novauris

Speak, *Find.* ...*EASY!*

Spoken List Access

The Strengths of Multistage ASR

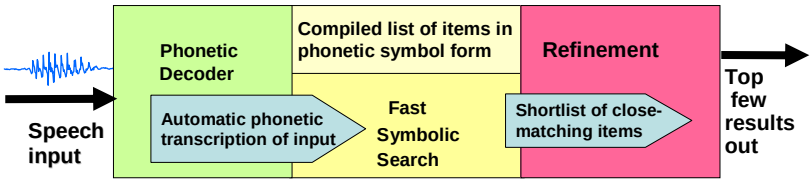
Melvyn Hunt

Co-Founder and President

- From its foundation, Novauris's aim has been to provide unparalleled speed and accuracy in spoken selection from very large sets of items.
- To this end, we developed a *multistage* approach to searching lists.
- First public demonstration was with 245 million names & addresses (error rate negligible; response time ~ 1 sec).
- The efficiency of the search allows it to be run on embedded processors
  - even for challenging apps such as all U.S. street addresses
- All known third-party tests have found that this approach provides the highest accuracy.
- This talk provides some reasons why.

The multistage approach

- Produce approximate phonetic transcription
- Match that against a large list of items,
  each consisting of networks of phonetic symbols
  - using special fast match techniques
- Produce a short list that has much smaller
- Refine the short list using a more conventional approach
  - Produce an n-best list that that is adequate given the UI



We aren't trying to recognize words

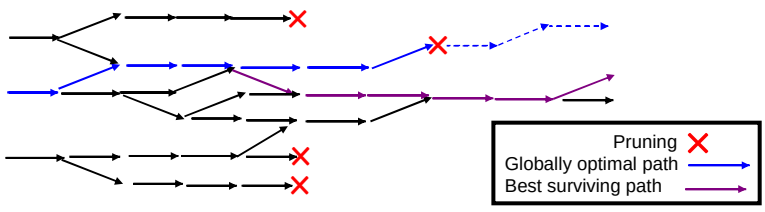
- We don't normally know or care what words were spoken
  - we're just trying to associate the input with a list item:
    - an address
    - a POI
    - a person to be phoned
    - an idea (request, question, statement, *etc*.) to be translated
    - …
  - each list item corresponds to a network of phonetic symbols
    - or maybe several alternative networks
  - each list item is associated with an action, such as:
    - display some text
    - dial a number
    - record a TV program
    - play some particular music
    - speak a loose translation with the sense of what was said

What are the Advantages of the Multistage Approach?

- For some tasks (*e.g.* small vocabulary, with short inputs or words in any order) probably no advantage.
- However, for many tasks, there are major advantages, including:
  - It's an effective fast match
    - symbol-to-symbol comparison *vs* frame-to-frame
  - It's particularly good in avoiding *pruning errors*
    - Both in the symbolic search and the refinement
    - Especially important for long-range dependencies, where an item near the end of an input (*e.g.* a ZIP code), can help distinguish earlier items (*e.g.* the street name)

What are *"Pruning Errors"?*

- Speech recognition normally matches the input speech to sequences of *acoustic models*
  - matching forwards in time, and
  - allowing for large variations in speaking rates
- This results in large numbers of *alignment paths*
- To keep computation in bounds, the less promising paths are *"pruned out"*
- But the path that would have turned out to be best, may well get pruned out — a *pruning error*



How can we Avoid Pruning Errors?

- Conventional speech recognition typically operates with 100 “frames” per sec.
  - that’s around 10 frames per “phoneme”.
  - Each frame typically contains 39 numbers.
  - Weighted Euclidean distances are computed between each frame and the, say, 16 Gaussians representing each of typically 3 states in a phonetic model.
  - That’s ~ 10*39*16 = **6,240 multiplies and 12,480 adds** for just one alignment path between one phonetic model and the frames of one spoken phoneme.
- The corresponding symbolic match process is **a single table lookup!**
- So symbolic matching can afford to keep many more hypotheses alive — avoiding pruning errors
  - the correct item is almost always in the set passed to the refinement stage
- The refinement process works at the frame rate, but there are small # of candidates, so – again – tight pruning is unnecessary.

Alphanumeric Product Codes*

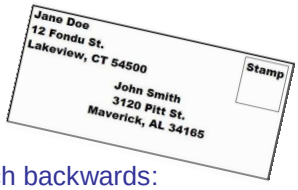
where multistage ASR is virtually essential

- One customer has 60,000 quasi-random codes of varying length
  - examples: PR7178701, 3624R, 6333514
- Using conventional ASR, reliable recognition would be very challenging
- But errors with Novauris ASR are very rare
  - because fast symbolic matching compares the input with complete patterns without any pruning
  - all the constraints in the sequences are taken into account
  - and grammar compilation from a text list takes <4 sec

*Also discussed in the Angel.com talk this morning

Reverse Hierarchies

- Addresses are a classic example of a list
- But the US has > 5.7 million streets
  - too big to use the simple method employed for product codes
- Solution?
  - Exploit the hierarchical structure:
    - Number, Street, City, State
  - But that's backwards!
  - No problem! — We do the symbolic search backwards:
    - first match the state, then the city, then the street
    - backtrack if necessary



- For conventional ASR to run backwards, no recognition can begin until the input is complete
  - but with multistage ASR, the phonetic transcription runs forwards
  - this, and the inherent efficiency, is why we can offer single-shot recognition of any US street address on an embedded processor

Flexibility

- Remember we aren't recognizing words
  – only list items
  (so "Slim Shady" and "Eminem" can be the same item)

- Next slide shows how we can use the pronouncing dictionary to handle variants

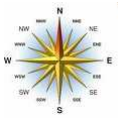
Anticipated Variant Forms of the Input

- Because the symbolic match can search many items quickly, we can include many anticipated variations in the items
  - The translation application described this morning has more than 100 variants for some items
- We have compact methods for describing the variants
- For some kinds of variants, a dictionary-based technique can speed up matching to the variants
  - Optional words can be given "null" pronunciations, and certain unimportant words can be given multiple pronunciations.
    - If <Title> = { "Mr", "Mrs", "Miss", "Ms", "Dr", "Rev", "Prof", Null } and <John> = { "John", "J.", Null" },
    - then <Title> <John> <Smith> <Title> <John> can match:
      - John Smith, Mr J. Smith, Smith John, Smith Dr J. …
    - It can also match "John Smith John", "Mr Smith Mrs" …
      - But usually that doesn't matter.

Flexibilities with Probabilities

- Recognition accuracy with alternative forms is optimized by taking probabilities into account
  - Examples:

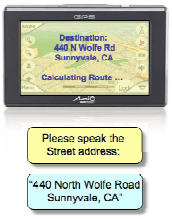
- In our Japan railway iPhone app (Friday @ 11:20am, Presidio) making the word for "station" (*eki*) optional after every station name lowers accuracy, but allowing it with a penalty raises it.



- In addresses, allowing compass points to be pronounced as letters (e.g. "north-west" = N.W.) works best with a penalty applied to the letter variants.
- In addresses, house numbers outside the published range for a street are allowed but penalized depending on how far outside the range they are.

  - This allows for extra buildings to be added and prevents errors in citing the number from disrupting the rest of the address.

Thank You



Demo Videos on YouTube

Broadcast Yourself™

www.youtube.com/novauristechnologies

novauris

Speak, Find. *...EASY!*