Introduction to SRILM Toolkit

Department of Computer Science & Information Engineering
National Taiwan Normal University

Hank Hao

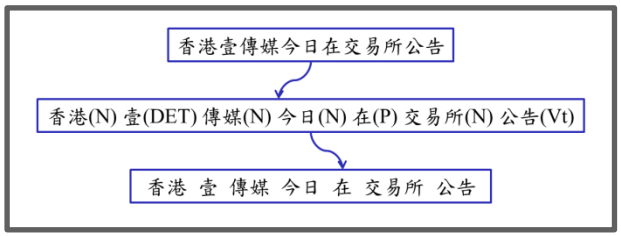
2013/12/11

Installing SRILM

- SRILM
  - Stanford Research Institute Language Modeling Toolkit
  - SRILM is a toolkit for building and applying statistical language models (LMs), primarily for use in speech recognition, statistical tagging and segmentation, and machine translation
  - http://www.speech.sri.com/projects/srilm/download.html
- Steps
  1. Download and unpack the SRILM toolkit
  2. Unzip the below solution archive to the SRILM root directory
  3. Load the srilm.sln in the **Visual Studio** environment
  4. Select either **Debug** or **Release** mode to build projects ->Configuration Manager
  5. Right-click on each project in the folder of libs, and select **Build**
  6. Build any of the command-line utilities that you need

Training a Language Model

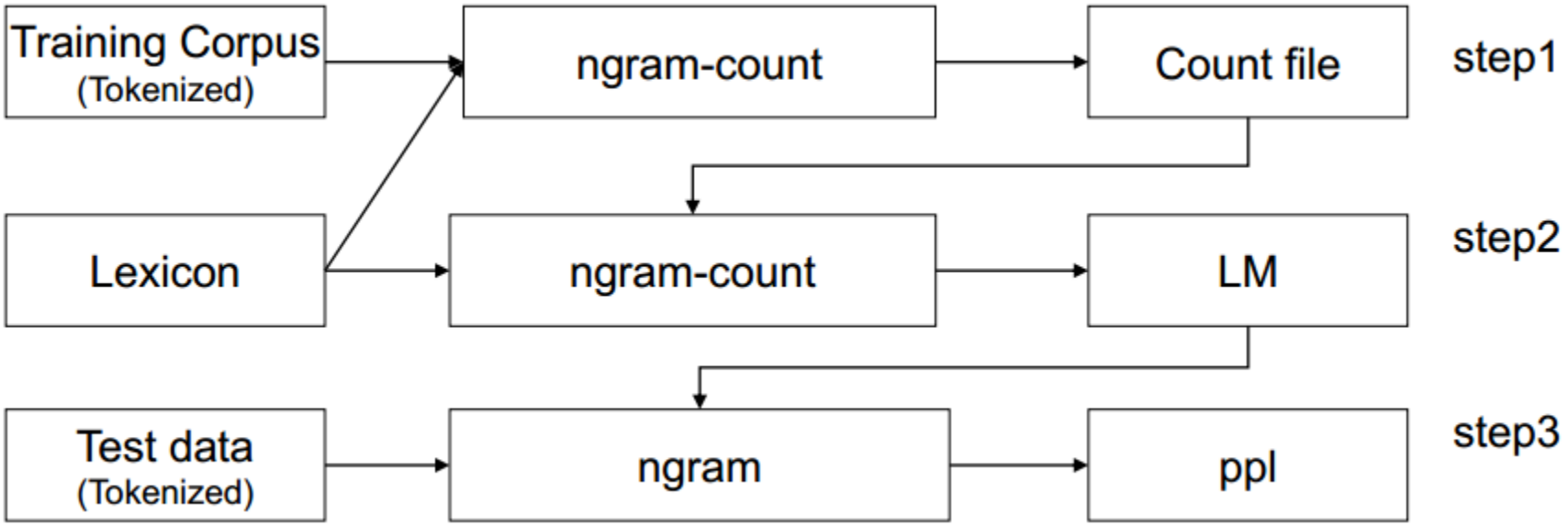
- Tokenization
  - It is the process of breaking up a stream of text into words, phrases, symbols or other meaningful elements (called tokens)
- Chinese Knowledge Information Processing (CKIP)
  - It is an online package developed by Institute of Information Science, Academia Sinica: http://ckipsvr.iis.sinica.edu.tw/



Human-Generated Text → Tokenization → Computer-Readable Text

Training a Language Model

- 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



Training a Language Model

- 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 Dictionary & Count Files

- Lexicon2003-72k.txt

Dictionary

```
巴
八
扒
叭

變小
變心
變相
變形

鎮定劑
振動器
鎮痛劑
...
```

- CNA0001-2M.count

Count File

```
鳳凰      1
鳳凰 公視          1
鳳凰 公視 新聞     1
希特勒    2
希特勒 統治        1
希特勒 統治 的     1
希特勒 過去        1
希特勒 過去 曾經           1
舉止      2
舉止 其實          1
舉止 其實 可以     1
```

Counts in training corpus

Training a Language Model (with Good-Turing Smoothing)

- 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

    -gtnmin: Good-Turing discounting for n-gram

Training a Language Model (with Kneser-Ney Smoothing)

- Generating the *N*-gram Language model
  - Command:

```
ngram-count  -order 3
             -vocab Lexicon2003-72k.txt
             -read CNA0001-2M.count
             -lm CNA0001-2M_N3_KN3.lm
             -kndiscount1
             -kndiscount2
             -kndiscount3
```

  - Parameter Settings

    -read: read count file

    -lm: output LM file name

    -kndiscountn: Use Kneser-Ney discounting for N-grams of order *n*.

Calculating the Perplexity of Test Data

- Perplexity Calculation
  - Command:

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

  - Parameter Settings

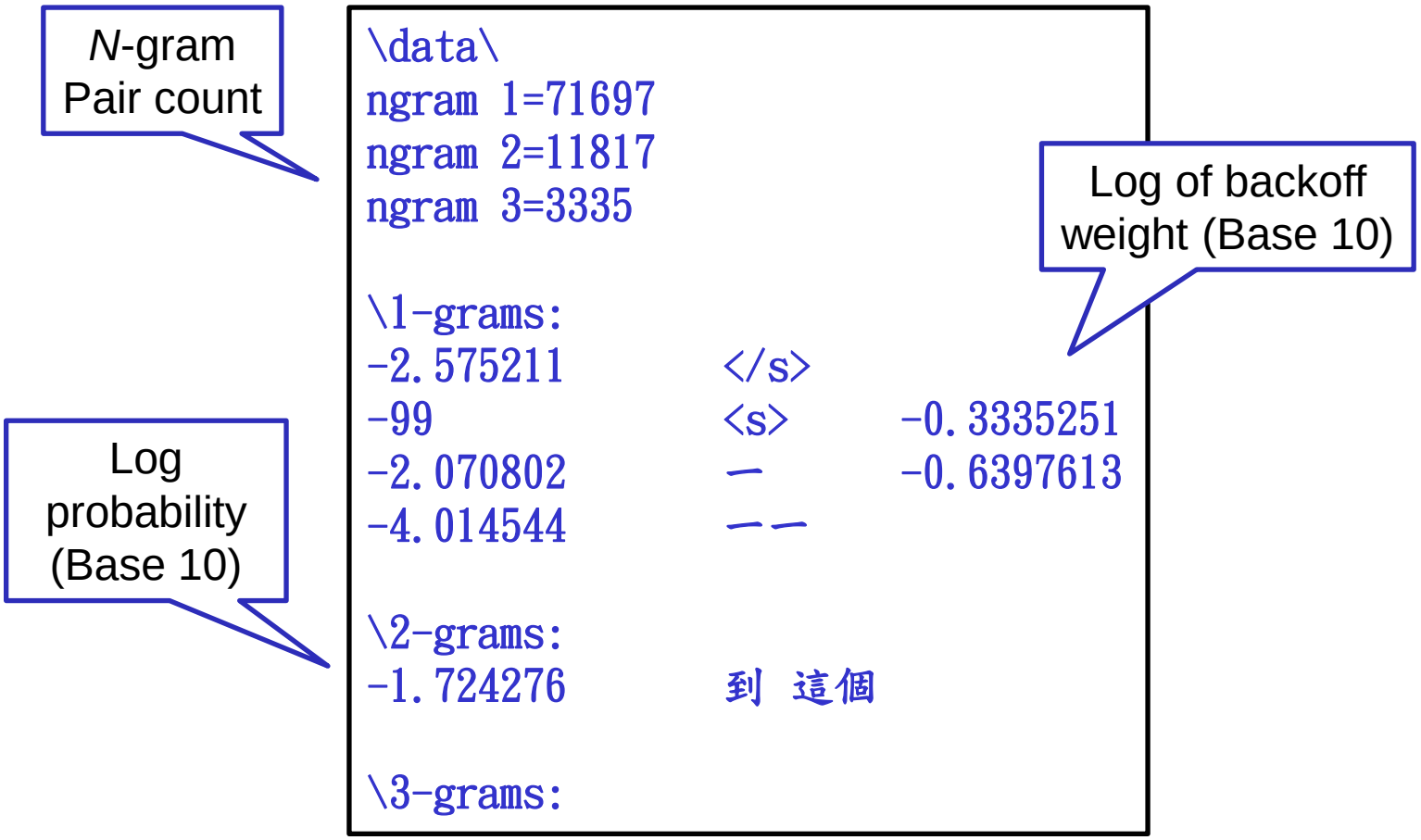
    -ppl: calculate perplexity for test data

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

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

Format of the *N*-gram Language Model File

- CNA0001-2M_N3_GT3-7.lm



Homework

- **Corpus link :** http://goo.gl/C3C0wo
- **Goal:** Document classification based on perplexity calculations
- **Step :**

1. Train an LMs for each set of documents with the SRILM toolkit and different LM settings
2. Calculate the perplexity for each test document and assign it into corresponding class
3. Evaluation the classification accuracy of each method

	Unigram	Bigram	Trigram
Acc(%)	?	87.00	?

Appendix

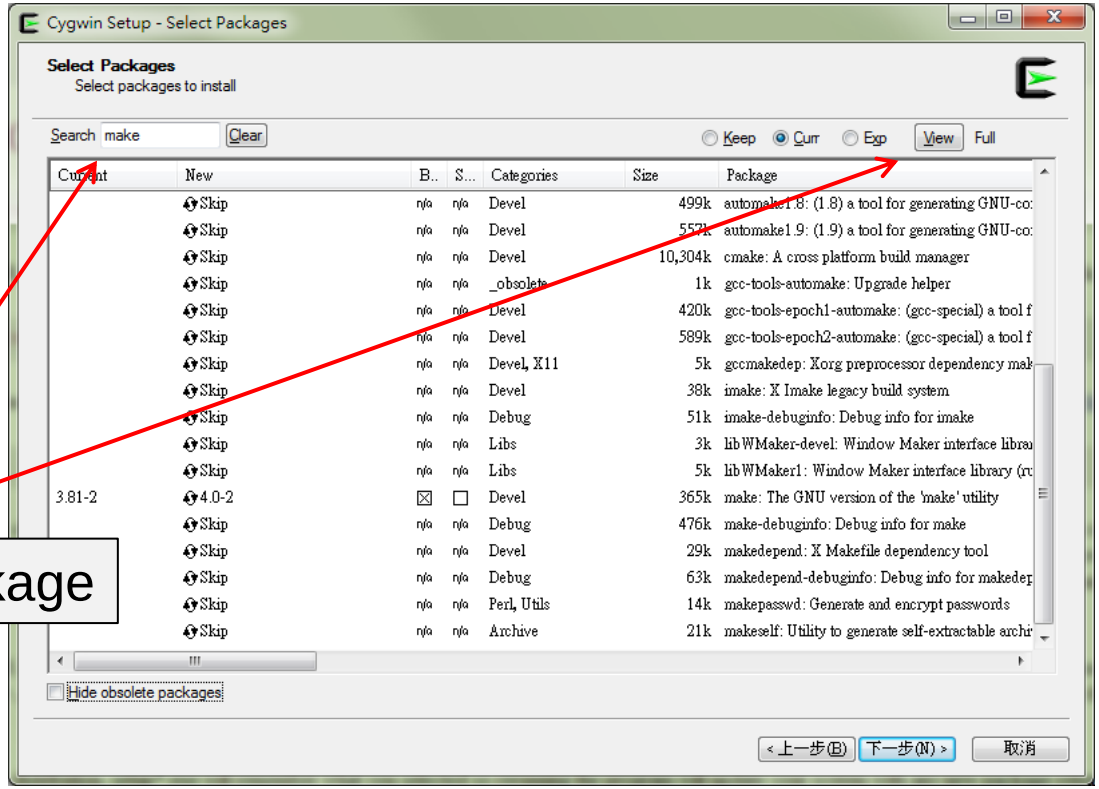
Available Web Resources

- **Cygwin**
  - A Linux-like environment for Windows making it possible to port software running on POSIX systems (such as Linux, BSD, and Unix systems) to Windows
  - http://cygwin.com/install.html

- **SRILM**
  - SRILM is a toolkit for building and applying statistical language models (LMs), primarily for use in speech recognition, statistical tagging and segmentation, and machine translation
  - http://www.speech.sri.com/projects/srilm/download.html

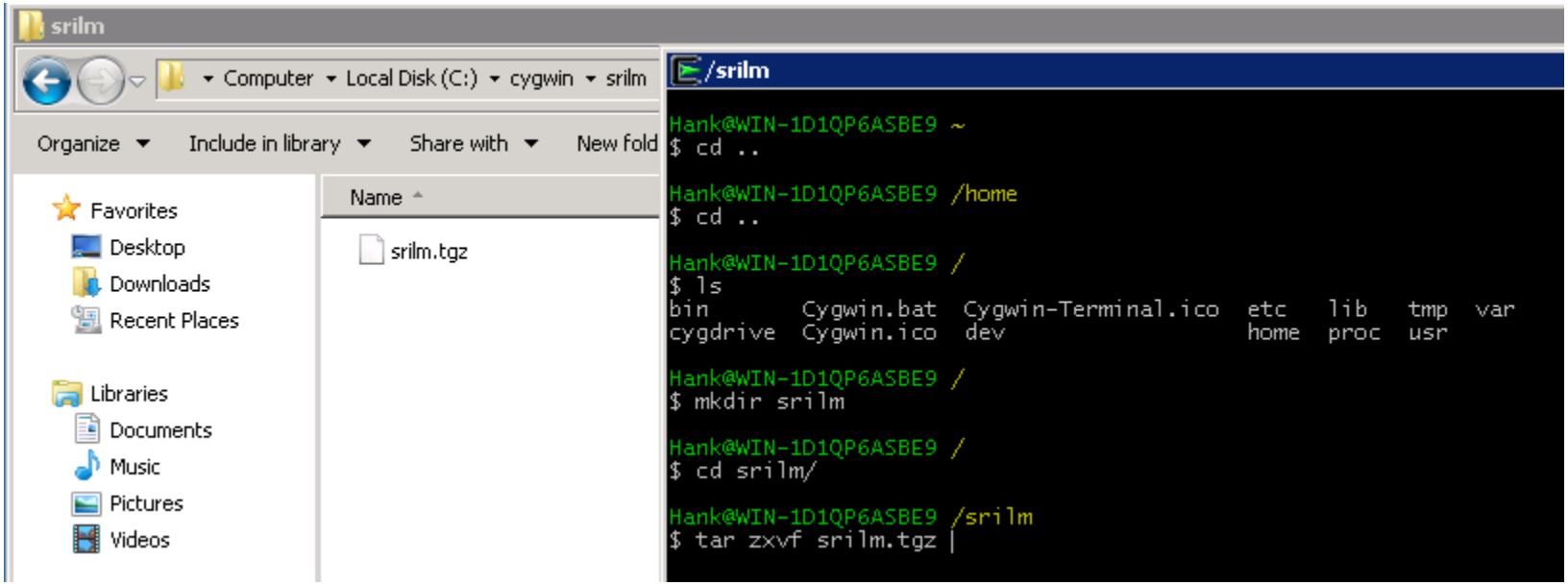
Installing Cygwin

- While installing, make sure all packages that “**SRILM**” needs are selected
  - “**binutils**”, “**gawk**”, “**gcc**”, “**gzip**”, “**make**”, “**tcltk**”, “**tcsh**”



Installing SRILM

- Now we need to install "**SRILM**" into cygwin environment
  1. Create the "**srilm**" directory under C:\cygwin
  2. Copy the compressed file of SRILM in it
  3. Extract "**srilm.tgz**"



Installing SRILM

- Edit “c:\cygwin\home\yourname\.bashrc”

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

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

```
SRILM = /srilm
#
# Top-level Makefile for SRILM#
# $Header: …
```

Installing SRILM

- Compile the SRILM source code files
  - 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
```

- Use Chinese Input In Cygwin

.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
```