Disclaimer

- The material provided in this document is not my original work and is a summary of some one else's work(s).
- A simple Google search of the title of the document will direct you to the original source of the material.
- I do not guarantee the accuracy, completeness, timeliness, validity, non-omission, merchantability or fitness of the contents of this document for any particular purpose.
- Downloaded from najeebkhan.github.io

Chapter # 10: Decoding with HVite

HMM TOOL KIT HTK

Outline

- Introduction
- Decoder Operation
- Decoder Organization
- Recognition using Test Databases
- Evaluating Recognition Results
- Generating Forced Alignments
- Recognition using Direct Audio Input
- N-Best Lists and Lattices

Introduction

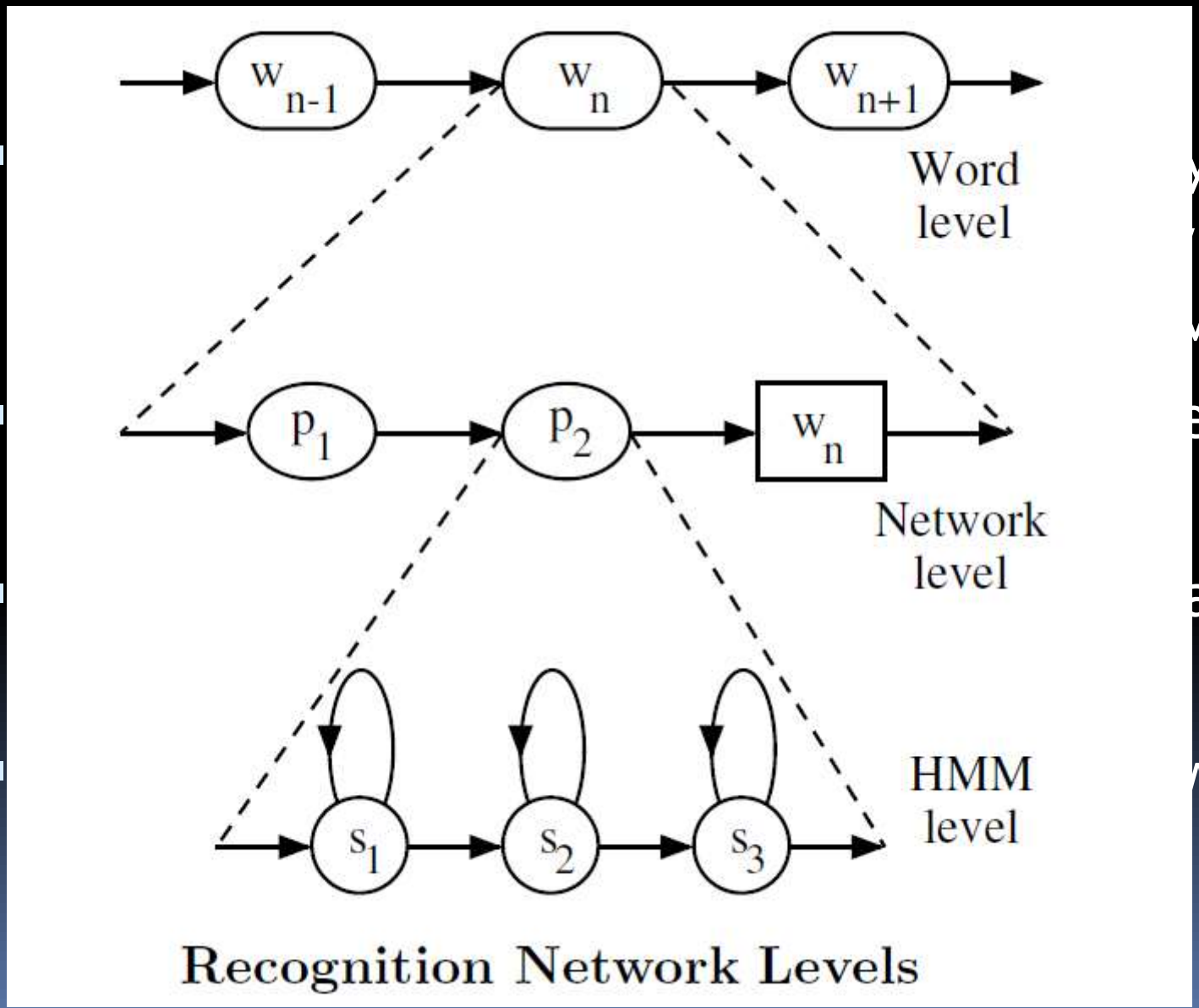
- A recognition network specifies what is allowed to be spoken and how each word is pronounced
- Given such a network, its associated set of HMMs, and an unknown utterance, the probability of any path through the network can be computed
- The task of a decoder is to find those paths which are the most likely

Decoder Operation

- Decoding in HTK is controlled by a recognition network compiled from a word-level network, a dictionary and a set of HMMs
- The recognition network consists of a set of nodes connected by arcs
- Each node is either an HMM model instance or a word-end
- Each model node is itself a network consisting of states connected by arcs

Decoder Operation

- y a
- ord-
- MMs
- et of
- ance
- work



Decoder Operation

- For an unknown input utterance with T frames, every path from the start node to the exit node of the network which passes through exactly T emitting HMM states is a potential recognition hypothesis
- Within-HMM transitions are determined from the HMM parameters, between-model transitions are constant and word-end transitions are determined by the language model likelihoods attached to the word level networks

Decoder Operation

- The job of the decoder is to find those paths through the network which have the highest log probability
- Each time step, tokens are propagated along connecting transitions stopping whenever they reach an emitting HMM state
- As the token passes across transitions and through nodes, its log probability is incremented by the corresponding transition and emission probabilities
- A network node can hold at most N tokens
- As each token passes through the network it must maintain a history recording its route

Decoder Operation

- Pruning is implemented at each time step by keeping a record of the best token overall and de-activating all tokens whose log probabilities fall more than a beam-width below the best
- For efficiency reasons, it is best to implement primary pruning at the model rather than the state level
- Thus, models are deactivated when they have no tokens in any state within the beam
- If the pruning beam-width is set too small then the most likely path might be pruned before its token reaches the end of the utterance

Decoder Organization

- The decoding process itself is performed by a set of core functions provided within the library module HRec

De

- T

s

l

d by a

n the

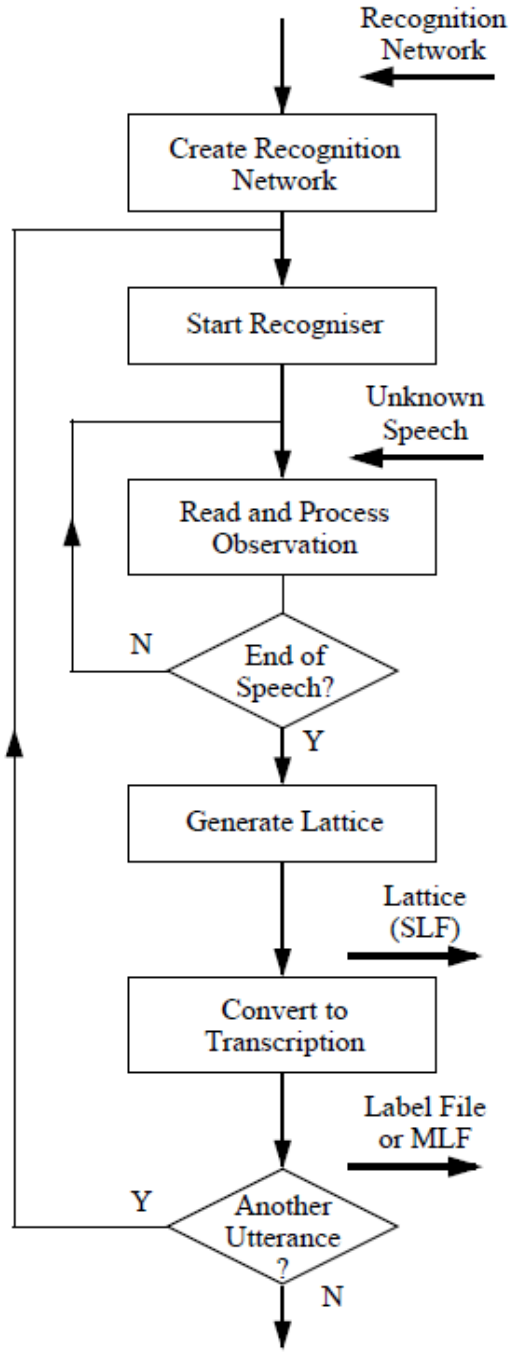


Fig. 13.2 Recognition Processing

Decoder Organization

- Input control in the form of a recognition network allows three distinct modes of operation
- Recognition: This is the conventional case in which the recognition network is compiled from a task level word network
- Forced Alignment: The recognition network is constructed from a word level transcription and dictionary
- Lattice-based Rescoring: The input network is compiled from a lattice generated during an earlier recognition run

Decoder Organization

- The second source of flexibility lies in the provision of multiple tokens and recognition output in the form of a lattice
- In addition to providing a mechanism for rescoring, lattice output can be used as a source of multiple hypotheses either for further recognition processing or input to a natural language processor

Recognition using Test Databases

```
HVite -w wdnet dict hmmlist testf1 testf2 ....

HVite -T 1 -S test.scp -H hmmset -i results -w wdnet dict hmmlist

File: testf1.mfc
SIL ONE NINE FOUR SIL
[178 frames] -96.1404 [Ac=-16931.8 LM=-181.2] (Act=75.0)

 "testf1.rec"
        0  6200000 SIL  -6067.333008
  6200000  9200000 ONE  -3032.359131
  9200000 12300000 NINE -3020.820312
 12300000 17600000 FOUR -4690.033203
 17600000 17800000 SIL   -302.439148
 .
```

`-o ST`

```
"testf1.rec"
SIL
ONE
NINE
FOUR
SIL
.
```

Recognition using Test Databases

- The relative levels of insertion and deletion errors can be controlled by scaling the language model likelihoods using the -s option and adding a fixed penalty using the -p option
- Setting -s 10.0 -p -20.0 would mean that every language model log probability x would be converted to 10x - 20 before being added to the tokens emitted from the corresponding word-end node
- As an extreme example, setting -p 100.0 caused the digit recognizer above to output

```
SIL OH OH ONE OH OH OH NINE FOUR OH OH OH OH SIL
```

Evaluating Recognition Results

- HResults compares the transcriptions output by HVite with the original reference transcriptions and then outputs various statistics
- Hresults matches each of the recognised and reference label sequences by performing an optimal string match using dynamic programming
- Identical labels match with score 0, a label insertion carries a score of 7, a deletion carries a score of 7 and a substitution carries a score of 10

Evaluating Recognition Results

- Once the optimal alignment has been found, the number of substitution errors (S), deletion errors (D) and insertion errors (I) can be calculated

$$\text{Percent Correct} = \frac{N - D - S}{N} \times 100\%$$

$$\text{Percent Accuracy} = \frac{N - D - S - I}{N} \times 100\%$$

Evaluating Recognition Results

```
HResults -I refs wlist results
====================== HTK Results Analysis =======================
  Date: Sat Sep  2 14:14:22 1995
  Ref : refs
  Rec : results
---------------------- Overall Results ----------------------------
SENT: %Correct=98.50 [H=197, S=3, N=200]
WORD: %Corr=99.77, Acc=99.65 [H=853, D=1, S=1, I=1, N=855]
====================================================================

       Aligned transcription: testf9.lab vs testf9.rec
        LAB: FOUR    SEVEN NINE THREE
        REC: FOUR OH SEVEN FIVE THREE
```

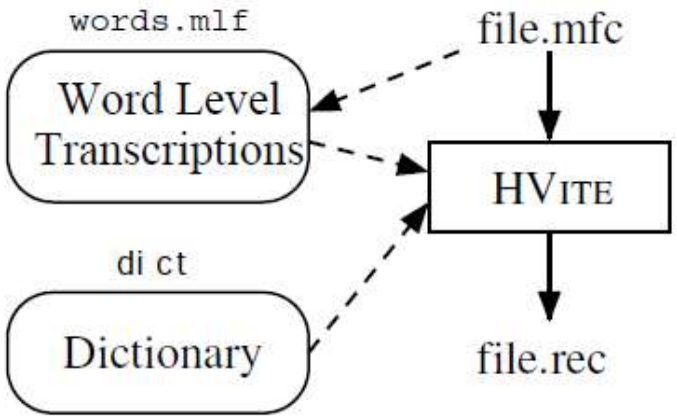
Evaluating Recognition Results

```
DIGITS_spkr_nnnn.rec

HResults -h -k '*_%%%%_????.*' ....

,---------------------------------------------------------.
| HTK Results Analysis at Sat Sep  2 15:05:37 1995        |
| Ref: refs                                               |
| Rec: results                                            |
|---------------------------------------------------------|
|    SPKR | # Snt |  Corr    Sub    Del    Ins    Err  S. Err |
|---------------------------------------------------------|
|    dgo1 |   20  | 100.00   0.00   0.00   0.00   0.00   0.00 |
|---------------------------------------------------------|
|    pcw1 |   20  |  97.22   1.39   1.39   0.00   2.78  10.00 |
|---------------------------------------------------------|
......
|=========================================================|
| Sum/Avg |  200  |  99.77   0.12   0.12   0.12   0.35   1.50 |
`---------------------------------------------------------'
```

Generating Forced Alignments




```
HVite -a -b sil -m -o SWT -I words.mlf -H hmmset dict hmmlist file.mfc
```

Recognition using Direct Audio Input

- Covered in Chapter 3 and 5

N-Best Lists and Lattices

- To generate an N-best list, the -n option is used to specify the number of N-best tokens to store per state and the number of N-best hypotheses to generate

```
-n 4 20
"testf1.rec"
FOUR
SEVEN
NINE
OH
///
FOUR
SEVEN
NINE
OH
OH
///
```

N-Best Lists and Lattices

- The lattices from which the N-best lists are generated can be output by setting the option –z ext

N-Best Lists and Lattices

```
VERSION=1.0
UTTERANCE=testf1.mfc
lmname=wdnet
lmscale=20.00  wdpenalty=-30.00
vocab=dict
N=31    L=56
I=0     t=0.00
I=1     t=0.36
I=2     t=0.75
I=3     t=0.81
... etc
I=30    t=2.48
J=0      S=0     E=1     W=SILENCE    v=0   a=-3239.01   l=0.00
J=1      S=1     E=2     W=FOUR       v=0   a=-3820.77   l=0.00
... etc
J=55     S=29    E=30    W=SILENCE    v=0   a=-246.99    l=-1.20
```

Thank You