# Introduction to SRILM Toolkit

Kuan-Yu Chen
Department of Computer Science & Information Engineering
National Taiwan Normal University

Main Reference：

1. A. Stolcke, “SRILM - An Extensible Language Modeling Toolkit”, in Proc. Intl. Conf. Spoken Language Processing, Denver, Colorado, September 2002.
2. S. F. Chen and J. Goodman, “An Empirical Study of Smoothing Techniques for Language Modeling”, Tech. Report TR-10-98, Computer Science Group, Harvard U., Cambridge, MA, August 1998.
3. SRILM Manual Pages: http://www.speech.sri.com/projects/srilm/manpages/

# Available Web Resources

- SRILM: " http://www.speech.sri.com/projects/srilm/ "
  - A toolkit for building and applying various statistical language models (LMs)
  - Current version: 1.5.8
  - Can be executed in Linux environment

- Cygwin: "http://www.cygwin.com/ "
  - Cygwin is a Linux-like environment for Windows
  - Current version: 1.5.25-15

# *Steps for Installing Cygwin*

1. Download the cygwin installation file "**setup.exe"** from the website
2. Run `setup.exe`
3. Choose "`Install from Internet`" (or others)
4. With a default setting, it will be installed in "**c:\cygwin**"
5. "Local Package Directory" means the temporary directory for packages
6. Choose a downloadable (mirror) website

# *Steps for Installing Cygwin (cont.)*

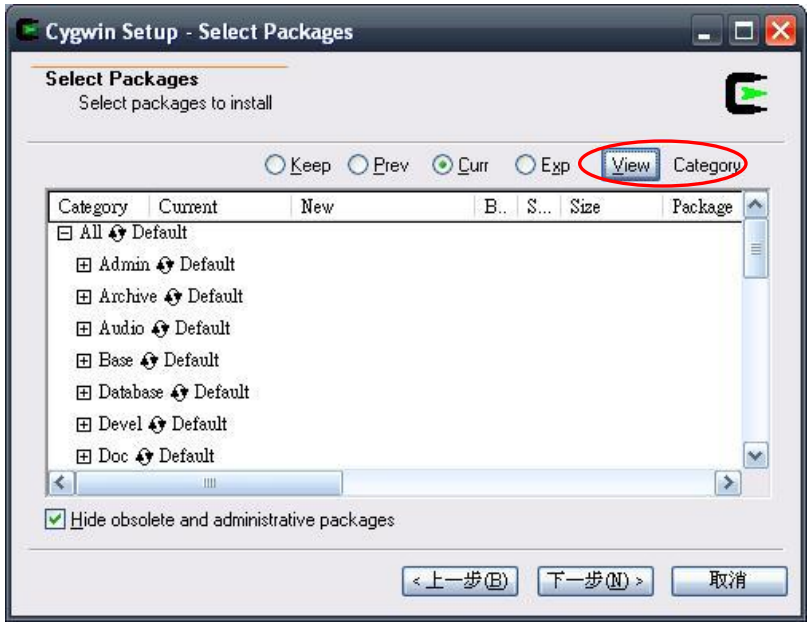
7. Note that:

   If you want to compile original source code

   Change Category “View” to Full

   Check if the packages **“binutils”, “gawk”, “gcc”, “gzip”, “make”, “tcltk”, “tcsh”** are selected

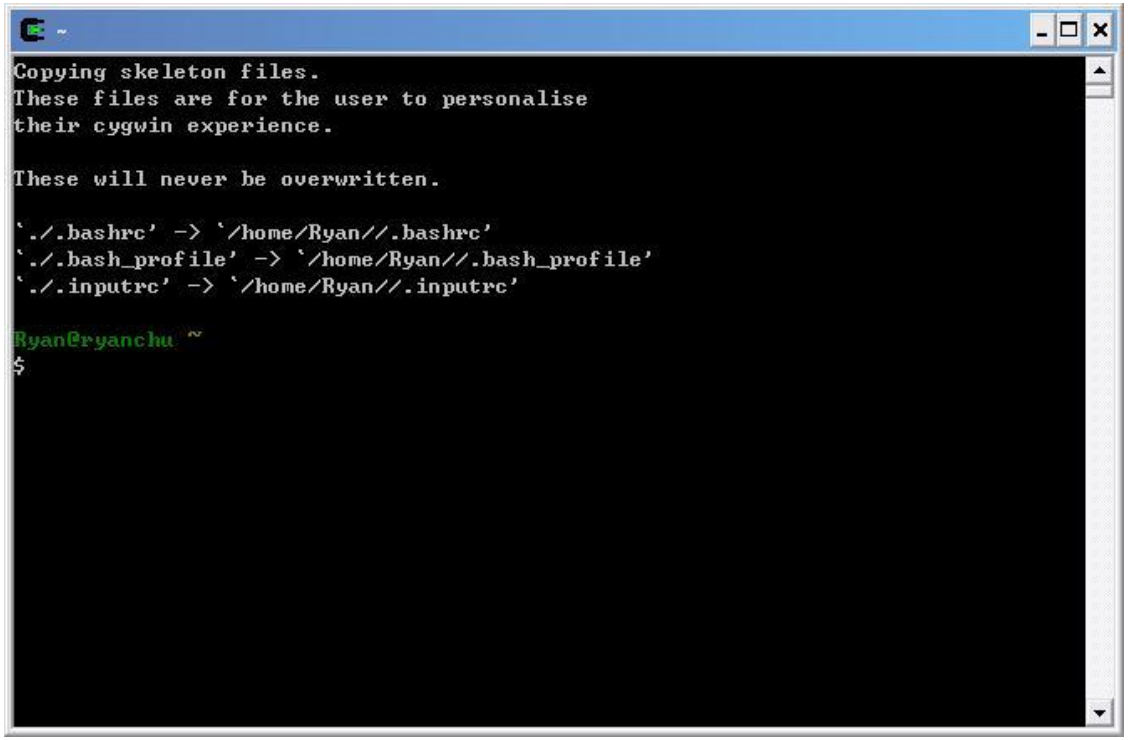
   If not, use the default setting



# *Steps for Installing Cygwin (cont.)*

8. After installation, run cygwin

   It will generate "**.bash_profile**", "**.bashrc**", "**.inputrc**" in "**c:\cygwin\home\yourname**"



# *Steps for Installing SRILM Toolkit*

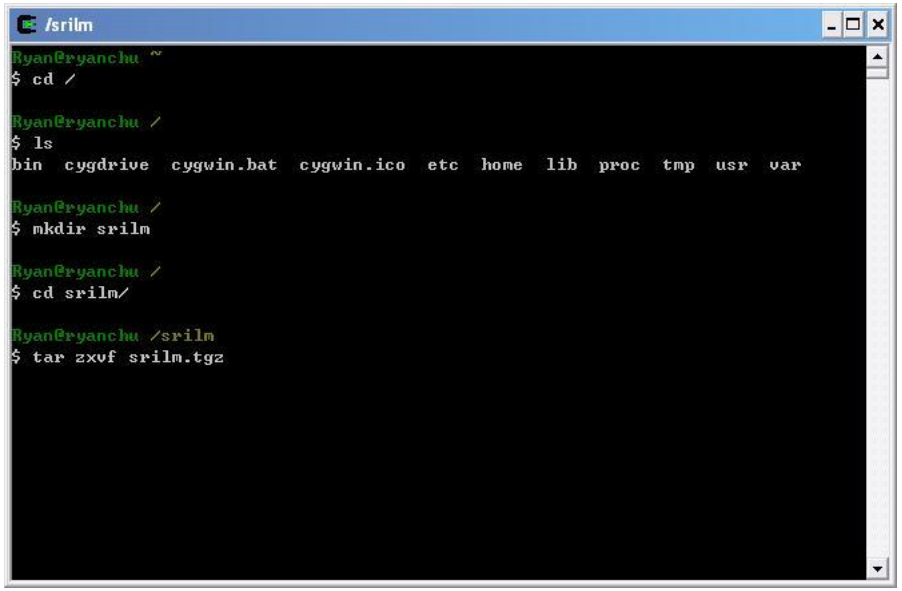
Now we then install "**SRILM**" into the "**Cygwin**" environment

1. Copy "**srilm.tgz**" to "**c:\cygwin\srilm**"
   - Create the "**srilm**" directory if it doesn't exist
   - Or, merely copy "**srilm.zip**" to c:\cygwin

2. Extract "**srilm.tgz**"

commands in cygwin:

```
$ cd /
$ mkdir srilm      //create the "srilm" directory
$cd srilm
$ tar zxvf srilm.tgz   //extract srilm.tgz
```



# *Steps for Installing SRILM Toolkit (cont.)*

3. Edit “c:\cygwin\home\yourname\.bashrc”
   - Add the following several lines into this file

```
export SRILM=/srilm
export MACHINE_TYPE=cygwin
export PATH=$PATH:$pwd:$SRILM/bin/cygwin
export MANPATH=$MANPATH:$SRILM/man
```

4. Edit “c:\cygwin\srilm\Makefile”
   - Add a line: “**SRILM = /srilm”** into this file

```
SRILM = /srilm
#
# Top-level Makefile for SRILM
#
# $Header: /home/srilm/devel/RCS/Makefile,v 1.48 2008/10/08 00:12:21 sto…
…
```

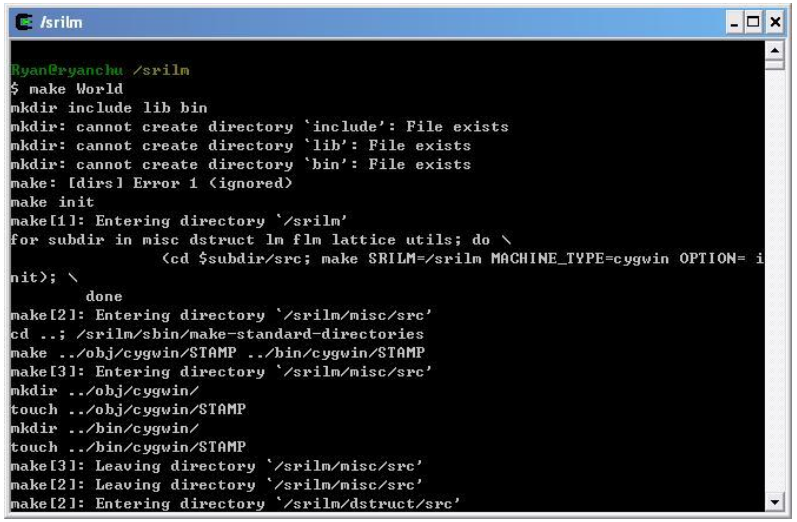
# *Steps for Installing SRILM Toolkit (cont.)*

5. Compile the SRILM source code files
   - Run cygwin
   - Switch current directory to "**/srilm**"
   - Execute the following commands

```
$ make World
```

   - Copy "**c:\cygwin\srilm\bin\make-big-lm**", to "**c :\cygwin\srilm\bin\cygwin**"
   - Execute the following commands

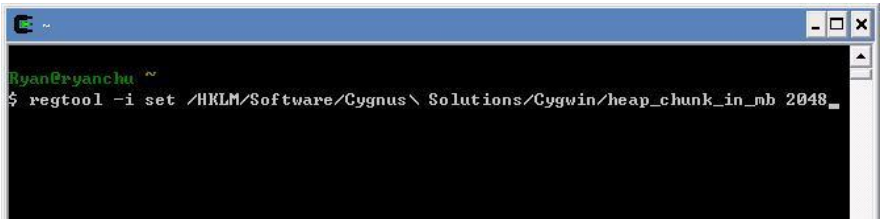
```
$ make all
$ make cleanest
```



# Environmental Setups

- Change cygwin's maximum memory
  - Referred to: " http://cygwin.com/cygwin-ug-net/setup-maxmem.html "

```
regtool -i set /HKLM/Software/Cygnus\ Solutions/Cygwin/heap_chunk_in_mb 2048
```




- Use Chinese Input In Cygwin
  - Manually edit the "**.bashrc**" and "**.inputrc**" files in "c**:\cygwin\home\yourname**"

.bashrc

```
export LESSCHARSET=latin1
alias ls="ls --show-control-chars"
```

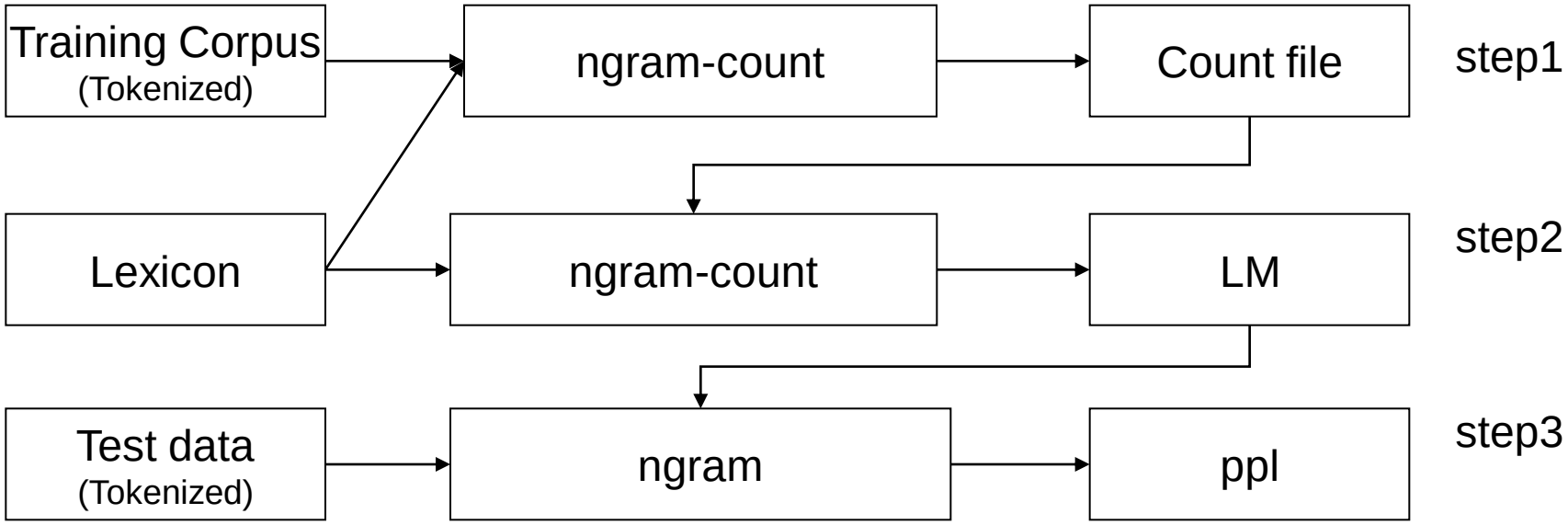
.inputrc

```
set meta-flag on
set convert-meta off
set output-meta on
set input-meta on
```

  - Referred to: " http://cygwin.com/faq/faq_3.html#SEC48 "

# *Functionalities of SRILM*

- Three Main Functionalities
  - Generate the n-gram count file from the corpus
  - Train the language model from the n-gram count file
  - Calculate the test data perplexity using the trained language model



# *Format of the Training Corpus*

- Corpus: e.g., “CNA0001-2M.Train” (56.7MB)
  - Newswire Texts with Tokenized Chinese Words

中華民國 八 十九 年 一月 一 日
萬
黃 兆 平
面對 這個 歷史性 的 時刻
由 中國 電視公司
昨晚 在 中 正紀念堂 吸引 了 超過 十 萬 人潮
共同 迎接 千禧 年
勤奮 努力
欣欣向榮 外
……

# *Format of the Lexicon*

- Lexicon: “Lexicon2003-72k.txt”

巴
八
扒
叭

墨竹
默祝
末梢
沒收
墨守
陌生
……

- Vocabulary size: 71695
- Maximum character-length of a word: 10

# *Generating the N-gram Count File*

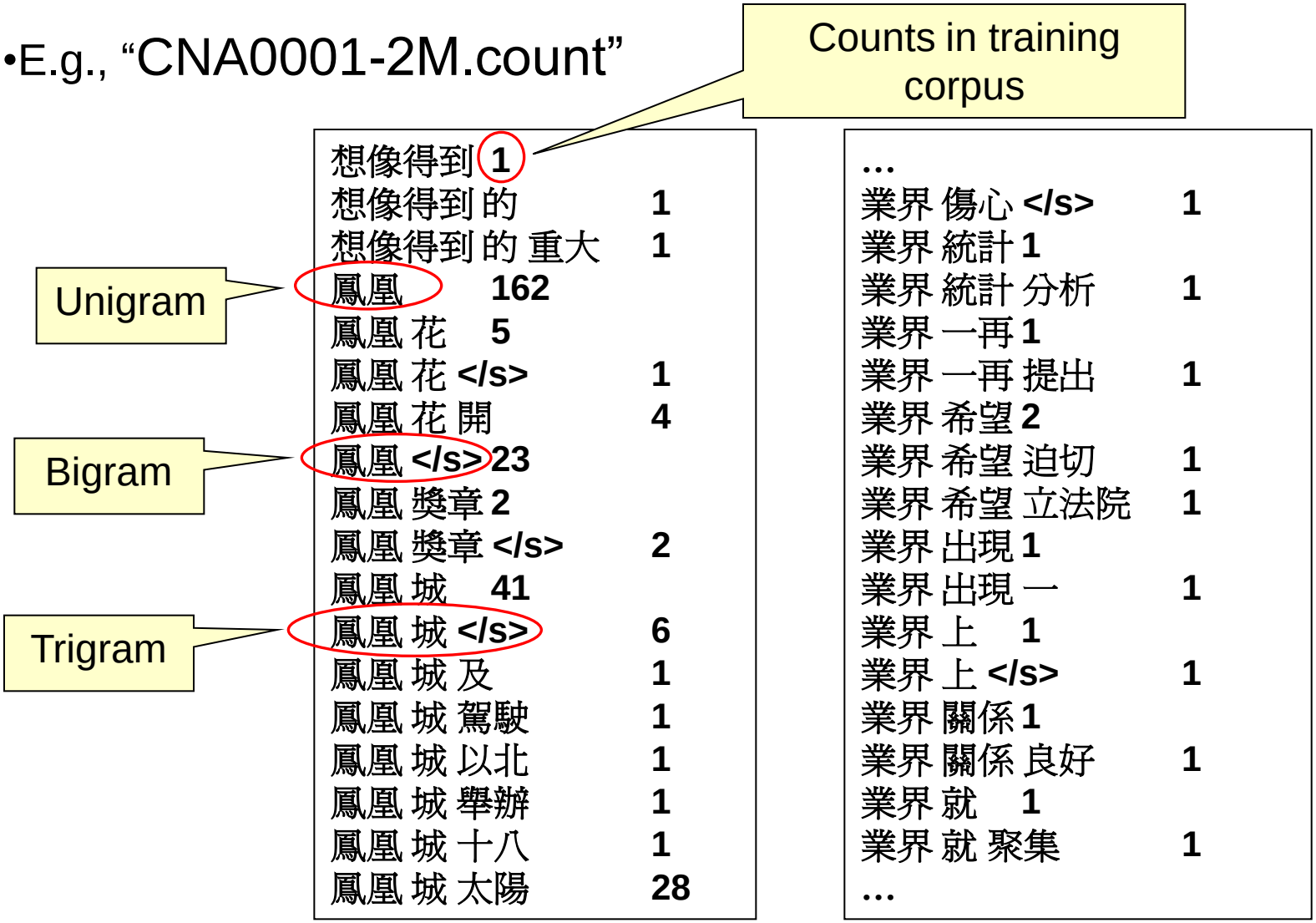
- Command

  ngram-count -vocab Lexicon2003-72k.txt
  -text CNA0001-2M.Train
  -order 3
  -write CNA0001-2M.count
  -unk

  – Parameter Settings
  -vocab: lexicon file name
  -text: training corpus name
  -order: n-gram count
  -write: output countfile name
  -unk: mark OOV as <unk>

# *Format of the N-gram Count File*

- E.g., "CNA0001-2M.count"



# *Generating the N-gram Language model*

- Command

  ngram-count -vocab Lexicon2003-72k.txt
  -read CNA0001-2M.count
  -order 3
  -lm CNA0001-2M_N3_GT3-7.lm
  -gt1min 3 -gt1max 7
  -gt2min 3 -gt2max 7
  -gt3min 3 -gt3max 7

  – Parameter Settings

    -read: read count file

    -lm: output LM file name

    -gt**n**min: Good-Turing discounting for **n**-gram

# *Format of the N-gram Language Model File*

- E.g., “CNA0001-2M_N3_GT3-7.lm”

```
\data\
ngram 1=71697
ngram 2=2933381
ngram 3=1205445

\1-grams:
-0.8424806              </s>
-99             <s>     -1.291354
-2.041174       一      -1.287858
-3.804316       一一    -0.8553778
-5.374712       一一恐怖        -1.269383
-4.772653       一一恐怖攻擊            -0.8950238
-9.690391       一丁點
-3.51804        一九九  -2.89049
-7.180892       一了百了        -0.1229095
-6.481923       一刀兩斷        -0.6672484
-4.802495       一下    -0.4828814
```

Log of backoff weight (Base 10)

```
-1.38444    <s> 裏 表現
-1.38444    <s> 裏 面
-1.076253   <s> 裏 海
-0.624772   戈 裏 峰
-0.624772   年 裏 </s>
-1.198803   那 裏 </s>
-0.3165856              哪 裏 去
-0.7112821              家 裏 的
-1.323742   家 裏 開
-0.4998333              時間 裏 </s>
-0.3147101              眼 裏 </s>
-0.323742   過程 裏 </s>
-0.721682   <s> 恒 生
-0.323742   億 恒 科技
-0.1760913              化 粧 品

\end\
```

Log probability (Base 10)

# *Calculating the Test Data Perplexity*

- Command:

  ngram -ppl 506.pureText
  -order 3
  -lm CNA0001-2M_N3_GT3-7.lm

  – Parameter Settings
  -ppl: calculate perplexity for test data

file 506.PureText: 506 sentences, 38307 words, 0 OOVs
0 zeroprobs, logprob= -117172 ppl= 1044.42 ppl1= 1144.86

$10^{-\frac{\text{logprob}}{\#\text{Sen}+\#\text{Word}}}$ $10^{-\frac{\text{logprob}}{\#\text{Word}}}$

# *Other Discounting Techniques*

- Absolute Discounting

```
ngram-count -vocab Lexicon2003-72k.txt
            -read CNA0001-2M.count
            -order 3
            -lm CNA0001-2M_N3_AD.lm
            -cdiscount1 0.5
            -cdiscount2 0.5
            -cdiscount3 0.5
```

- Witten-Bell Discounting

```
ngram-count -vocab Lexicon2003-72k.txt
            -read CNA0001-2M.count
            -order 3
            -lm CNA0001-2M_N3_WB.lm
            -wbdiscount1
            -wbdiscount2
            -wbdiscount3
```

# *Other Discounting Techniques (cont.)*

- Modified Kneser-Ney Discounting

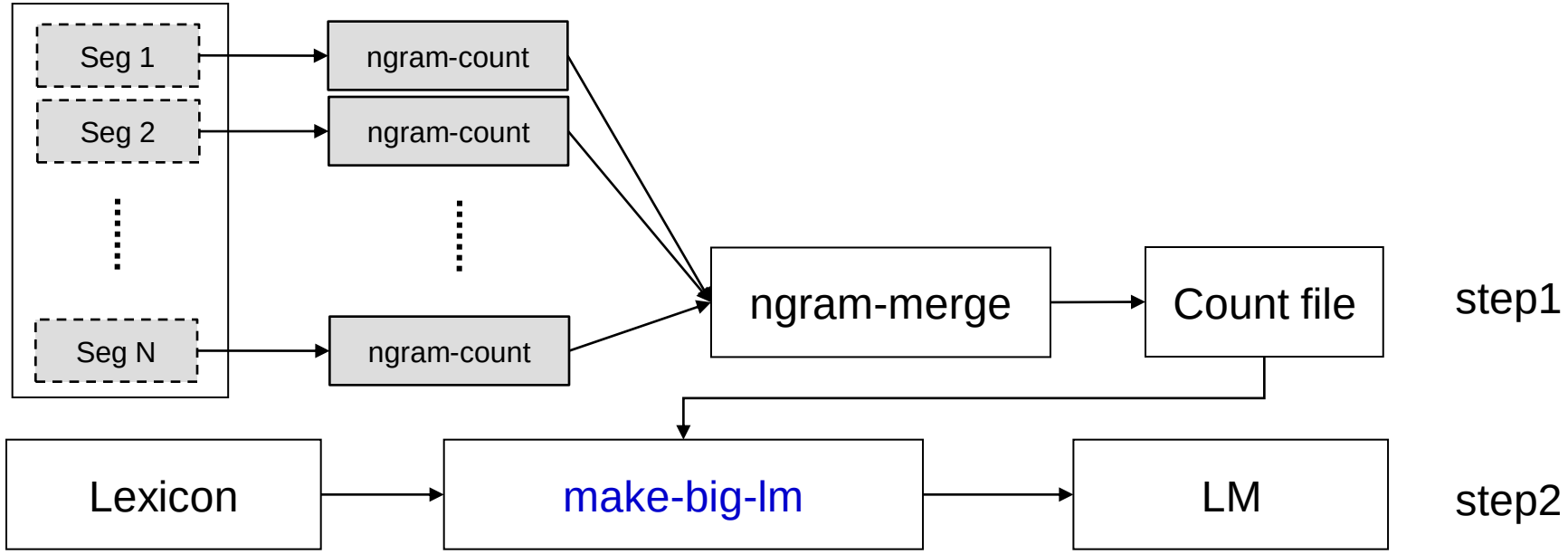
```
ngram-count -vocab Lexicon2003-72k.txt
            -read CNA0001-2M.count
            -order 3
            -lm CNA0001-2M_N3_KN.lm
            -kndiscount1
            -kndiscount2
            -kndiscount3
```

- Available Online Documentation:

  " http://www.speech.sri.com/projects/srilm/manpages/ "

# *Large Data and High-Order Language Model*

- "**make-big-lm**" constructs large N-gram models in a more memory-efficient way than **ngram-count** by itself.
- Smoothing methods other than Good-Turing and modified Kneser-Ney are not supported by **make-big-lm**.



# *Generating the N-gram Count File*

- Command

```
ngram-count -vocab Lexicon2003-72k.txt
            -text CNA0102_seg1.Train
            -order 5
            -write CNA0102_seg1.count
            -unk
            -sort
```

  – Parameter Settings

    -vocab: lexicon file name

    -text: training corpus name

    -order: n-gram count

    -write: output countfile name

    -unk: mark OOV as <unk>

    -sort: output counts in lexicographic order

# *Generating the N-gram Count File (cont.)*

- Command

  ngram-merge -write CNA0102_n5.count
  CNA0102_seg1.count
  CNA0102_seg2.count
  CNA0102_seg3.count
  …

  – Parameter Settings

    -write: output merged count file name

- It is best to merge the resulting sorted counts using **ngram-merge** pairwise, and continue doing so in a binary tree pattern until a single count file containing all N-grams remains.

# *Generating the N-gram Language model*

- Command

```
make-big-lm  -vocab Lexicon2003-72k.txt
             -read CNA0102_n5.count
             -order 5
             -lm CNA0102_n5_GT3-7.lm
             -gt1min 3 -gt1max 7 -gt2min 3 -gt2max 7
             -gt3min 3 -gt3max 7 -gt4min 3 -gt4max 7
             -gt5min 3 -gt5max 7
```

  - Parameter Settings

    -read: read count file

    -lm: output LM file name

    -gt**n**min: Good-Turing discounting for **n**-gram

# *Other Discounting Techniques*

- Modified Kneser-Ney Discounting

```
make-big-lm  -vocab Lexicon2003-72k.txt
             -read CNA0102_n5.count
             -order 5
             -lm CNA0102_n5_KN.lm
             -kndiscount1
             -kndiscount2
             -kndiscount3
             -kndiscount4
             -kndiscount5
```

  - Kneser-Ney smoothing also requires enough disk space to compute

# *Take-home Exercise*

- Practice for use of SRILM toolkit
  - Training corpus (tokenized text)
    - TrainingCorpus.txt
    - 8.8MB
    - 50,000 sentences
  - Lexicon
    - Lexicon2003-72k.txt
    - Vocabulary size: 71695
    - Maximum character-length of a word: 10
  - Test corpus (tokenized text)
    - TestCorpus.txt
    - 3.0MB
    - 16,667 sentences

# Take-home Exercise

- Three major steps
  - Step1: Generating the N-gram Count File (3-gram)

```
ngram-count -order 3 -vocab Lexicon2003-72k.txt
            -text TrainingCorpus.txt   -write TrainingCorpus.count
```

  - Step2: Generating the N-gram Language model
    - Step2-1: Good-Turing Smoothing

```
ngram-count -order 3 -vocab Lexicon2003-72k.txt
            -read TrainingCorpus.count  -lm TrainingCorpus.GT3-7.lm
            -gt1min 3 -gt1max 7 -gt2min 3 -gt2max 7 -gt3min 3 -gt3max 7
```

    - Step2-2: Kneser-Ney Discounting

```
ngram-count -order 3 -vocab Lexicon2003-72k.txt
            -read TrainingCorpus.count  -lm TrainingCorpus.KN3.lm
            -kndiscount1 -kndiscount2 -kndiscount3
```

# Take-home Exercise

- Step2: Generating the N-gram Language model
  - Step2-3: Witten-Bell Discounting

    ngram-count -order 3 -vocab Lexicon2003-72k.txt
    -read TrainingCorpus.count -lm TrainingCorpus.WB3.lm
    -wbdiscount1 -wbdiscount2 -wbdiscount3

- Step3: Calculating the Test Data Perplexity
  - Step3-1: Using Good-Turing Smoothing

    ngram -ppl TestCorpus.txt -order 3 -lm TrainingCorpus.GT3-7.lm

    file TestCorpus.txt: 16655 sentences, 695353 words, 0 OOVs
    0 zeroprobs, logprob= -1.90411e+06 ppl= 472.375 ppl1= 547.445

  - Step3-2: Using Kneser-Ney Discounting

    ngram -ppl TestCorpus.txt -order 3 -lm TrainingCorpus.KN3.lm
  - Step3-3: Using Witten-Bell Discounting

    ngram -ppl TestCorpus.txt -order 3 -lm TrainingCorpus.WB3.lm