space is no longer a voting ballot, now it's a *geographical map*

- First we *survey the landscape*, then we *navigate a path* through it
- Survey the landscape:
  - Classify each window by it's phoneme probabilities
  - Build a *probability landscape* from these classification results
- Navigate a path:
  - Certain paths are inherently more likely than others
  - E.g. "cat" is a more likely output than "ctpcat"
  - Find a path that makes sense given our probability landscape **and** our knowledge of
    - grammar
    - lexicon
    - sentence context
    - etc…

Finite State Transducer

- FSTs are graphs
- In graph theory, this means they have *nodes* and *edges*
- In FST-land, we call nodes ‘states’ and edges ‘arcs’

Figure 1.1: An old-fashioned gumball machine. (Image credit: Dario Lo Presti/Shutterstock.com)

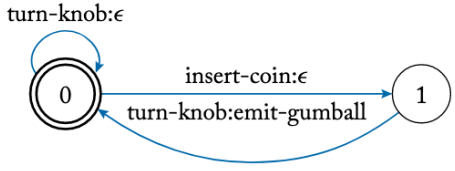


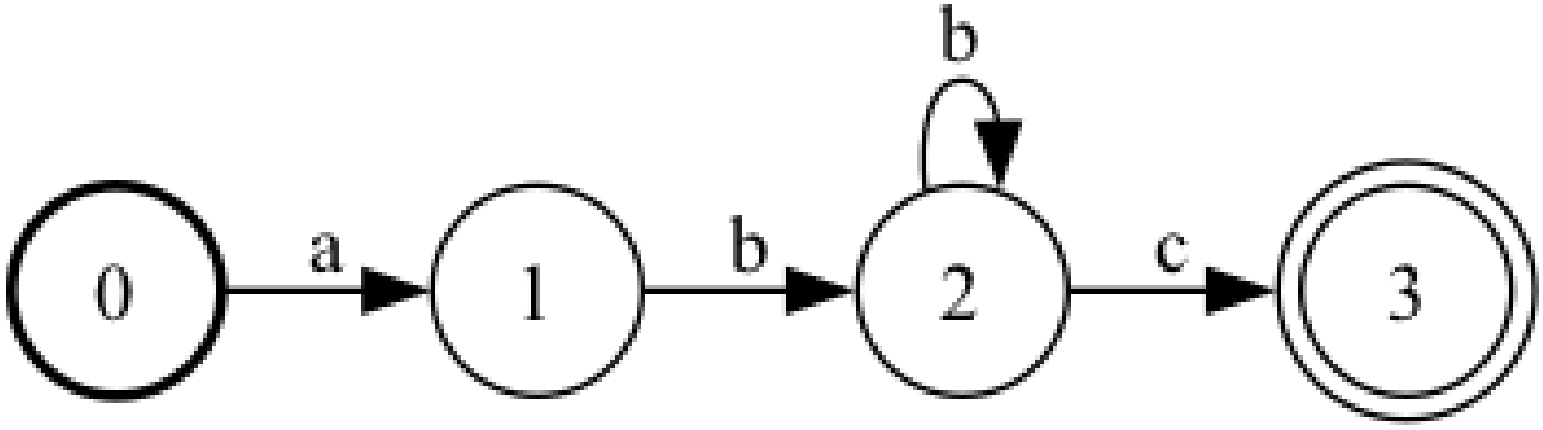
Figure 1.2: An old-fashioned gumball machine schematized as a state machine.

Figure 1: Picture of gumball machine and FST

(Gorman and Sproat 2021)

Finite State Acceptor

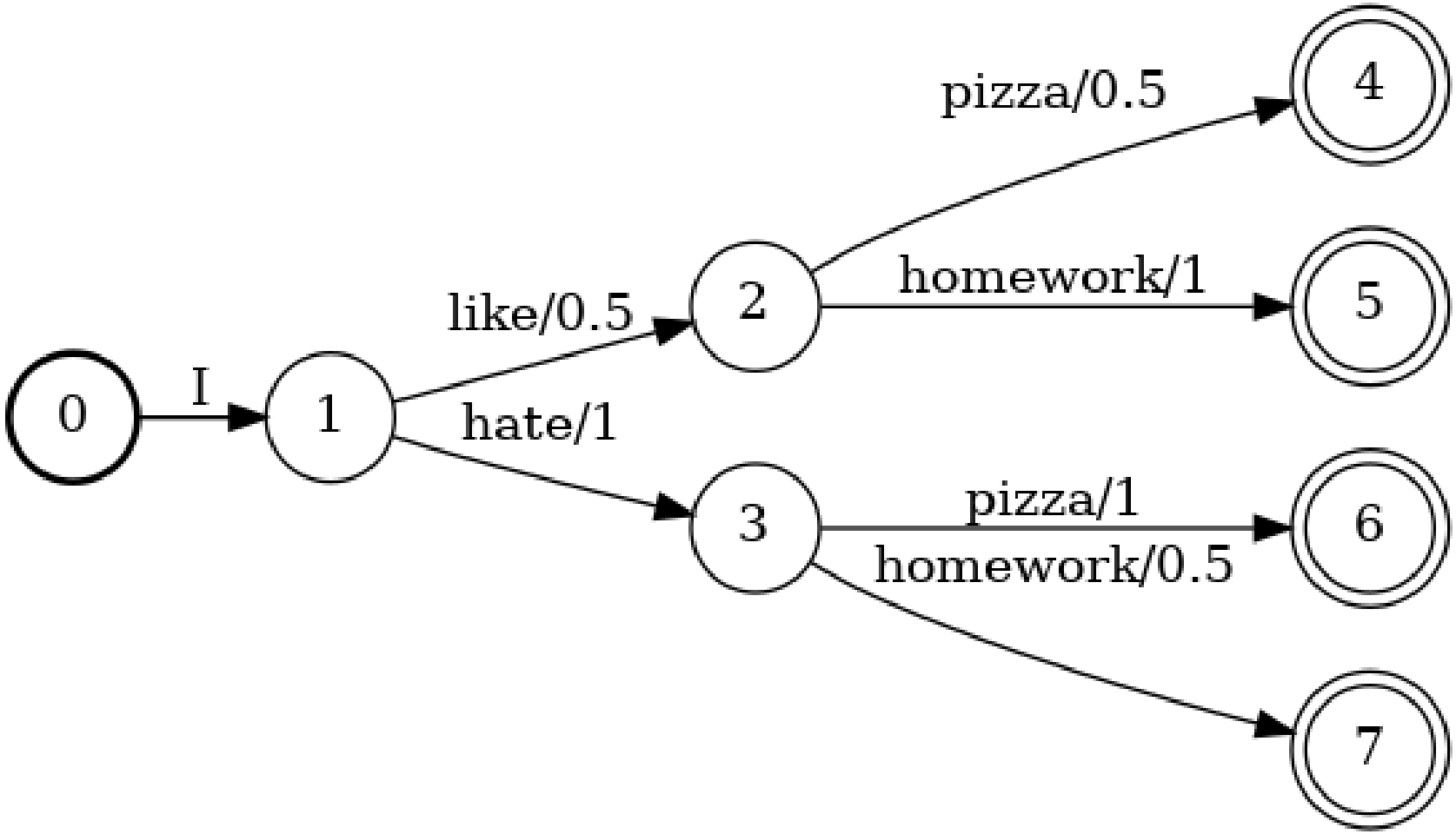
- A graph representing a language
- A ‘language’ in this sense is just a set over strings



- This graph represents the language
  {abc, abbc, abbbc, …}

Language modeling with WFSTs

WFSTs can represent an n-gram language model as described in (Jurafsky and Martin 2025: Chapter 3)



ASR with WFSTs

How can WFSTs be used for ASR? See Week 1's python notebok to find out!

Wrap-up

- We reviewed ASR and KWS at a conceptual level
- We discussed the challenges of doing ML with sequential data like speech
- We introduced the *navigation* metaphor for understanding how to handle sequential data
- We introduced WFSTs, a simple graph model used in ASR

Gorman, Kyle, and Richard Sproat. 2021. *Finite-State Text Processing*. Synthesis Lectures on Human Language Technologies. Cham: Springer International Publishing. https://doi.org/10.1007/978-3-031-02179-4.

Jurafsky, Daniel, and James H. Martin. 2025. *Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition, with Language Models*. 3rd ed.

Panayotov, Vassil, Guoguo Chen, Daniel Povey, and Sanjeev Khudanpur. 2015. “Librispeech: An ASR Corpus Based on Public Domain Audio Books.” In *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 5206–10. South Brisbane, Queensland, Australia: IEEE. https://doi.org/10.1109/ICASSP.2015.7178964.