Hill Climbing on Speech Lattices: A New Rescoring Framework

Ariya Rastrow, Markus Dreyer, Abhinav Sethy, Sanjeev Khudanpur, Bhuvana Ramabhadran and Mark Dredze
• Availability of large amounts of training data and computational resources
  • building more complex models with sentence level knowledge and longer dependencies is the active area of research for ASR

• Many of these complex and sophisticated models can not be integrated into the first pass decoding

• They can not be represented as weighted finite-state automata (WFSA)
  • difficult to even incorporate them in a lattice-rescoring pass
Motivation

Exponential number of word-sequences
Motivation

Exponential number of word sequences enumerating all the hypothesis is not feasible.
• Instead, \textit{N-best rescoring} strategy is employed
  
  • Enumerating over the list of \textit{N} best hypotheses (w.r.t the initial model)

• \textit{N-best rescoring} suffers from known \textit{deficiencies} and \textit{inefficiencies}
N-best rescoring is not a smart strategy!

Motivation

Points of the search space

Model Score

$f_{\text{new}}$

$f_{\text{init}}$
Points of the search space

Model Score

Points of the search space

$f_{\text{new}}$

$f_{\text{init}}$
Points of the search space

Model Score

Points of the search space

Motivation
Selected points *need not* be representing the best points of the *rescoring* model, in the search space (lattice).
Motivation

search errors for small $N$ 😞

Points of the search space

Model Score

$f_{\text{new}}$

$f_{\text{init}}$
$N$ needs to be increased to get closer to the optimal solution.

But ....
Motivation

Considering a large $N$ makes the rescoring computationally expensive.

Points of the search space

Model Score

Points of the search space
Motivation

Our Solution:

Use the more complex model to aid hypotheses selection, as opposed to considering the $N$ hypotheses chosen by the simpler model.
Motivation

Our Solution:
Motivation

Our Solution:

*Hill Climbing* on speech lattices
Motivation

Our Solution:

Hill Climbing on speech lattices 😊
Hill Climbing

• An iterative improvement search strategy:
  i. Starts with an initial solution in the search space
  ii. Examines a neighborhood of the initial point and steps to the best point in the neighborhood (objective function is increasing most steeply)
  iii. Iterates the procedure for the new selected point
  iv. Stops when the current solution cannot be further improved

• For a broad class of problems, hill climbing is guaranteed to reach a local maximum solution
Hill Climbing

Points of the search space

Model Score

initial point

Tuesday, April 5, 2011
Hill Climbing

Points of the search space

Model Score

Tuesday, April 5, 2011
Hill Climbing

Points of the search space

Neighborhood set

Model Score

Points of the search space
Hill Climbing

best solution in the neighborhood

Model Score

Points of the search space
Hill Climbing

Model Score vs. Points of the search space
Hill Climbing

Points of the search space

Model Score

Tuesday, April 5, 2011
Hill Climbing

Points of the search space

Model Score

Tuesday, April 5, 2011
Hill Climbing
Hill Climbing

Points of the search space

local maximum

Model Score

Points of the search space
The search space consists of set of word-sequences

\[ \Rightarrow w_1 w_2 \cdots w_n \in L \]

- It is natural to define the neighborhood function using the edit-distance function

Specifically, the neighborhood set is defined by editing at specific position \( i \) of word sequence \( W \)

- This neighborhood is represented by \( \mathcal{N}(W, i) \)
- deleting, substituting or inserting a word to the left of \( w_i \)

- How to generate \( \mathcal{N}(W, i) \) efficiently? (will be explained later)
In this work, we use hill climbing for LM rescoring

- The lattice-generating LM is replaced with a long-span/complex LM
- We gradually climb the search space (word-sequences in the lattice) to maximize:

\[
g(X, W; \Lambda, \Gamma_{\text{new}}) = \alpha \log P(X|W, \Lambda) + \log P(W|\Gamma_{\text{new}})
\]
**Initialization:** the highest scoring word sequence (the *viterbi* path) is selected from the initial lattice.
Hill Climbing on Speech Lattices

1. **Initialization**: the highest scoring word sequence (the *viterbi* path) is selected from the initial lattice.

2. **Neighborhood Generation**: for a selected position $i$, all paths in the lattice corresponding to word-sequences are extracted.

\[ W' \in \mathcal{N}(W, i) \]
Hill Climbing on Speech Lattices

1. **Initialization:** the highest scoring word sequence (the viterbi path) is selected from the initial lattice.

2. **Neighborhood Generation:** for a selected position $i$, all paths in the lattice corresponding to word-sequences are extracted. $W' \in \mathcal{N}(W, i)$

3. **Neighborhood Rescoring:** evaluating all the word sequences in the neighborhood set, and selecting the word sequence with maximum score for the next step.

$$g(X, W'; \Lambda, \Gamma_{\text{new}}) = \alpha \log P(X|W', \Lambda) + \log P(W'|\Gamma_{\text{new}})$$
Hill Climbing on Speech Lattices

1. **Initialization:** The highest scoring word sequence (the *viterbi* path) is selected from the initial lattice.

2. **Neighborhood Generation:** For a selected position $i$, all paths in the lattice corresponding to word-sequences are extracted. $W' \in \mathcal{N}(W, i)$

3. **Neighborhood Rescoring:** Evaluating all the word sequences in the neighborhood set, and selecting the word sequence with maximum score for the next step.

$$g(X, W'; \Lambda, \Gamma_{\text{new}}) = \alpha \log P(X|W', \Lambda) + \log P(W'|\Gamma_{\text{new}})$$
Hill Climbing on Speech Lattices

1. **Initialization:** the highest scoring word sequence (the viterbi path) is selected from the initial lattice.

2. **Neighborhood Generation:** for a selected position $i$, all paths in the lattice corresponding to word-sequences are extracted. $W' \in \mathcal{N}(W, i)$

3. **Neighborhood Rescoring:** evaluating all the word sequences in the neighborhood set, and selecting the word sequence with maximum score for the next step.

$$g(X, W'; \Lambda, \Gamma_{\text{new}}) = \alpha \log P(X|W', \Lambda) + \log P(W'|\Gamma_{\text{new}})$$

The evaluation method of the new LM is called
Hill Climbing on Speech Lattices

1. **Initialization:** the highest scoring word sequence (the *viterbi* path) is selected from the initial lattice

2. **Neighborhood Generation:** for a selected position $i$, all paths in the lattice corresponding to word-sequences are extracted. $W' \in \mathcal{N}(W, i)$

3. **Neighborhood Rescoring:** evaluating all the word sequences in the neighborhood set, and selecting the word sequence with maximum score for the next step.

$$g(X, W'; \Lambda, \Gamma_{\text{new}}) = \alpha \log P(X|W', \Lambda) + \log P(W'|\Gamma_{\text{new}})$$

Tuesday, April 5, 2011
Hill Climbing on Speech Lattices

1. **Initialization**: the highest scoring word sequence (the viterbi path) is selected from the initial lattice.

2. **Neighborhood Generation**: for a selected position \( i \), all paths in the lattice corresponding to word-sequences are extracted.

3. **Neighborhood Rescoring**: evaluating all the word sequences in the neighborhood set, and selecting the word sequence with maximum score for the next step.

4. **Stop**: until all the positions are visited and there is no change to the current word-sequence.
Initialization: the highest scoring word sequence (the viterbi path) is selected from the initial lattice.

Neighborhood Generation: for a selected position \( i \), all paths in the lattice corresponding to word-sequences are extracted.

Neighborhood Rescoring: evaluating all the word sequences in the neighborhood set, and selecting the word sequence with maximum score for the next step.

Stop: until all the positions are visited and there is no change to the current word-sequence.
• The set of all word sequences that can be generated from $W$ with one deletion, insertion or substitution can be represented by a FSA.

• Let us call this machine $LC(W, i)$

• We will illustrate how $LC(W, i)$ can be constructed through an example
Efficient Generation of Neighborhoods

\[ W = w_1 w_2 w_3 w_4 w_5 \]
Efficient Generation of Neighborhoods

\[ W = w_1 w_2 w_3 w_4 w_5 \]

\[ LC(W, 2) \]
Efficient Generation of Neighborhoods

\[ W = w_1 w_2 w_3 w_4 w_5 \]

\[ LC(W, 2) \]

Substitutions
Efficient Generation of Neighborhoods

\[ W = w_1 w_2 w_3 w_4 w_5 \]

\[ LC(W, 2) \]

deletion
$W = w_1 w_2 w_3 w_4 w_5$

$LC(W, 2)$

insertions (to the left)
\[ W = w_1w_2w_3w_4w_5 \]

- Due to the arbitrary decision to insert only to the left of a position, we also define \( LC(W, n + 1) \) which permits insertions to the right of the last word.
Efficient Generation of Neighborhoods

\[ W = w_1 w_2 w_3 w_4 w_5 \]

\[ \text{LC}(W, 6) \]

insertions (at the end)
Efficient Generation of Neighborhoods

- To restrict the neighboring set to word sequences in the lattice (our search space), \( LC(W, i) \) is intersected with a weighted FSA representation of the lattice, \( L_{\text{acoustic}} \):

\[
LN(W, i) \leftarrow LC(W, i) \circ L_{\text{acoustic}}
\]
Efficient Generation of Neighborhoods

• To restrict the neighboring set to word sequences in the lattice (our search space), \( LC(W, i) \) is intersected with a weighted FSA representation of the lattice, \( L_{\text{acoustic}} \):

\[
LN(W, i) \leftarrow LC(W, i) \circ L_{\text{acoustic}}
\]

Acoustic scores are needed to be combined with the new LM score, according to our rescoring Eqn.
To restrict the neighboring set to word sequences in the lattice (our search space), \( LC(W, i) \) is intersected with a weighted FSA representation of the lattice, \( L_{\text{acoustic}} \):

\[
LN(W, i) \leftarrow LC(W, i) \circ L_{\text{acoustic}}
\]

represents the neighboring set, including the corresponding acoustic scores.
Local Maxima

- Our algorithm is not guaranteed to find the \textit{global maximum} and may get stuck in a local maximum solution.
  - This is true in general for hill climbing algorithms which are applied to non-convex space.

- Two common solutions to overcome this problem:
  1. **Random-restart hill climbing:** hill climbing is carried out using different random starting points.
  2. **Simulated Annealing:** unlike hill climbing it is possible to accept random moves from the neighborhood.

  S. Kirkpatrick and et. al, Science 1983
Local Maxima

• In this work, we consider random-restart technique
  • This is true in general for hill climbing algorithms which are applied to non-convex space

• Our hill climbing algorithm is repeated $M$ times, each time with a different initial word sequence
  • We will have $M$ different stopping paths along with their corresponding scores (under the new model)
  • The path with the maximum score is selected as the final output of the algorithm

• The initial paths are selected by sampling the initial lattices
  • We make sure sampled paths are not repeated
  • For the first iteration, we always start with viterbi path
Experimental Setup

• The ASR system is based on the 2007 IBM speech transcription system for GALE

• The initial lattices are generated using a 3-gram LM with Kneser-Ney smoothing
  • It has about 2.4M N-grams and is built on 400M broadcast news LM training text

• We use two different models for rescoring experiments:
  • 4-gram LM with about 64M N-grams
  • Model M shrinking based exponential LM

• Results are reported on the following sets:
  • rt04 on which the WER of initial lattices is 15.51% (using 3-gram LM)
  • dev04f with initial WER of 17.03%
Evaluation of the Efficacy

• We evaluate two different aspects of our proposed algorithm:
Evaluation of the Efficacy

- We evaluate two different aspects of our proposed algorithm:

  Comparison of the proposed hill climbing method and $N$-best rescoring based on the average number of sentence level evaluations needed for both methods to get to a particular WER.
Evaluation of the Efficacy

- We evaluate two different aspects of our proposed algorithm:

1. Comparison of the proposed hill climbing method and N-best rescoring based on the average number of **sentence level evaluations** needed for both methods to get to a particular WER

2. The algorithms are also analyzed based on how close they can get to the WER of the optimal solution (global maximum) of the rescoring model
Results

4-gram LM on rt04

Optimal WER% using 4-gram LM
Results

Model M LM on rt04

Number of Evaluations (log scale)

WER(%)
Results

4-gram LM on dev04f

Optimal WER% using 4-gram LM
Results

Model M LM on dev04f

Optimal WER% using Model M LM
The results show that our proposed method results in far fewer evaluations to reach competitive WERs, including optimal WER.
The results show that our proposed method results in far fewer evaluations to reach competitive WERs, including optimal WER.

At each step the moves are selected (from neighborhood set) based on their *quality under the new model* (in contrast to $N$-best rescoring where the evaluating points are selected based on the initial model)
The results show that our proposed method results in far fewer evaluations to reach competitive WERs, including optimal WER.

At each step the moves are selected (from neighborhood set) based on their quality under the new model (in contrast to N-best rescoring where the evaluating points are selected based on the initial model).

The problem with N-best rescoring (non-efficiency in terms of effective evaluations) is more severe when the rescoring model is different/orthogonal to the initial model.
Questions?

Thank you!