Active LeZi: An Incremental Parsing Algorithm for Sequential Prediction

Karthik Gopalratnam and Diane J. Cook
Department of Computer Science and Engineering
The University of Texas at Arlington, Arlington, Texas 76019-0015
{gopalara, cook}@cse.uta.edu

Abstract:

Prediction is an important component in a variety of domains in Artificial Intelligence and Machine Learning, in order that Intelligent Systems may make more informed and reliable decisions. Certain domains require that prediction be performed on sequences of events that can typically be modeled as stochastic processes. This work presents *Active LeZi*, a sequential prediction algorithm that is founded on an Information Theoretic approach, and is based on the acclaimed LZ78 family of data compression algorithms. The efficacy of this algorithm in a typical Smart Environment – the Smart Home, is demonstrated by employing this algorithm to predict device usage in the home. The performance of this algorithm is tested on synthetic data sets that are representative of typical interactions between a Smart Home and the inhabitant.

Introduction

As Intelligent Systems become more common and diversified, an important capability that they need to possess is the ability to predict the occurrence of various events in order to be able to adapt and have the versatility to make decisions in a variety of situations. Especially common is the problem of sequential prediction – given a sequence of events, how do we predict the next event based on a limited known history.

In this work, we present *Active LeZi*, a prediction algorithm that approaches this prediction problem from an Information Theoretic standpoint. For any sequence of events that can be modeled as a stochastic process, this algorithm employs the power of Markov models to optimally predict the next symbol in any stochastic sequence. Many situations demand that the prediction algorithm be capable of incrementally gathering information and deliver real time, "online" predictions. *Active LeZi* is based on the LZ78 data compression algorithm, which employs incremental parsing, thereby addressing this requirement.

Consider a sequence of events being generated by an arbitrary deterministic source, which can be represented by the stochastic process $X = \{x_i\}$. The sequential prediction problem can then be stated as follows. Given the sequence of symbols $\{x_1, x_2, \ldots. x_i\}$, what is the next symbol x_{i+1}? Well-investigated text compression methods have established that good compression algorithms are also good predictors. According to Information Theory, a predictor that builds a model whose entropy approaches that of the source achieves greater predictive accuracy. Also, it has been shown that a predictor with an order that grows at a rate approximating the entropy rate of the source is an optimal predictor. Another motivation to look to the field of text compression is that such algorithms are essentially incremental parsing algorithms, which is an extremely desirable quality in our search for an online predictor. *Active LeZi* is a predictor addressing this prediction problem, and is based on the above mentioned motivations.

Related Work

The problem of constructing a Universal Predictor for sequential prediction of arbitrary deterministic sequences was first considered in (Feder, Merhav and Gutman 1992). where they proved the existence of Universal Predictors that could optimally predict the next of *any* deterministic sequence. They also defined the concept of *predictability*, which is central to the idea of sequential prediction, and proved that Markov predictors based on the LZ78 family of compression algorithms attained optimal predictability.
These concepts were implemented in branch prediction in computer programs (Federovsky, Feder and Weiss 1998), and page pre-fetching into memory (Vitter and Krishnan 1996). Another predictive method based on the LZ78 algorithm was developed by Bhattacharya et al for a predictive framework for mobility tracking in PCS networks (Bhattacharya and Das 2001).

Predictability & Finite State (FS) Predictors

Consider a stochastic sequence $x_1^n = x_1, x_2, \ldots\ldots x_n$. At time *t*, the predictor will have to predict what the next symbol x_t is going to be, based on the past history – the sequence of input symbols $x_1^{t-1} = x_1, x_2, \ldots\ldots x_{t-1}$. Let the predicted symbol be b_t. There is a loss function $l(b_t, x_t)$ associated with every such prediction, and the object of any predictor is to minimize the fraction of prediction errors, π associated with the predictor - i.e., the quantity:

$$\pi = \frac{1}{n}\sum_{t=1}^{n} l(b_t, x_t)$$

has to be minimized. (Feder, Merhav and Gutman 1994)

Given the resources in a practical situation, the predictor that is capable of possibly meeting these requirements has to be a member of the set of all possible finite state machines (FSM's). Consider the set of all possible finite state predictors with *S* states. Then the *S-state predictability* of the sequence x^n (denoted by $\pi_S\left(x^n\right)$), is defined as the minimum fraction of prediction errors made by an FS predictor with *S*-states. This is a measure of the performance of the best possible predictor with *S* states, with reference to a given sequence. For a fixed length sequence, as *S* is increased, the best possible predictor for that sequence will eventually make zero errors. The *finite state predictability* for a particular sequence is then defined as the *S* – state predictability for very large *S*, and very large *n*, i.e. the finite state predictability of a particular sequence is

$$\lim_{S\to\infty}\lim_{n\to\infty}\pi_S\left(x^n\right).$$

FS predictability is an indicator of the best possible sequential prediction that can be made on an arbitrarily long sequence of input symbols by any FSM. This quantity is analogous to FS compressibility, as defined in (Ziv and Lempel 1978), where a value of zero for the FS predictability indicates perfect predictability and a value of ½ indicates perfect unpredictability.

This notion of predictability enables a different optimal FS predictor for every individual sequence, but it has been shown in (Feder, et al. 1992) that there exist *universal* FS predictors that, independent of the particular sequence being considered, always attain the FS predictability.

Prediction Using Incremental Parsing – The LZ78 Algorithm

An important result that is derived in (Feder, Merhav and Gutman 1992) is that a sub-class of the class of FS predictors, the class of Markov Predictors, performs asymptotically as well as *any* FSM. That is, a sequential Markov predictor whose order varies at the appropriate rate with respect to the number of symbols seen in the sequence will eventually attain FS predictability.

The LZ78 data compression algorithm as suggested by Lempel and Ziv (Ziv and Lempel 1978), is an incremental parsing algorithm that introduces exactly such a method for gradually changing the Markov order at the appropriate rate. This algorithm has been interpreted as a universal modeling scheme that sequentially calculates empirical probabilities in each context of the data, with the added advantage that the generated probabilities reflect contexts seen from the beginning of the parsed sequence to the current symbol. The LZ code length of any individual sequence attains the Markovian empirical entropy for any finite Markov order, i.e., the LZ algorithm in effect attains the Markov entropy of any given source, thereby functioning as a Universal Predictor.

LZ78 is a dictionary-based text compression algorithm that performs incremental parsing of an input sequence. This algorithm parses an input string "x_1, x_2, …. x_i" into *c(i)* substrings "w_1, w_2, …. $w_{c(i)}$" such that for all j>0, the prefix of the substring w_j (i.e., all but the last character of w_j) is equal to some w_i for *1<i<j.* Because of this prefix property, parsed substrings can efficiently be maintained in a trie. Since LZ78 is a compression algorithm, it traditionally consists of two parts: the encoder and the decoder. Both encoder and decoder maintain dictionaries of phrases seen so far to encode/decode. In our case, however, we do not need to reconstruct the parsed sequence and therefore do not need to consider this as an encoder/decoder system, but simply a system that breaks up a given sequence (string) of states into phrases. From this perspective, Figure 1 shows the pseudo code representation of LZ78 parsing.

```
initialize dictionary := null
initialize phrase w := null
loop
    wait for next symbol v
    if ((w.v) in dictionary):
        w := w.v
    else
        add (w.v) to dictionary
        w := null
        increment frequency for every
            possible prefix of phrase
    endif
forever
```

Figure 1: The LZ78 algorithm

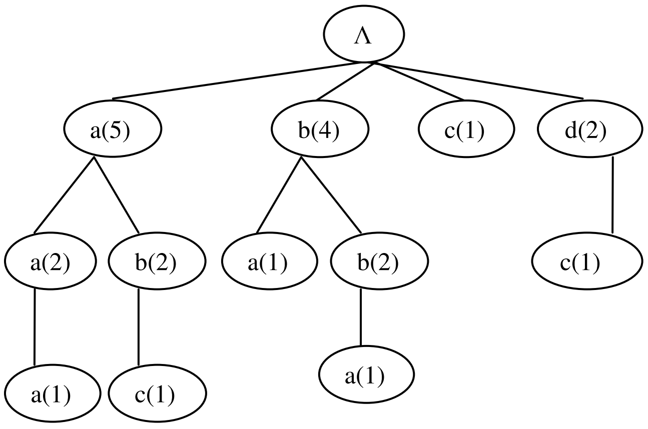

Figure 2: Trie formed by LZ78 parsing

Consider the sequence of input symbols x^n = "*aaababbbbbaabccddcbaaaa*". An LZ78 parsing of this string of input symbols would yield the following set of phrases: "*a,aa,b,ab,bb,bba,abc,c,d,dc,ba,aaa*". This algorithm, as described above maintains statistics for all contexts seen within the phrases w_i. For example, the context '*a*' occurs 5 times (at the beginning of the phrases "*a, aa, ab, abc, aaa*"), the context "*bb*" is seen 2 times ("*bb,bba*"), etc. These context statistics are stored in a trie. (Figure 2).

Active LeZi

As shown in the previous section, a prediction scheme based on the LZ78 parsing algorithm can be viewed as a Universal Markov encoder with time-varying order, and therefore attains the desired FS predictability. In this section we describe the *Active LeZi* algorithm, which is constructed based on LZ78 compression.

Problems in LZ78 parsing

Any practical implementation of LZ78 suffers from the following two drawbacks:

a) In any LZ parsing of an input string, all the information crossing phrase boundaries is lost. This is a major drawback for device usage prediction – there might be significant patterns crossing phrase boundaries that affect the next likely event.
b) The convergence rate of LZ78 to the optimal predictability as defined above is slow. The results outlined in (Feder, Merhav, Gutman 1991) state that LZ78 *asymptotically* approaches optimal predictability. The authors have pointed out in a later commentary on the work (Feder, Merhav and Gutman 1994b), that any practical implementation will have to address this issue.

Bhattacharya, et al. have addressed the issue of slow convergence rate to some extent in the *LeZi Update* algorithm (Bhattacharya and Das 2002) , by keeping track of all possible contexts within a given phrase, and not just the prefixes to be found within a phrase. This method, however, does not address the issue of information lost across phrase boundaries.

Active LeZi (ALZ)

Active LeZi is an enhancement of LZ78 and *LeZi Update* (Bhattacharya and Das 2002) that incorporates a sliding window approach to address the drawbacks outlined earlier. This approach also demonstrates various other desirable characteristics.

As the number of states seen in an input sequence grows, we can see that the amount of information being lost across the phrase boundaries increases rapidly. Our solution to this problem involves maintaining a variable length window of previously-seen symbols. We choose the length of the window at each stage to be equal to the length of the longest phrase seen in a classical LZ78 parsing. The reason for selecting this window size is that the LZ78 algorithm is essentially constructing an (approximation to an) order-*k-1* Markov model, where *k* is equal to the length of the longest LZ78 phrase seen so far. (See Figures 2 and Figure 4).

Within this window, we can now gather statistics on all possible contexts. This builds a better approximation to the order-k Markov model, because it has captured information about contexts in the input sequence that cross phrase boundaries in the classical LZ78 parsing. Therefore, we gain a better convergence rate to optimal predictability as well as greater predictive accuracy.

Figure 3 illustrates the algorithm itself.

```
initialize dictionary := null
initialize phrase w := null
initialize window := null
initialize Max_LZ_length = 0
loop
   wait for next symbol v
   if ((w.v) in dictionary):
      w := w.v
   else
      add (w.v) to dictionary
      update Max_LZ_length if necessary
      w := null
   endif

   add v to window
   if (length(window) > Max_LZ_length)
      delete window[0]
   endif
   Update frequencies of all possible
         contexts within window that
      includes v
forever
```

Figure 3: *Active LeZi* Algorithm

Figure 4 shows the trie formed by the Active LeZi parsing of the input sequence "*aaababbbbbaabccddcbaaaa*".

We can see that this is a "more complete" order-*Max_LZ_length-1* Markov model than the one shown in Figure 2, in that it now incorporates more information about the contexts seen. (In our input sequence, this is an order-2 Markov model).

Active LeZi demonstrates the following characteristics:

a) It is in essence a growing order Markov model that attains optimal FS predictability, due to the optimality of LZ78.
b) As the length of the longest LZ78 phrase grows, it stores more and more information, which implies that as the input sequence, i.e., the experience grows, the algorithm performs better. This is a desirable characteristic of any learning algorithm.
c) The convergence to the optimal FS predictor is faster since ALZ now gathers information that was formerly inaccessible to the LZ78 parser.

Probability Assignments for Prediction

In order to predict the next event of the sequence that ALZ has built a model of, we calculate the probability of each state occurring in the sequence, and predict the one with the highest probability as the most likely next action.

It has been pointed out in (Feder, Merhav, Gutman 1994b) that in order to achieve better convergence rates to optimal predictability, the predictor must "lock on" to the minimum possible set of states that is representative of the sequence being considered.

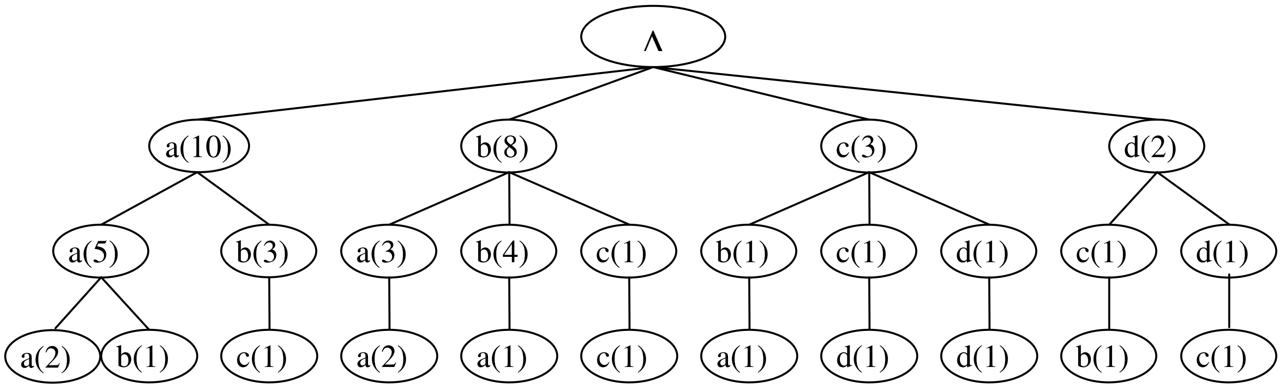

Figure 4: Trie formed by the ALZ parsing of the string "*aaababbbbbaabccddcbaaaa*

For sequential prediction, it has been shown that this is possible by using a "mixture" of all possible order models in assigning the next sequence its probability estimate. For a method that considers different orders of models, we turn once again to data compression and the Prediction by Partial Match (PPM) family of predictors. This has been used to great effect in (Bhattacharya and Das 2002), for a predictive framework based on LZ78.

PPM algorithms consider different order Markov models in order to build a probability distribution by weighting different order models appropriately. In our predictive scenario, Active LeZi builds an order-*k* Markov model. We now employ the PPM strategy of *exclusion* (Bell, Cleary and Witten 1990) to gather information from all the orders 1 through *k* models in assigning the next symbol its probability value. This method is illustrated by considering the example sequence used in the previous sections - "*aaababbbbbaabccddcbaaaa*".

The window maintained by Active LeZi is the set of *contexts* used to compute the probability of the next symbol. In our example, the last phrase "*aaa*" (which is also the ALZ window) is used. Within this phrase, the contexts that can be used are suffixes within the phrase, except itself (i.e. "*aa*", "*a*", and the null context).

Suppose the probability that the next symbol is an *a* is being computed. From Fig 4 we see that an *a* occurs two out of the five times that the context "*aa*" appears, the other cases producing two null outcomes and one "*b*". Therefore the probability of encountering an "*a*" at the context "*aa*" is 2/5, and we now fall back (escape) to the order-1 context (i.e. the next lower order model) with probability 2/5. At the order-1 context, we see an "*a*" five out of the ten times that we see the "*a*" context, and of the remaining cases, we see two null outcomes. Therefore we predict the "*a*" at the order-1 context with probability 5/10, and escape to the order-0 model with probability 2/10. At the order 0 model, we see the "*a*" ten out of 23 symbols seen so far, and we therefore predict "*a*" with probability 10/23 at the null context. The *blended* probability of seeing an "*a*" as the next symbol is therefore

$$\frac{2}{5}+\frac{2}{5}\left\{\frac{5}{10}+\frac{2}{10}\left(\frac{10}{23}\right)\right\}$$

Similarly, let us compute the probability that the next symbol is a "*c*". In this case, the order-2 and the order-1 contexts do not yield a "*c*". Therefore, we escape to the order-0 model and predict a "*c*" with a probability of 3/23. In this case the total probability of seeing a "c" would be

$$\frac{0}{5}+\frac{2}{5}\left\{\frac{0}{10}+\frac{2}{10}\left(\frac{3}{23}\right)\right\}=\frac{2}{5}*\frac{2}{10}*\frac{3}{23}.$$

This method of assigning probabilities has the following advantages:

a) It solves the zero-frequency problem. In the above example, if only the longest context had been chosen to make a decision on probability, it would have returned a zero probability for the symbol "*c*", whereas lower-order models show that this probability is indeed non-zero.
b) This blending strategy assigns greater weight to higher-order models in calculating probability if the symbol being considered is found in that context, while lower-order models are suppressed owing to the null context escape probability. This is in keeping with the advisability of making the *most informed* decision.

Application & Results

The Smart Home provides a ready environment for employing sequential prediction algorithms such as ALZ. As an Intelligent Agent, the goals of the Smart Home include maximizing inhabitant comfort and optimizing energy usage (Das et al. 2001), by reducing interaction between the inhabitant and the Home. To achieve this end, one of the tasks that the Home has to perform is predict which of the devices in the home the inhabitant will interact with next, so that that activity may be automated. The home will have to make this prediction based only on previously seen inhabitant interaction with various devices. From the comfort standpoint as well as for optimizing energy consumption, it is essential that the number of prediction errors that the house makes is kept to a minimum – not only would it be annoying for the inhabitant to have to reverse home decisions, but prediction errors will lead to energy wastage.

A smart home inhabitant typically interacts with various devices as part of his routine activities, interactions that may be considered as a sequence of events with some inherent pattern of recurrence. For example, our routines when we wake up in the morning are most likely the same everyday – turn on the kitchen light, turn on the coffee maker, turn on the bathroom light, turn off the bathroom light, turn on the toaster, turn off the coffee maker, etc.

Typically, each inhabitant-home interaction event '*e*', is characterized as a triple consisting of the device with which the user interacted, the change that occurred in that device, and the time of interaction. In this model we assume that devices are associated with the binary-valued ON and OFF states.

$$e = \langle Device\#, ON/OFF, TIME \rangle$$

The following tests were designed to evaluate the sequential prediction capability of ALZ, so the time information of the events was not considered. Each unique input symbol, x_t is therefore identified by the two-tuple consisting of the device ID, and the state change of that device.

ALZ has been tested on data obtained from a Synthetic Data Generator (SDG), which approximates the data that would likely be obtained in a real Home scenario. The SDG can be used to generate data from various configurable scenarios of user interaction.

The first set of tests was performed on data sets with a great deal of inherent repetitiveness, and lacking noise. The ALZ Learning Curves were plotted by testing the number of correct predictions from the next 100 events, while increasing the training data set. This generated the learning curve in Figure 5, and as can be seen the performance converges to 100% accuracy rapidly. This proves that ALZ is a strong sequential predictor.

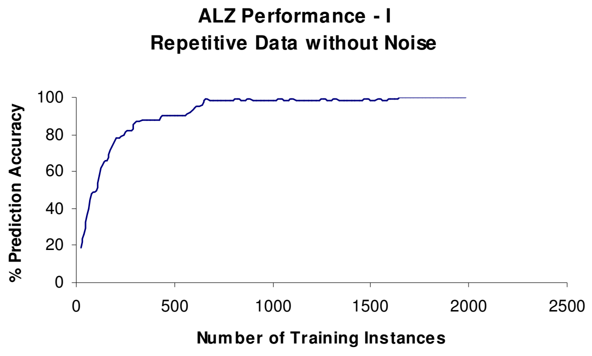

Figure 5: ALZ Learning Curve – I

The second series of tests used SDG data sets of 2000 points generated from a set of 6 typical home scenarios, which incorporated significant noise. This yielded the learning curve shown in Figure 6, which converges to about 86% accuracy.

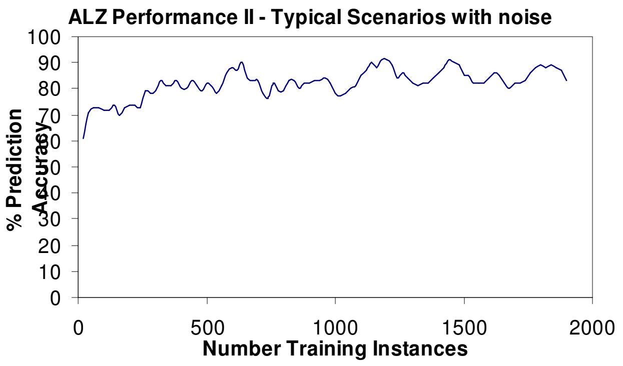

Figure 6: ALZ Learning Curve - II

Conclusions

We can see that the method adopted by ALZ is indeed effective in a practical prediction scenario, in that it reaches high prediction accuracy with a relatively small training data set. This is because of the capability of ALZ to build an accurate model of the source generating events, a feature inherited from its Information Theoretic background and the LZ78 text compression algorithm. The sound theoretical principles on which ALZ is founded also mean that ALZ is an optimal Universal Predictor, and can be used in a variety of prediction scenarios.

References

1. Feder M., Merhav N. and Gutman M. 1992, Universal Prediction of Individual Sequences, *IEEE Transactions on Information Theory,* Vol 38, No.4.

2. Ziv J. and Lempel A. 1978, Compression of Individual Sequences via Variable Rate Coding, *IEEE Trans. Inform Theory*, vol. IT-24, pp. 530-536.

3. Bhattacharya A. and Das S. K. 2002, LeZi-Update: An Information-theoretic framework for personal mobility tracking in PCS networks, *ACM/Kluwer Wireless Networks Journal*, vol. 8, no. 2-3, pp. 121-135.

4. Das S., Cook D., Bhattacharya A., Heierman E., and Lin T. 2001, The Role of Prediction Algorithms in the MavHome Smart Home Architecture, *IEEE Personal Computer Systems*.

5. Bell T.C, Cleary J.G., Witten I.H. 1990, Text Compression, *Prentice Hall Advanced Reference Series*.

6. Cover T. and Thomas J. 1991, Elements of Information Theory, *Wiley* 1991.

7. Vitter J.S. and Krishnan P. 1996, Optimal Prefetching via data compression, *Journal of the ACM* 43(5), 771-793.

8. Feder M., Merhav N. and Gutman M. 1994, Reflections on 'Universal Prediction of Individual Sequences' , Invited article, *IEEE Information Theory Society Newsletter*.

9. Federovsky E., Feder M. and Weiss S. 1998, Branch Prediction based on Universal Data Compression Algorithms, *Proc. Ann. Symposium on Computer Architecture*, pp. 62-72.