NSC: A New Progressive Sampling Algorithm

Alfonso Estrada[1] and Eduardo F. Morales[2]

[1] Computer Science Department
ITESM - Campus Ciudad de México
alestrad@exatec.itesm.mx
[2] Computer Science Department
ITESM - Campus Cuernavaca,
Temixco, Morelos, 62589, México
eduardo.morales@itesm.mx

Abstract. As the size of the databases increases, machine learning algorithms face more demanding requirements for efficiency and accuracy. Rather than analyzing all the data, an alternative approach is to use only a small subset trying to achieve competitive results with less resources. Progressive sampling follows this approach, starting with a small sample of data and increasing the size progressively until the accuracy (or other performance measure) cannot be improved. Progressive sampling aims for equivalent accuracies obtained with all the data but with significantly less resources. This paper introduces a new progressive sampling algorithm called NSC which: (i) defines an initial data set with a simple to evaluate measure and close to the sample size where the accuracy no longer improves, (ii) proposes a sampling schedule which is more aggressive than arithmetic sampling but more conservative than the standard geometric sampling with a *geometric factor* of 2, and (iii) introduces a termination criterion based on an adaptive local linear regression algorithm and on the complexity of the machine learning algorithm used to generate models. It is shown experimentally that NSC consistently outperforms both arithmetic and geometric sampling on several databases.

1 Introduction

The last decades have seen an accelerated increase in the size of the databases and it is expected for this trend to continue. It is relatively common to find nowadays databases of several gigabytes and there are starting to emerge databases with several terabytes (see for instance [1] for a survey of techniques to deal with very large data bases). Machine learning algorithms have been used to try to extract useful knowledge from databases. As the size of the databases increases, machine learning faces more demanding requirements for efficiency and accuracy. Accurate models require in general more data to uncover useful and possibly complex patterns. However, efficiency requires less data since most inductive algorithms grow at least linearly with the size of the training set. With the increasing size of the databases, it is becoming impractical to consider all the data.

Progressive sampling has been relatively recently proposed as an alternative approach to achieve competitive results while considering only a small subset of data. The general idea is to start with a small sample of data and increase the size progressively until the accuracy (or other performance measure) cannot be longer improved.

Figure 1 depicts a prototypical learning curve with increasing training sets. In general, these learning curves have a steep slope early in the curve, follow by a more gentle slope in the middle and a plateau in the last section. The size of the training set where the plateau starts (marked with a vertical line in Figure 1), which we will refer to as N_{min}, represents the size of the smallest training set after which it is not possible to improve the accuracy considering additional data. Models built with smaller training sets than N_{min} are less accurate, while models with larger sets have the same accuracy.

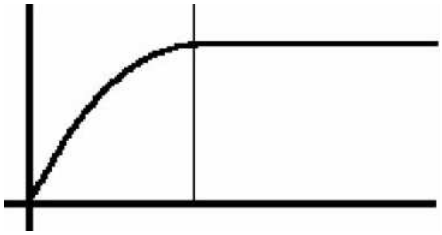

Fig. 1. A prototype learning curve with increasing training set size.

The idea of progressive sampling is to detect convergence in the learning curve (N_{min}) after a few samples to efficiently produce models with high accuracy. Progressive sampling requires the definition of at least three main components: (i) the sampling schedule, (ii) the initial data sample, and (iii) the termination criterion.

The two most common approaches for progressive sampling, arithmetic and geometric sampling, have focused mainly on the definition of the sampling schedule $S = \{n_0, n_1, \ldots, n_k\}$, where the n_is represents the size of the sample used by the induction algorithm.

Arithmetic sampling has the following schedule: $S_{arith} = n_0 + (i \cdot n_\delta)$, where n_0 is a starting data set and n_δ is a fixed increment in the size of the data to consider [2]. For instance: $\{100, 200, 300, 400, \ldots\}$. Its obvious drawback is that databases with a large N_{min} require a large number of samples, that can eventually produce longer training times than analyzing the whole data set in the first place.

Geometric sampling was proposed as an improved sampling mechanism. It uses the following schedule: $S_{geom} = a^i \cdot n_0 = \{n_0, a \cdot n_0, a^2 \cdot n_0, a^3 \cdot n_0, \ldots\}$ where a and n_0 are constants [3]. For instance: $\{100, 200, 400, 800, \ldots\}$. Although a, which we will call it the *geometric factor*, can be any constant, by default it has been taken to be equal to 2. Its main drawback is that it is too aggressive

and can easily surpass by a large amount of data N_{min}, and consequently build models with more data of what it is really needed.

In this paper a new progressive sampling algorithm, called NSC, is described. NSC uses geometric sampling but with a less aggressive *geometric factor* of 1.1 instead of 2. This makes NSC not as conservative as arithmetic sampling, nor as aggressive as geometric sampling. NSC proposes a new and simple to evaluate initial training sample size that is, in general, close to N_{min}. Finally, in NSC two termination criteria are introduced, one is based on the detection of convergence with an adaptive mechanism of points used for a local linear regression algorithm, and the other one is based on the complexity of the induction algorithm used to generate the models.

Section 2 describes NSC in more detail. In section 3 experimental results on several databases are presented. Finally, section 4 provides conclusions and future research directions.

2 Progressive Sampling with NSC

NSC follows a general progressive sampling algorithm as described in Table 1. It first defines a sampling schedule based on a small geometric factor and on the complexity of the inductive algorithm (see Table 2). NSC continues taking samples according to the sampling schedule until it detects convergence (described in Section 2.3) or when there are no more elements in the sampling schedule.

Table 1. The NSC algorithm.

Let:
$N \leftarrow$ data size
$M \leftarrow$ number of attributes
$CML \leftarrow$ time complexity of the machine learning algorithm
with N and M
$S = \{n_0, n_1, \ldots, n_k\} = NSC\_schedule(N, CML)$
$i \leftarrow 0$
while not *NSC_detect_convergence* or $i \leq k$
$n_i \leftarrow$ element i from S
$M \leftarrow$ model induced with n_i instances
$i \leftarrow i + 1$
end while
return M

To generate the sampling schedule, NSC defines: (i) an initial data set, (ii) a new geometric factor for the sampling schedule, and (iii) the size of the sampling schedule. These will be described in the following sections.

2.1 Selection of N_{min}

Although previous progressive sampling algorithms have paid little attention to a proper definition of the initial training set, this is relevant for the efficiency of the algorithm. The closer the initial size to N_{min} the more efficient the progressive sampling algorithm will be.

Trying to determine an estimate for N_{min} is not an easy task, as it depends on the particular database and on the learning algorithm. There have been some previous approaches with little success. For instance, in [2] arithmetic sampling is compared against, what they called a *static* sampling. Static sampling computes $\hat{N}_{min}$ based on a subsample's statistical similarity with the entire set using Kullback information measure. They show that arithmetic sampling produces more accurate models than static sampling. The main problem is that once $\hat{N}_{min}$ is determined no further sampling is performed. Other approaches have tried to estimate N_{min} from all the data (e.g., [4]), this is however, computationally too expensive and not as effective as expected, having to use a progressive sampling approach after a proposed $\hat{N}_{min}$. NSC also continues with a progressive sampling approach after proposing an initial sample size, however its estimate only depends on the size of the database. It is a very easy to evaluate estimate that has given good results on the databases where it has been tested. The idea is to guide the size of the sample to consider from a database with N instances, by the size of the sample require to estimate, with certain confidence level, the mean of a population of size N. The size of this sample depends on a confidence level. From statistics (e.g., [5]), the size of a sample to estimate the mean of a population with a certain confidence level, is given by:

$$N_{min} = \frac{N * \sigma^2}{(N-1) * D + \sigma^2}$$

where D is $\frac{(N*C)^2}{4}$ and C is the confidence level of this estimate. We need to estimate σ^2, which in the absence of any information it is approximated in NSC by $\sigma \approx \frac{N}{4}$. Since $N - 1 \approx N$ for large N, and after some simple manipulation:

$$N_{min} = \frac{N}{4 \cdot N \cdot C^2 + 1}$$

One option is to fix a confidence level (C) and apply it to all the databases. One problem with this approach is that a good confidence level, and consequently the estimate of N_{min} will depend, in general, on the particular database under study. Trying to diminish this, the N_{min}'s of 11 diverse databases, mostly taken from the UCI repository of machine learning databases [7][3], were obtained using a conservative and expensive arithmetic progressive sampling approach. These values were used to adjust a curve for the confidence level that could produce good approximations for the N_{min}'s on these databases (see Figure 2). An exponential function fitted best these values with the following form:

[3] The UCI databases were: Autohorse, Autoprice, Breast, Iris, Kropt, SE, Sonar, Soybean, Spambase and Ticdata_categ.

$C = 41.468 * N^{-0.479} + 0.05$. It basically decreases exponentially the size of C, and consequently of N_{min} with respecto to the size of the database.

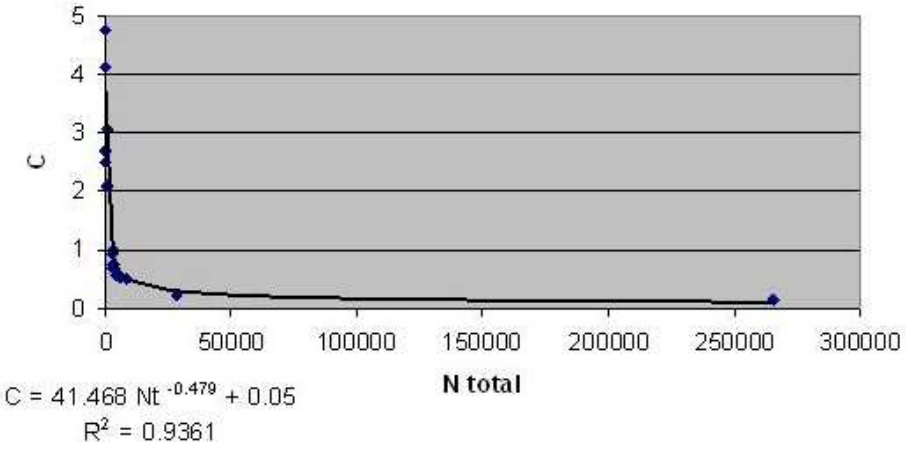

Fig. 2. Confidence value curve for good estimates of N_{min} with several databases.

To summarize, we used an estimate of the size of a sample of a population of size N to estimate its mean as a good estimate for N_{min} and adjust the confidence level considering several databases. Although this is a very rough approximation it has produced very good results and it is very easy to evaluate.

2.2 The sampling schedule

The sampling schedule follows a geometric sampling approach with a geometric factor of 1.1. That is: $S_{NSC} = 1.1^i \cdot n_0 = \{n_0, 1.1 \cdot n_0, 1.1^2 \cdot n_0, 1.1^3 \cdot n_0, \ldots\}$ (see Table 2). This is more aggressive than arithmetic sampling but more conservative than a geometric sampling with a geometric factor of 2. The geometric factor was obtained after performing several tests on 11 databases. Although the geometric factor in NSC has been fixed to 1.1, as future work we will like to adjust this number dynamically. One of the main ideas behind this research is that a combination of a good estimate for N_{min} with a geometric factor which is greater than 1 but much less than 2 can produce a more effective and efficient progressive sampling approach.

With this sampling schedule, we can estimate the number of samples where NSC is as expensive as generating a model with all the data. This of course depends on the complexity of the learning algorithm and determines the size of the sampling set. NSC stops considering bigger training sets when its accumulated computational cost is equivalent to the computational cost of generating a model with all the data. We can obtain the size of the sampling set as follows:

Table 2. The NSC scheduling algorithm.

NSC_schedule(N, CML)
  Estimate N_{min} considering N
  Evaluate $MaxIt$ (maximum number of samples) considering
    CML and N
  Let $geo\_factor = 1.1$
  $i \leftarrow 0$
  $n_i \leftarrow N_{min}$
  while $i < MaxIt$
    $i \leftarrow i + 1$
    $n_i \leftarrow n_{i-1} * geo\_factor$

$$\begin{aligned} CML(N) &= CML(N_{min}) + 1.1 * CML(N_{min}) + 1.1^2 * CML(N_{min}) + \ldots \\ &= CML(N_{min})(1.1^0 + 1.1^1 + 1.1^2 + \ldots) \\ &= CML(N_{min}) \sum_{i=0}^{x} 1.1^i \\ &= CML(N_{min}) * \frac{1.1^{x+1} - 1}{0.1} \end{aligned}$$

where $CML(N)$ and $CML(N_{min})$ are the computational costs of the machine learning algorithm used to generate a model with all the data and with N_{min} respectively. Then the size of the sample set is:

$$x = \frac{log(0.1 * \frac{CML(N)}{CML(N_{min})} + 1)}{log(1.1)} - 1$$

Depending on the complexity of the machine learning algorithm used to generate the models, we can take the floor of the above expression and limit the total number of samples to this value.

For instance, for an algorithm whose complexity is linear with the number of instances:

$$x = \lfloor \frac{log(0.1 * \frac{N}{N_{min}} + 1)}{log(1.1)} - 1 \rfloor$$

For an algorithm like C4.5, its complexity is $O(A*Nlog(N))+O(N(log(N))^2)$ [6], where A is the number of attributes. In practice, it has been shown to be roughly between $O(N^{1.2})$ and $O(N^{1.4})$. Considering $O(N^{1.3})$, the number of samples is:

$$x = \lfloor \frac{log(0.1 * (\frac{N}{N_{min}})^{1.3} + 1)}{log(1.1)} - 1 \rfloor$$

Similar measures can be obtained with for different progressive sampling schedules with other geometric factors and other machine learning algorithms.

2.3 Detecting convergence

The final part that needs to be defined is how can convergence be detected effectively and efficiently. This is still an open problem for progressive sampling. Statistical estimation may require a complex functional form to estimate accuracy. The curve shown in Figure 1 has three regions of behavior which are difficult to capture with simple functional forms. Provost et al. [3] suggest using linear regression with local sampling (LRLS). In their proposal they take the last sample size n_i and sample l additional points in the local neighborhood of n_i. Although they report effective convergence detection (with $l = 10$), it is far from been efficient due to the additional sampling and executions of the machine learning algorithm.

In NSC the last points used in the sampling schedule are fed to a local regression algorithm. The number of points vary with respect to the behavior of the accuracy curve. Starting with two points, the number of points remains constant while the slope is increasing steeply (> 0.025), it increases one point when it shows a more gently positive slope ($0.01 <$ slope ≤ 0.025) in the middle region, and it increases two points with a negative slope (< 0.0), trying to cover oscillating behaviors. The different threshold values were obtained experimentally after several tests on 11 diverse databases. One important aspect of this strategy is that it captures the shape of the learning curve followed by machine learning algorithms and dynamically adjustes the number of points to consider in the local linear regression algorithm.

3 Experiments

We compared NSC against several schedules: S_N, a single sample with all the data, $S_0 = \{N_{min}\}$, the optimal sample size determined by experimentation, arithmetic sampling (S_{arith}), and geometric sampling (S_{geom}) with a *geometric factor* of 2. The experiments are reported on 8 databases and averaged over ten runs.

First, we evaluated our estimated $\hat{N}_{min}$ against the real N_{min} obtained after a long experimentation process. Table 3 shows these results along with the size of the databases (N) and their number of attributes ($Atts$). These databases were taken from UCI machine learning repository [7], except for the first and the last one. The first database is from the Mexican dependency in charge of the main toll roads in Mexico (CAPUFE), and covers road accidents from 1995 ([8]). The last database are records from an insurance company. It has 5,100,611 instances, but we only took a little more than 250 thousand in our tests due to limitations in the computational resources. This is a clear example where progressive sampling is needed.

Table 3. Comparison of our estimated $\hat{N}_{min}$ and N_{min}.

DataBase	N	Atts.	$\hat{N}_{min}$	Value of C	N_{min}	Percentage
CAPUFE	3,209	28	1,800	0.917	1,543	86
CH	3,196	37	2,400	0.919	1,536	64
HY	3,163	26	1,900	0.923	1,522	80
Kropt	28,956	7	21,700	0.357	11,544	53
MU	8,124	28	5,500	0.606	3,703	67
SE	3,163	26	2,500	0.923	1,522	61
Ticdata_Categ	5,822	86	4,500	0.702	2,710	60
Transactional	265,691	8	91,000	0.155	74,773	82

As it can be seen from Table 3, $\hat{N}_{min}$ is, at least on these databases, a reasonable estimate of N_{min}. Our estimate is, however, only a rough approximation of N_{min} and can clearly overestimate in some cases. It can be easily shown that if we duplicate several times the instances in the databases, our $\hat{N}_{min}$ will be much higher than N_{min}. Nevertheless it has given very promising results on the databases where it has been tested.

NSC was evaluated in terms of accuracy and execution time. Both arithmetic and geometric sampling started with 100 initial samples as suggested in [3]. The same local regression algorithm introduced by NSC was used to detect convergence for the arithmetic, geometric and NSC sampling. This avoided sampling 10 new points in the case of geometric sampling which could artificially increase its execution times. In terms of the size of the sampling schedule, arithmetic sampling was stopped if it did not reach the plateau within 60% of the data (after which the accumulated computational costs were in general much higher than generating a model with all the data), geometric sampling was stopped if it did not reach the plateau after analyzing 50% of the data (since the next sample considers all the data), and NSC was stopped with the scheme described in the last section which depends on the machine learning algorithm used. The results are averaged over 10 runs and shown in Tables 4 and 5.

From 4 it can be seen that the accumulated times used by S_{NSC} are competitive with S_{geom} and consistently smaller than S_{arith}. Although geometric sampling performs much larger leaps than NSC, NSC starts with an initial training set which is much closer to N_{min}.

From Table 5 it can be seen that NSC has consistently higher accuracy than arithmetic and geometric sampling.

4 Conclusions and future work

This work introduced a new progressive sampling algorithm called NSC. It was shown experimentally to be superior to arithmetic and geometric sampling in terms of accuracy and competitive to geometric sampling in terms of execution

Table 4. Execution times for $N, N_{min}, S_{arith}, S_{geom}, S_{NSC}$.

DataBase	N	N_{min}	S_{arith}	S_{geom}	S_{NSC}
CAPUFE	22.1	11.07	25.05	27.36	**12.01**
CH	10.08	6.04	13.15	7.59	**5.78**
HY	12.07	5.33	13.6	**4.24**	4.85
Kropt	31.05	21.92	85.07	55.17	**31.85**
MU	6.52	4.56	11.63	7.87	**1.51**
SE	23.91	18.05	25.97	**8.64**	9.19
Ticdata_Categ	45.89	34.98	93.98	**33.26**	40.23
Transactional	542.25	90.27	1,751.74	**47.78**	83.16

Table 5. Accuracy for $N, N_{min}, S_{arith}, S_{geom}, S_{NSC}$.

DataBase	N	N_{min}	S_{arith}	S_{geom}	S_{NSC}
CAPUFE	0.97	0.97	0.96	0.95	**0.97**
CH	1.00	1.00	**0.99**	0.98	**0.99**
HY	0.99	0.99	0.95	**0.99**	**0.99**
Kropt	0.57	0.53	0.28	0.38	**0.47**
MU	1.00	1.00	**1.00**	**1.00**	**1.00**
SE	0.98	0.97	0.96	**0.97**	**0.97**
Ticdata_Categ	0.90	0.90	0.83	0.88	**0.90**
Transactional	0.68	0.66	0.54	0.61	**0.66**

time. In NSC: (i) an initial training set is introduced, with a simple to evaluate measure that is reasonably close to N_{min}, (ii) the sampling schedule is more aggressive than arithmetic sampling but more conservative than geometric sampling with a geometric factor of 1.1, (iii) a new convergence criterion which considers the three regions of the accuracy curve and which changes dynamically the number of points to consider in a local regression algorithm is introduced, and (iv) the size of the sampling schedule is limited by the accumulated execution time which is determined by the inductive algorithm used to generate models.

There are several research areas for future work. We would like to perform more experiments on different and much larger databases. We believe that our estimate for N_{min} could be improved if we consider additional information besides the size of the database. Finally, the geometric factor in NSC is fixed to 1.1. As future work, we would like to adjust it dynamically depending on the behavior of the learning curve.

References

1. Provost, F., Kolluri, V.: A survey of methods for scaling up inductive algoithms. Data Mining and Knowledge Discovery **2** (1999) 131–169
2. John, G., Langley, P.: Static versus dynamic sampling for data mining. In: Proceedings of the Second International Conference on Knowledge Discovery and Data Mining, AAAI Press (1996) 367–370
3. Provost, F.J., Jensen, D., Oates, T.: Efficient progressive sampling. In: Knowledge Discovery and Data Mining. (1999) 23–32
4. Gu, B., Liu, B., Hu, F., Liu, H.: Efficiently determining the starting sample size for progressive sampling. In de Raedt, L., Flach, P., eds.: Proceedings of the 12th European Conference on Machine Learning (ECML-01), Springer-Verlag (2001) 192–202
5. Scheaffer, R., Mendenhall, W., Ott, L.: Elementary Survey Sampling. PWS Publishers (1986)
6. Witten, I., Frank, E.: Data Mining: practical machine learning tools and techniques with Java implementations. Morgan Kaufmann, San Francisco, CA (2000)
7. Blake, C., Merz, C.: UCI repository of machine learning databases. (1998)
8. Morales, E., Heredia, D., Rodríguez, A.: Mining road accidents. In: Mexican International Conference on Artificial Intelligence (MICAI-2002), LNAI 2313, Springer-Verlag (2002) 516–525