CenTaD - SiS Project

Disarranged Sentence Reconstruction

Wang Shuwei, Hwa Chong Institution; Mentored by Dr Wang Xuancong, Institute for Infocomm Research

Introduction

In this project, a popular task in many primary school Chinese exams, disarranged sentence reconstruction（词句重组）, is studied using the machine learning approach. This task mainly requests the students to reorder and re-arrange a list of Chinese words in random order to form a grammatically correct and meaningful sentence.

村庄 浓密的 小湖 树林里 那边的 掩隐 在

→ 小湖那边的村庄掩隐在浓密的树林里

The project's aim is to design a computer algorithm which can automatically solve such problems. To fit the amount of work within the constraint of the project, we currently limit our vocabulary to those words which commonly occur in primary school texts.

The final deliverable of this project is a Python program that run at the back-end of a website. You can input to the program disarranged word lists through the webpage, and view the answers generated by the program. Scan the QRCode on the right to access the website, or directly go to the address:

http://tinyurl.com/centaddsr

Our project attempts to restore/recover the word order information in natural languages. Such information can be of crucial importance to various natural language processing applications. For example, this algorithm, if fully trained with a larger scope of text data, may be useful as an add-on that can increase the performance of our current machine translation systems by correcting the words order in the translation result.

程序可以自动解决此类问题具有较高的精度。

Expected: 程序可以自动以较高的精度解决此类问题。

Models

Word-based N-gram language model

P($(i)^{th}$ word = "xxx" | $(i - n \sim i - 1)^{th}$ words = "xxx xxx xxx")

我 正在 看 小说
他 最 喜欢 的 是 看 书
我们 要 多 看 书 才能 长 知识

P($(i)^{th}$ = "书" | $(i - 1)^{th}$ = "看") = 2/3

Word-based Backward N-gram language model

P($(i)^{th}$ word = "xxx" | $(i + 1 \sim i + n)^{th}$ words = "xxx xxx xxx")

这 真是 一朵 漂亮 的 花
那 本 书 有着 漂亮 的 封面
她 穿 得 真 漂亮 呢

P($(i)^{th}$ = "漂亮" | $(i + 1)^{th}$ = "的") = 2/3

Char-based N-gram language model

Use characters as basic units to consider the similarities between different words, hence account for rare words

他 高兴 地 跳了 起来
小明 高兴 地 笑了

High Possibility

→ 大家 高高兴兴 地 去 上学

High Possibility

POS-based N-gram language model

Use part-of-speech instead of words to consider the similarities between different words, hence account for rare words

(Assume the program does not know the word □ ዛ)

我 喜欢 阅读 故事	???
n. + v. + v. + n.	High Possibility

Bigram Occurrence language model

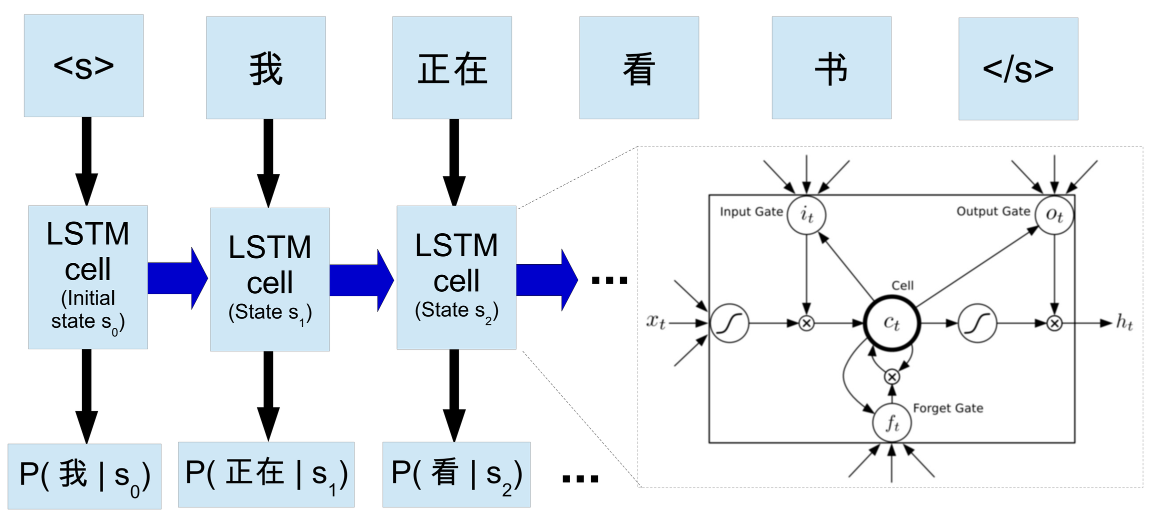
Emphasis on common collocations like " 数学 书"

$n(bigram)$ = occurrence of $bigram$ in data

$\mathbf{score}(bigram) = [\log n(bigram)]^2 + 1$

Recurrent Neural Network language model (RNN)

Long Short-Term Memory (LSTM) based RNN can capture the entire history context



Data Collection

Large amount of Chinese text for the training and verification of statistical models

Essays → In-domain Training Data

I²R Resources → In-domain Training Data; Out-domain Training Data

Real-world Questions → Tuning Data; Testing Data

Chinese Primary School Essays:

- ~11M characters original, filtered to 340k sentences.
- Crawled from " 作文网" (www.zuowen.com) with Python script

I²R Resource

- 15M sentences that are partially relevant

Real-world 词句重组 questions and answers

- 2033 sentences, split into 1500 lines of tuning data and 533 lines of testing data
- Collected from Chinese education websites and digital versions of exam papers online

Programming

In order to determine the best hypothetical answer to a task, we maximise a score, which is a weighted summation of scores in different models

$$score(hypothesis) = w_1 s_1 + w_2 s_2 + w_3 s_3 + \ldots + w_n s_n$$

The maximum is approximated by adding words to the hypothesis one by one, using the following pseudocode:

```
function generate_hypothesis:
   init hypos to empty array
   hypos.append empty string
   for each index in 1 to n:
      init new_hypos to empty array
      for each hypothesis in hypos:
         for each word in hypothesis.remaining_words:
            new_hypos.append (new hypothesis)
         end
      end
      new_hypos.sort by score
      if new_hypos.contains more than 100 items:
         set hypos to first 100 items in new_hypos
      else:
         set hypos to all items in new_hypos
      end
   end
   return first item in hypos
end
```

Generation Directions:

Left-to-right

Old hypothesis + Word = New hypothesis

Right-to-Left

Word + Old hypothesis = New hypothesis

Bi-directional

Old hypothesis + Word = New hypothesis 1

Word + Old hypothesis = New hypothesis 2

Tuning

Initial Weights → Evaluate tuning data → Determine Δw → Add Δw to weights → Evaluate tuning data

Weights with higher accuracy

Evaluation of hypothesis: BLEU (BiLingual Evaluation Understudy) is used to compare generated hypotheses to correct answers. It considers the precision of the hypothesis, or the percentage of phrases of a certain length in the hypothetical sentence that also appear in the correct answer.

Machine Output: 我 喜欢 读 这本 书 漫画

Correct answer: 我 喜欢 读 这本 漫画 书

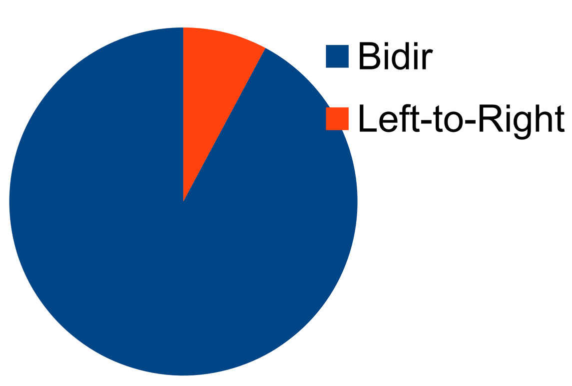
Phrase Length	1	2	3	4
Total in hypothesis	6	5	4	3
In correct answer	6	3	2	1
Precision	1	0.6	0.6	0.3

Tuning methods: efficient means of determining Δw that can result in an improvement in accuracy

- Minimum Error Rate Tuning (MERT): Minimise BLEU difference between best hypothesis and correct answer
- Margin Infused Relaxed Algorithm (MIRA): Maximise score difference between hypothesis closest to correct answer and other hypotheses
- Alternative Tuning: switch between the two algorithms iteration by iteration to benefit from both algorithms

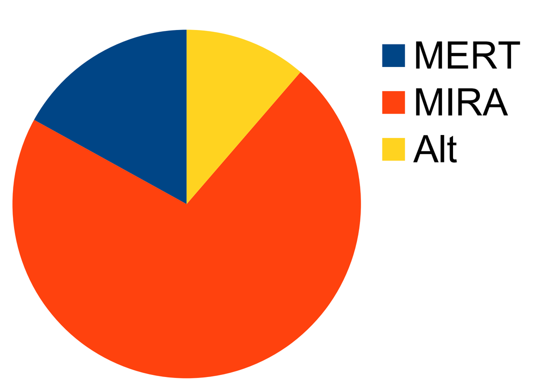
Results

Hypothesis Generation Direction



Right-to-Left does not show very good results, due to the fact that most languages form their sentences from left to right

Tuning methods



Adjusting tuning methods causes improvements as small as 0.005, hence such adjustment is of the least importance

Comparison between models

Evaluating RNN language model is quite time-consuming, but is able to introduce the greatest improvement in accuracy. Hence, RNN should be used when computational requirements can be met.

Most other language models are useful as well, as there is a general trend that the more models we include, the better the results can get.

Some not-so-useful models:

- Backward n-gram language model: due to words are still more related to their precedents
- Models built from lexicons: high irrelevance to primary school context

Conclusion

The project has achieved decent accuracy on texts in our target domain, i.e., primary school Chinese exams. It performs better on short and common sentences because the amount of information lost is greater when we shuffle words in long sentences.

In the future, the project can be extended to larger scopes and even other languages, if enough data is made available to the program. We believe that the findings in this project can be very useful in such extensions.

Reference

Dustin Hillard and Sarah Petersen. N-gram Language Modeling Tutorial. Retrieved from http://ssli.ee.washington.edu/ WS07/notes/ngrams.pdf on Dec 23, 2016

Slava M. Katz. (1987). Estimation of Probabilities from Sparse Data for the Language Model Component of a Speech Recognizer. IEEE Transactions on Acoustics, Speech, and Signal Processing, vol. ASP-35, No. 3, March 1987.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. (2002). BLEU: a Method for Automatic Evaluation of Machine Translation. Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics (ACL), Philadelphia, July 2002, pp. 311-318.

Franz Josef Och. Minimum Error Rate Training in Statistical Machine Translation. Retrieved from http://www.aclweb.org/ anthology/P03-1021 on Dec 23, 2016

Eva Hasler, Barry Haddow, Philipp Koehn. (2011). Margin Infused Relaxed Algorithm (MIRA) for Moses. Retrieved from http://mt-archive.info/MTMarathon-2011-Hasler.pdf on Dec 23, 2016

Sundermeyer, Martin, Ralf Schlüter, and Hermann Ney. LSTM Neural Networks for Language Modeling. Interspeech. 2012.