A Particle Filter for Bayesian Word Segmentation

Benjamin Börschinger Mark Johnson

Macquarie University

November 30, 2011

Outline

Bayesian Word Segmentation

The Particle Filter learner

Experiments

Conclusion and Outlook

Word Segmentation

- one of the first tasks children have to master is to break speech into smaller units (e.g. words)

$$\mathrm{j} \,_{\triangle}\, \mathrm{u} \,_{\blacktriangle}\, \mathrm{w} \,_{\triangle}\, \text{ɑ} \,_{\triangle}\, \mathrm{n} \,_{\triangle}\, \mathrm{t} \,_{\blacktriangle}\, \mathrm{t} \,_{\triangle}\, \mathrm{u} \,_{\blacktriangle}\, \mathrm{s} \,_{\triangle}\, \mathrm{i} \,_{\blacktriangle}\, \text{ð} \,_{\triangle}\, \text{ə} \,_{\blacktriangle}\, \mathrm{b} \,_{\triangle}\, \text{ʊ} \,_{\triangle}\, \mathrm{k}$$

"you want to see the book"

- learning to segment utterances $\leftrightarrow$ learning a lexicon for the language

Bayesian Word Segmentation

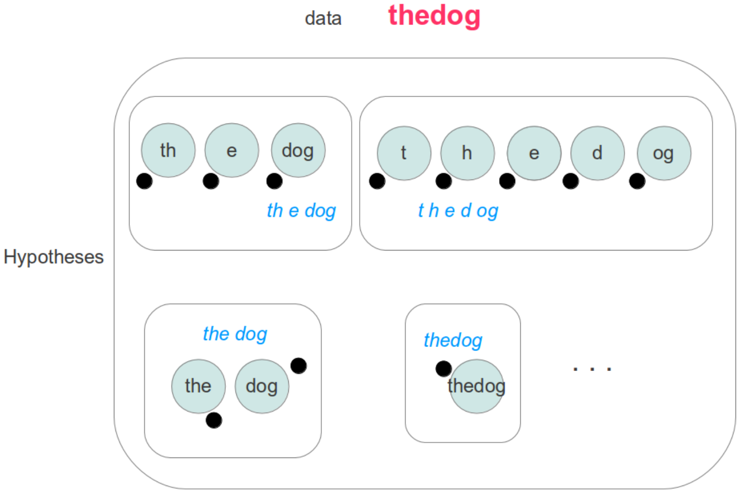
- observed utterances are produced by drawing words from an *unknown lexicon* and concatenating the words
- given unsegmented data, infer the *segmentation* and the *lexicon*
- Bayesian bit: prefer smaller lexicons
- MDL approaches dating back to de Marcken, Brent and others
- State-of-the-art: Adaptor Grammars encoding linguistically motivated knowledge (syllable structure, tones,...)
- here: *non-parametric* model introduced by Goldwater 2007

The Goldwater Model for Word Segmentation

- lexicon is a distribution over words
- data assumed to arise from i.i.d. draws from (unknown) lexicon
- don't know number nor nature of the words in advance
- $\Rightarrow$ lexicon is a draw from a Dirichlet Process Prior
- $\Rightarrow$ the base-distribution is a distribution over all possible words
- $\Rightarrow$ the lexicon assigns probability mass to a subset
- in a Bigram model, there is a special lexicon for each word, and a shared back-off lexicon (hierarchical DP)

Inference

- data is corpus (unsupervised task)
- find posterior distribution over hypotheses, given data
- hypotheses are segmentations $\Leftrightarrow$ lexicons



Inference

- intractable to calculate posterior analytically
- MCMC sampling algorithms produce samples from the posterior
- $\Rightarrow$ Monte Carlo approximation using the samples
- requires multiple iterations over the training data

Why Particle Filters?

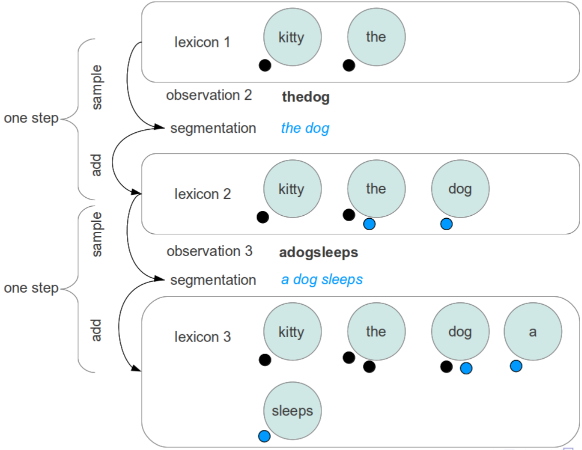
- online (or sequential) learning algorithm
- "make use of observations one at a time, [...] and then discard them before the next observations are used" (Bishop 2006:73)
- practical interest, e.g. large datasets or sequentially arriving data
- scientific interest, e.g. whether algorithm behaves similar to human learners
- this work: starting point for adressing these questions by showing how to build a Particle Filter for models like this

Particle Filters — The Idea

- update the posterior distribution, one observation at a time
- not exactly a new idea for Bayesians
- consider a hypothesis H, and two observations O_1, O_2
- $P(H|O_1) \propto P(O_1|H)P(H)$
- $P(H|O_1, O_2) \propto P(O_2|H)P(H|O_1)$
- "posterior at time t is prior at time $t + 1$"
- approximate each posterior with weighted set of samples or particles (Monte Carlo method, if number of particles goes to infinity, approximation converges on the true posterior)
- to get new posterior, simply update each particle and calculate weights

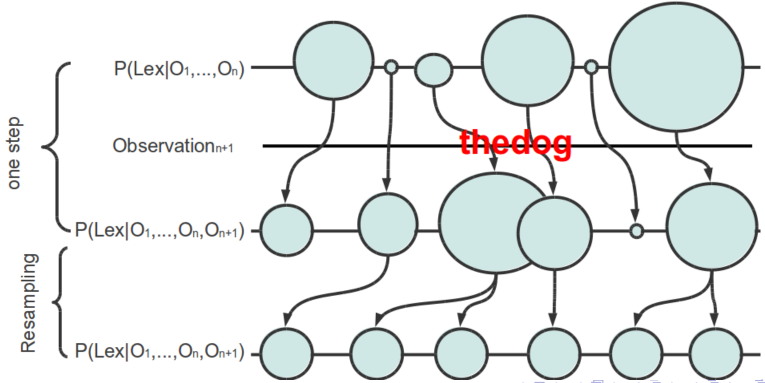
Updating an individual Particle

- each particle is a lexicon (cum grano salis)
- updating a lexicon corresponds to
  - sampling a segmentation given the current lexicon
  - adding the words in this segmentation to the lexicon



Updating a set of Particles

- weighted particles ⇒ finite approximation of posterior over lexicons
- updating weights based on likelihood of the observation
- here: also corrects for use of a proposal distribution during propagation (no efficient sampling method for true distribution)
- one particle tends to take all the mass ⇒ resample (SIS**R** algorithm)



Experiments

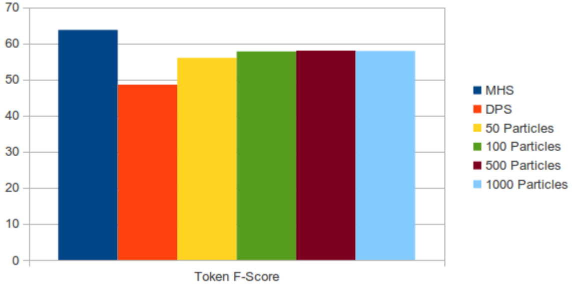
- **unsupervised** segmentation of the Brent (1999) data
  - 9790 phonemically transcribed CDS utterances
- compare to a batch learner, and Pearl et al.'s DPS learner
- two questions of interest
  - recovering true posterior $\Rightarrow$ look at log-probability of training data at end
    - expect to find a high probability solution
  - (doing Word Segmentation $\Rightarrow$ look at segmentation metric)
- it's known to be a hard task...

Pearl et al. (2011)'s algorithms

- an utterance based Metropolis Hastings sampler
  - batch learner, run for 20,000 iterations
- Dynamic Programming Sampling algorithm
  - samples a segmentation, given current lexicon
  - adds the words to the lexicon, considers next utterance
  - $\Rightarrow$ a 1 particle Particle Filter
  - no possibility at all to correct earlier mistakes

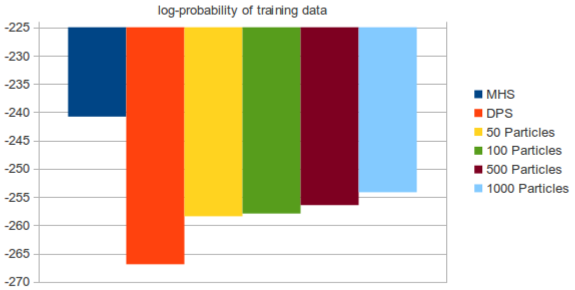
Bigram model — token f-score

- Particle Filters considerably worse than batch learner
- 1 (DPS) vs 50 particles makes big difference
- seems to ceil rather quickly $\Rightarrow$ presumably, even larger numbers of particles required



Bigram model — log probability

- clear trend that more particles lead to higher probability solutions
- again, large improvement in going from 1 to 50



Bigram model — discussion

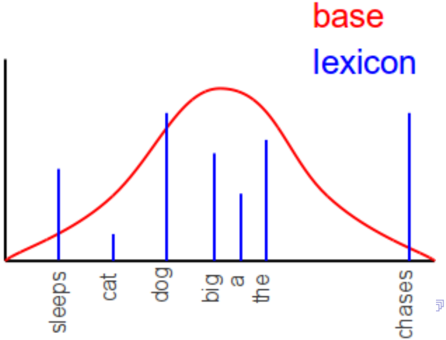
- marked difference between 1 and 50 particles
- trend that larger numbers lead to better performance
- Particle Filter "never looks back", which may explain the need for large numbers
  - correcting earlier mistakes only indirectly by keeping many alternatives
  - number of possible segmentations is exponential
- $\Rightarrow$ possibly relaxing the strict online nature is an alternative to the use of ever larger numbers of particles

Conclusion and Outlook

- presented a Particle Filter algorithm for Bayesian Word Segmentation
- a strict online learner can only get so far (theoretical guarantee, but...)
- starting point for extensions to the basic algorithm
  - already started experimenting with "resampling the past"
  - framework to study learning trajectories
    - can track learners progress in time
  - idea ought to be applicable to other Bayesian Non-Parametric models (e.g. Adaptor Grammars)

The Goldwater Model for Word Segmentation

- lexicon is a distribution over words
- data assumed to arise from i.i.d. draws from (unknown) lexicon
- don't know number nor nature of the words in advance
- $\Rightarrow$ lexicon is a draw from a Dirichlet Process Prior
- $\Rightarrow$ the base-distribution is a distribution over all possible words
- $\Rightarrow$ the lexicon assigns probability mass to a subset

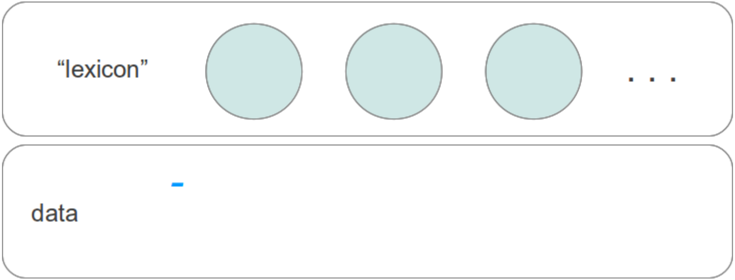


The Goldwater Unigram Model

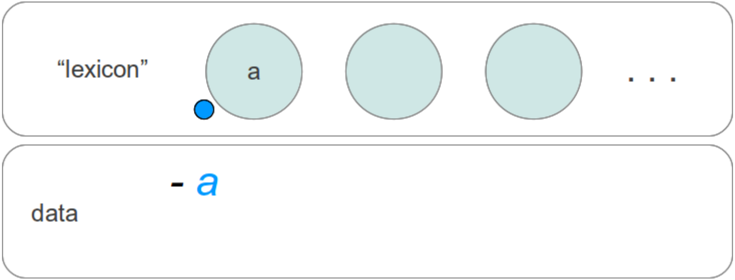
$$
\begin{aligned}
\theta_{phon} &\sim Dirichlet(\alpha_{phon}) \\
P_{phon}(x|\theta_{phon}) &= \theta_{phon,x} \\
P_0(w = x_1 \ldots x_n|\theta_{phon}) &= \left(\prod_{i=1}^{n} P_{phon}(x_i|\theta_{phon})\right) P_{phon}(stop|\theta_{phon}) \\
Lex|\gamma, P_0, \theta_{phon} &\sim DP(\gamma, P_0) \\
W_i|Lex &\sim Lex
\end{aligned}
$$

- prior on θ_{phon} allows us to learn a distribution over phonemes from the lexicon
- in practice, integrate out θ_{phon} and Lex $\Rightarrow$ Chinese Restaurant Process over words
- cum grano salis: utterance boundaries as special word

Chinese Restaurant Process as Generative Process

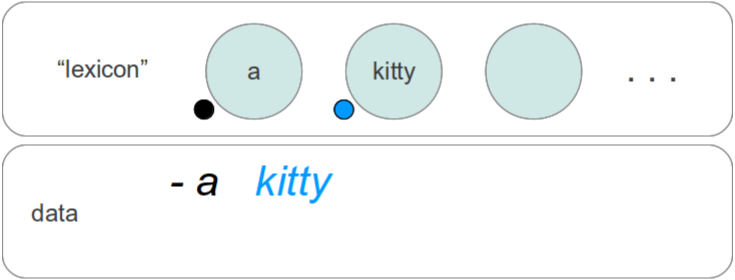


Illustration



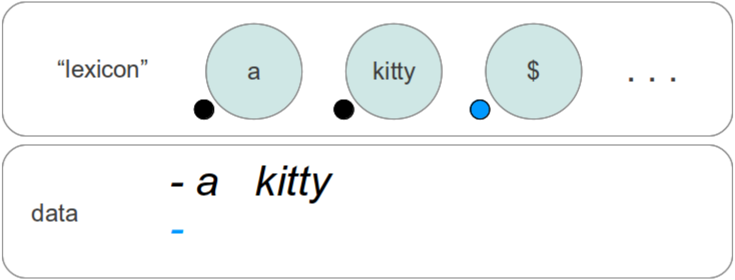
$$P_{data} = P_0(a)$$

Illustration



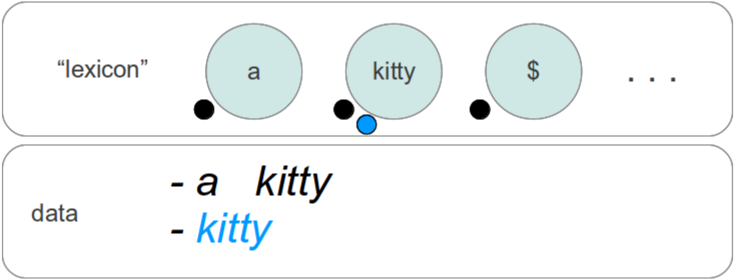
$$P_{data} = P_0(a) \times \frac{\gamma P_0(kitty)}{\gamma+1}$$

Illustration



$$P_{data} = P_0(a) \times \frac{\gamma P_0(kitty)}{\gamma+1} \times \frac{\gamma P_0(\$)}{\gamma+2}$$

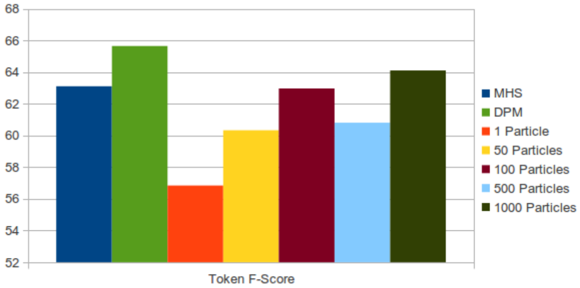
Illustration



$$P_{data} = P_0(a) \times \frac{\gamma P_0(kitty)}{\gamma+1} \times \frac{\gamma P_0(\$)}{\gamma+2} \times \frac{1}{\gamma+3}$$

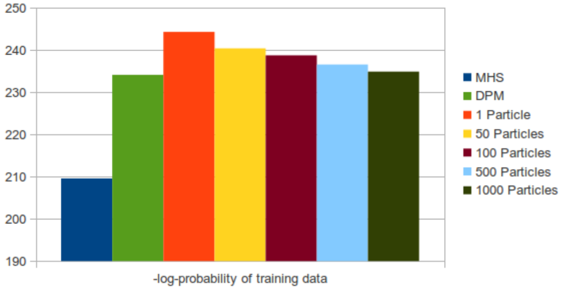
Unigram model — token f-score

- higher is better
- known that lower probability solutions "look" better (next slide)



Unigram model — log probability

- smaller is better
- batch algorithm wins by a large margin
- trend that more particles lead to better log probability



Unigram model — discussion

- Brent heuristic does extremely well for an online learner
- large numbers of particles required $\Rightarrow$ unlikely to scale
- high dimensional state space (number of possible segmentations exponential)
- relaxation of "don't look back" most likely to make Particle Filters useful in practice