Transcribing Out-of-Vocabulary words

Elena Khasanova

University of Lorraine, IDMC, 2019

M1 NLP

Introduction

The project aims to review the state-of-the-art in the field and experiment with several types of data and conversion methods that offer **transcriptions** of Out-of-Vocabulary words (OOV).

OOV words: *words not contained in the reference dictionary of the speech recognition system.*

Why? *The size of the reference vocabulary is not limitless*

Why are they a problem?

- *OOV leave parts of the input unrecognized;*
- *OOV confuse surrounding context;*
- *OOV are often important content words;*
- *OOV affect the performance of the system.*

Solution: a system that does not depend on OOV

Joint-sequence model
Bisani&Ney (2008)

- pronunciation model + sub-lexical language model = "graphoneme" sequences estimated with expectation maximizarion algorithm;
- trained on pronunciation dictionary.
- efficient with OOV;
- can be symmetrically applied to grapheme-to-phoneme and phoneme-to-grapheme conversion.

Challenge:
beat the word-error rate **of 47%** for phoneme-to-grapheme conversion (Bisani&Ney)

Data

Carnegie-Mellon University Pronouncing Dictionary
http://www.speech.cs.cmu.edu/cgi-bin/cmudict

- 134 K words and their transcriptions
  - APRAbet symbols
  - Pronunciation variants

Example:

ACERO AH0 S EH1 R OW0
ACERO(1) AH0 S Y EH1 R OW0
ACERO(2) AH0 TH EH1 R OW0

Graphemes	Phonemes	Word length	Phonemes /word	Pronunc. / word	Words in train	Words in test
27	39	7.5	6.3	1.06	106.873	12000

TRAIN 90%	TEST 10%

Performance Metrics

Character Error Rate (CER):	Word Error Rate (WER):	Recall:
Levenshtein distance (H, R) / # total char	distance (H, R) / # total words	#in-lexicon conversions / #total conversions

*** R – reference word, H – hypothesis, output of the converter**

Experiments

One word input

Basis: *train a joint-sequence model with Sequitur trainable grapheme-to-phoneme converter up to 7-grams*

Solution 1. Dictionary lookup in n-best lists

- Output n-best options for each word
- Select the best found in the reference vocabulary
- Compute the recall

Solution 2. Apply character-based language model

Implemented and trained character-based n-gram model with Kneser-Ney smoothing

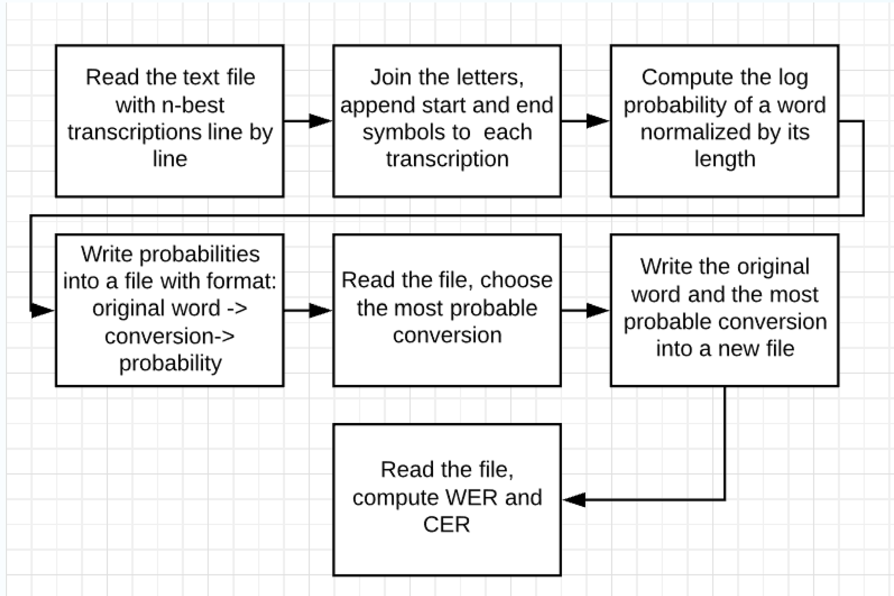
Building the model
obtain the frequency counts for the n-grams from the n-grams of the desired length to unigrams;
calculate the lambda parameter, store it in the model:

$$\lambda(w_{i-1}) = \frac{d}{c(w_{i-1})}\left|\{w : c(w_{i-1}, w) > 0\}\right|$$

Obtaining probability P(ch|hist):
start with the highest order n-gram
While not hist :
decrease hist, back off to lower order n-grams;
Recursively compute the probability according to the formula:

$$P_{KN}(w_i|w_{i-n+1}^{i-1}) = \frac{\max(c_{KN}(w_{i-n+1}^{i}) - d, 0)}{c_{KN}(w_{i-n+1}^{i-1})} + \lambda(w_{i-n+1}^{i-1}) P_{KN}(w_i|w_{i-n+2}^{i-1})$$

Select the best conversion option for each word in the test set:



Results

One word input: phoneme-to-grapheme-conversion: Dictionary lookup: Recall

P2G conversion: Recall

		Word error, %				No Conversion, %				Character error, %			
		5 best	10 best	20 best	50 best	5 best	10 best	20 best	50 best	5 best	10 best	20 best	50 best
4 gram	Full vocab	31	26	31	31	11	7	11	11	4.5	5.4	6	6.6
	Test vocab	19	20	20	20	18	18	18	18	4.5	5.4	6	6.6
7 gram	Full vocab	32	32	28	32	9	9	3	9	2.1	2.8	4.2	6
	Test vocab	18	18	9	18	16	16	6	16	2.1	2.8	4.2	6

One word input: phoneme-to-grapheme-conversion: estimation with Kneser-Ney smoothed character-based language model

P2G conversion:

Precision using the LM with Kneser-Ney smoothing, d = 0.75

	Word error, %				Character error, %			
	5 best	10 best	20 best	50 best	5 best	10 best	20 best	50 best
4 gram	63	70	76	85	8	9	10	11
5 gram	47	51	49	65	6	6	6	8
7 gram	35	36	39	47	4	4	5	6
8 gram	35	36	39	46	4	4	5	6
9 gram	35	36	39	46	4	4	5	6

Two words and a word boundary

Train two-word model

Modify the data to take 2 words with a boundary symbol and their corresponding pronunciation, train joint-sequence model with Sequitur

Grapheme to phoneme conversion baseline results:

Model	WER, %	CER, %
4 gram	70	11.83
7 gram	68	10.87

Solution 1. N-best lists and character based language model
Obtain n-best lists with two-word model (7-gram), choose the best conversion using character-based language model trained on two-words data

Results:

Model	WER, %	CER, %
5 best	82	10.1
10 best	84	10.25
20 best	86	10.85

Experiments

Solution 2. Trying all word boundaries

- insert a word boundary symbol at every possible place in the
- phonemic sequence
- for each pair of words:
  - perform the conversion with Sequitur trained on single words
  - compute the probability of a word according to the language model
  - compute the joint probability of two words
- select the best sequence
- write the best sequence into a file
- check if the resulting words are contained in the vocabulary / use the language model

Multiple words input

- Try all permutations of substrings on :
- (1) phonemic sequence / (2) converted letter sequences
- for each group of words:
  - perform the conversion with Sequitur trained on single words for (1), skip for (2)
  - compute the probability of a word according to the language model
  - compute the joint probability of a group of words
- select the best sequence
- write the best sequence into a file
- check if the resulting words are contained in the vocabulary / use the language model

Conclusion

- Kneser-Ney character-based language model helps to decrease error rate by 10-12\%;
- Error rate drops systematically with the increase of the order up to the average word length;
- 5 to 10 conversion results seem to give the best variation to improve accuracy;
- The model trained with a word boundary helps to determine it in a sequence of two words;
- The model trained with a word boundary does not seem to be efficient in handling conversions;
- Trying all possible word boundaries is time and memory consuming and doesn't seem to be promising.

References:

Ney H. Bisani M. “Joint-Sequence Models for Grapheme-to-Phoneme Conversion”. In: Speech Communication (2008). doi : 10.1016/j.specom.2008.01.00