JMLR: Workshop and Conference Proceedings 21:254–258, 2012 The 11th ICGI

Simple Variable Length N-grams for Probabilistic Automata Learning

Fabio N. Kepler FABIOKEPLER@UNIPAMPA.EDU.BR
Sergio L. S. Mergen SERGIOMERGEN@UNIPAMPA.EDU.BR
Cleo Z. Billa CLEOBILLA@UNIPAMPA.EDU.BR
LEA Group, Alegrete, Federal University of Pampa – UNIPAMPA

Editors: Jeffrey Heinz, Colin de la Higuera, and Tim Oates

Abstract

This paper describes an approach used in the 2012 Probabilistic Automata Learning Competition. The main goal of the competition was to obtain insights about which techniques and approaches work best for sequence learning based on different kinds of automata generating machines. This paper proposes the usage of n-gram models with variable length. Experiments show that, using the test sets provided by the competition, the variable-length approach works better than fixed 3-grams.

Keywords: Probabilistic automata learning; Variable length n-grams; Markov models.

1. Introduction

This paper describes the approach used in the 2012 Probabilistic Automata Learning Competition – PAutomaC[1]. The competition was about learning non-deterministic probabilistic finite state machines, based on artificial data automatically generated by four kinds of machines: Markov Chains, Deterministic Probabilistic Finite Automata, Hidden Markov Models and Probabilistic Finite Automata. The main goal of the competition was to obtain insights about which techniques and approaches would work best for these machines given their kind and parameters set. For the details about how the machines were generated and an overview of the area of probabilistic automata learning see Verwer et al. (2012).

Basically, a number of different problems, i.e., generated by different machine settings, were provided, each consisting of a training set and a test set. Each problem comprises a number of sentences, which are sequences of integer symbols inside a given vocabulary. The goal is to predict the probability of a sequence of symbols. Three baselines were provided: one based on the frequency of each token; a 3-gram model; and the ALERGIA algorithm.

Our approach to solve these problems is based on an n-gram model with variable length. This kind of solution handles the memory cost of larger n-grams by performing different pruning strategies to effectively shrink the state space. In this paper we show how we employed this idea using a simple tree structure.

This paper is organized as follows: in Section 2 we explain the vln-gram approach we used; in Section 3 we show the results this approach achieved; and in Section 4 we draw some considerations.

1. http://ai.cs.umbc.edu/ecgi2012/challenge/Pautomac

2. N-grams with variable length

The use of a trigram (3-gram) model as a baseline for many tasks is due to its simplicity and yet relatively good performance. Several researchers argued that a higher order model, i.e., an n-gram model with $n > 3$, would increase performance, if a sufficiently large training set was given. However, this incurs in an intractable state space in terms of required memory.

A solution developed since the 1990's to deal with such restrictions is to use n-gram models with variable length (Ron et al., 1996; Kneser, 1996). These models use larger values for n, which means a longer context window is used, and then prune the state space, allowing contexts with different lengths to occur.

In order to effectively store contexts with variable length we use a simple tree structure, which we call a context tree. A context tree is built using the sequences of symbols from a training set. An example can be seen in Figure 1. Each node is a symbol in the vocabulary (in this case $\{1, 2, 3, 4\}$) and has an associated table that shows the symbols that appear after the sequence formed by the nodes of the path up to the root.

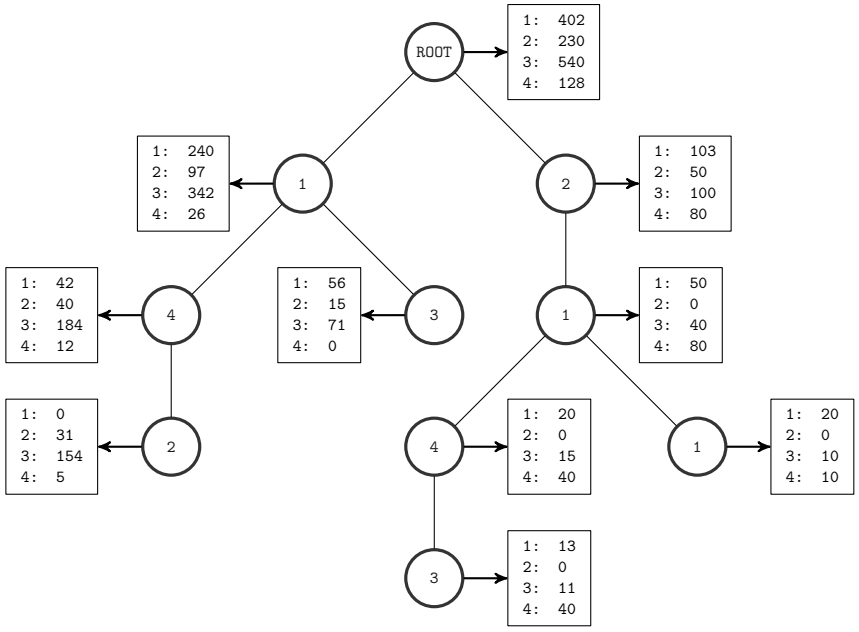

Figure 1: Example of a context tree.

Taking the leaf node with the symbol 2, the context it represents is 2 4 1, and 2 3 4 are symbols that appear after 2 4 1. That means the training set contains the subsequences 2 4 1 2, 2 4 1 3, and 2 4 1 4, which occur 31, 154, and 5 times, respectively.

The process for building the tree involves two steps. First, having defined n as the longest size of subsequences to consider, we run over the training set, adding sequences with n symbols to the tree, either by inserting new nodes or by updating counts.

Then the tree is pruned. For every leaf, the Kullback-Leibler divergence is calculated between the leaf and its parent. If the divergence is greater than a given cut value, defined

empirically, it means the leaf node does not add more relevant information than its parent, and thus is pruned off the tree.

To predict the probability of a sequence of symbols we go the usual way. For each symbol, from left to right, we search the tree for the longest subsequence to its left, returning its corresponding probability and multiplying it by the probabilities of the previous symbols.

3. Results

We use the data made available during the training phase of the competition. This allows us to use the solution also made available and compare the perplexities of different models. For each kind of machine used to generate the data, we choose at least two problems: one where vln-grams perform well and one where it does not. The kinds of machine are HMM (Hidden Markov Models), MC (Markov Chains), PDFA (Probabilistic Deterministic Finite Automata), and PNFA (Probabilistic Non-deterministic Finite Automata). We report the perplexities for the 3-gram baseline, the leader during the training phase[2], and three vln-gram instances with different combinations of parameters.

Table 1 shows the results. $|V|$ is the vocabulary size, and **vln-gram1**, **vln-gram2** and **vln-gram3** use as parameters, respectively, $n = 4$, $K = 5 \times 10^{-4}$; $n = 5$, $K = 5 \times 10^{-4}$; and $n = 5$, $K = 5 \times 10^{-5}$, where n is the maximum subsequence length and K is the cut value used for pruning the tree according to the Kullback-Leibler divergence. We can see that the 3-gram baseline is easily surpassed by a vln-gram model in almost all cases, the exceptions being problems HMM 22 and PNFA 10.

Table 1: Perplexity of various models with respect to the solution.

Dataset	$\|V\|$	**3-gram**	**Leader**	**vln-gram1**	**vln-gram2**	**vln-gram3**
HMM 23	5	45.2930	41.3413	44.7347	44.6885	41.4695
HMM 41	21	3.1160	3.0573	3.0655	3.0623	3.0676
HMM 22	18	11.1563	11.1389	11.5601	11.6288	11.3496
MC 7	11	28.2403	27.4760	27.9483	28.0040	27.7656
MC 8	8	69.8465	68.2840	69.2366	69.1741	68.8033
MC 6	5	68.4676	68.4409	68.4464	68.4471	68.4480
PDFA 19	10	4.6605	4.6032	4.6075	4.6147	4.6071
PDFA 21	7	41.9779	41.2604	42.1521	42.0768	41.4281
PDFA 43	9	41.4766	28.5565	36.2411	32.2015	32.1959
PNFA 17	5	40.0527	39.6367	40.4574	40.4567	39.6922
PNFA 48	20	43.9553	34.7346	36.2312	35.6318	35.3566
PNFA 10	7	34.4641	34.4641	35.0555	35.0376	34.5221

However, we can also see that not always the same vln-gram model yields the best result for all problems. The reasons for that are unclear, although they are obviously due to the

2. Available in http://ai.cs.umbc.edu/icgi2012/challenge/Pautomac/results_old.php.

specificities of each problem's generating automata. For example, a large vocabulary could cause data sparsity with long contexts. But in problems HMM 41 and specially PNFA 48 we see that longer contexts have lower perplexity, despite the large vocabularies.

Regarding the different kinds of problems, we can see that no single kind can be said to be easier than another. Each of them has both harder and easier problems. Some of the problems clearly need just short contexts, and getting longer contexts with a higher cut value does not always yield a similar perplexity. However, as noted above, using n-grams greater than 3 and pruning them yields better results than a fixed 3-gram.

Table 2 shows results over the HMM 23 problem with different parameter values. Neither increasing the context size nor decreasing the cut value shows a predictable result. There seems to be a tipping point around $K = 5 \times 10^{-6}$ after the context is larger than 4.

Table 2: Perplexities using different model parameters over a single problem (HMM 23).

K $(n = 4)$	**Perplexity**	K $(n = 5)$	**Perplexity**	K $(n = 6)$	**Perplexity**
5×10^{-4}	44.7347	5×10^{-4}	44.6885	5×10^{-4}	43.9577
5×10^{-5}	42.2438	5×10^{-5}	41.4695	5×10^{-5}	41.7114
5×10^{-6}	42.1051	5×10^{-6}	41.4659	5×10^{-6}	41.6727
5×10^{-7}	42.1046	5×10^{-7}	41.4685	5×10^{-7}	41.7205

There are also the aspects of the resulting model size and the influence of the training set size, but due to space restrictions we do not analyze them here.

4. Considerations

We showed the simple approach of using variable length n-grams for sequence modeling, which is used by the community since the 1990's for various different problems, and reported its development and results on the automata learning competition. Our goal was to apply the simplest possible approach which would be able to surpass the baselines, taking advantage of not having to deal with unknown words and not having to actually predict sequences, thus not requiring the use of any algorithm like Viterbi's. The approach expands over the classic 3-gram model, being somewhat easy to implement and yielding usually better results. However, it raises the questions about how to determine the best size of the context to be used along with the best cut value. A way of determining these parameters depending on aspects of the training set remains as a future work, which has the potential of raising the baseline bar in the future.

To allow the replication of the experiments, the full source code is available at https://gist.github.com/e

Acknowledgement

The authors would like to thank the competition committee for providing open data and solutions, and encouraging participation and the writing of this paper.

References

Reinhard Kneser. Statistical language modeling using a variable context length. In *Spoken Language, 1996. ICSLP 96. Proceedings., Fourth International Conference on*, volume 1, pages 494–497. IEEE, 1996.

Dana Ron, Yoram Singer, and Naftali Tishby. The power of amnesia: Learning probabilistic automata with variable memory length. *Machine Learning*, 25:117–149, 1996.

Sicco Verwer, Rémi Eyraud, and Colin de la Higuera. PAutomaC: a PFA/HMM Learning Competition. In *Proceedings of the 11th International Conference on Grammatical Inference*, 2012.