Distributed Language Models

Thorsten Brants and Peng Xu, Google Inc.

Language models are used in a wide variety of natural language applications, including machine translation, speech recognition, spelling correction, optical character recognition, etc. Recent studies have shown that more data is better data, and bigger language models are better language models: the authors found nearly constant machine translation improvements with each doubling of the training data size even at 2 trillion tokens (resulting in 400 billion n-grams). Training and using such large models is a challenge. This tutorial shows efficient methods for distributed training of large language models based on the MapReduce computing model. We also show efficient ways of using distributed models in which requesting individual n-grams is expensive because they require communication between different machines.

Tutorial Outline

1) Training Distributed Models

* N-gram collection
Use of the MapReduce model; compressing intermediate data; minimizing communication overhead with good sharding functions.

* Smoothing
Challenges of Katz Backoff and Kneser-Ney Smoothing in a distributed system; Smoothing techniques that are easy to compute in a distributed system: Stupid Backoff, Linear Interpolation; minimizing communication by sharding and aggregation.

2) Model Size Reduction

* Pruning
Reducing the size of the model by removing n-grams that don't have much impact. Entropy pruning is simple to compute for Stupid Backoff, requires some effort for Katz and Kneser-Ney in a distributed system. Effects of extreme pruning.

* Quantization
Reducing the memory size of the model by storing approximations of the values. We discuss several quantizers; typically 4 to 8 bits are sufficient to store a floating point value.

* Randomized Data Structures
Reducing the memory size of the model by changing the set of n-grams that is stored. This typically lets us store models in 3 bytes per n-gram, independent of the n-gram

order without significant impact on quality. At the same time it provides very fast access to the n-grams.

3) Using Distributed Models

* Serving
Requesting a single n-gram in a distributed setup is expensive because it requires communication between machines. We show how to use a distributed language model in the first-pass of a decoder by batching up n-gram request.

Target Audience

Target audience are researchers in all areas that focus on or use large n-gram language models.

Presenters

Thorsten Brants received his Ph.D. in 1999 at the Saarland University, Germany, on part-of-speech tagging and parsing. From 2000 to 2003, he worked at the Palo Alto Research Center (PARC) on statistical methods for topic and event detection. Thorsten is now a Research Scientist at Google working on large, distributed language models with focus on applications in machine translation. Other research interests include information retrieval, named entity detection, and speech recognition.

Peng Xu joined Google as a Research Scientist shortly after getting a Ph.D. in April 2005 from the Johns Hopkins University. While his research is focused on statistical machine translation at Google, he is also interested in statistical machine learning, information retrieval, and speech recognition.