Good–Turing estimation for the Frequentist n-tuple Classifier

Michał Morciniec and Richard Rohwer
Dept. of Computer Science and Applied Mathematics
Aston University
Birmingham, UK B4 7ET

Abstract

We present results concerning the application of the Good–Turing (GT) estimation method to the frequentist n-tuple system. We show that the Good-Turing method can, to a certain extent rectify the Zero Frequency Problem by providing, within a formal framework, improved estimates of small tallies. We also show that it leads to better tuple system performance than Maximum Likelihood estimation (MLE). However, preliminary experimental results suggest that replacing zero tallies with an arbitrary constant close to zero before MLE yields better performance than that of GT system.

1 Introduction

The frequentist n-tuple system can be obtained from the original, binary system by setting the tally truncation threshold θ to ∞ instead of the more usual 1. This allows one to use full tallies to estimate low-order conditional feature densities and apply a Bayesian framework to the classification problem [6].

Given $p(c|\boldsymbol{\alpha})$, the probability of class c conditioned on feature vector $\boldsymbol{\alpha}$ (the set of all memory locations addressed by an unknown pattern), optimal classification results can be obtained by assigning the unknown pattern to the most probable class. Because estimates of conditional feature densities arise naturally in the n-tuple system, Bayes' rule is applied to obtain class probabilities. The likelihood and evidence for the full feature vector are impossible to compute directly, but these can be estimated from low order densities using independence assumptions. The most common approach [11] assumes that $p(\alpha_i|c)$ as well as $p(\alpha_i)$ are independent[1], where α_i is the address of the pattern in n-tuple i. The conditional class density can then be approximated by

$$p(c|\boldsymbol{\alpha}) \approx p(c) \prod_i \frac{p(\alpha_i|c)}{p(\alpha_i)}. \tag{1}$$

However implausible this assumption may appear, there have been reports of reasonable results obtained with this method [3, 2]. The major advantage of frequentist systems is that they do not suffer from saturation. This makes them superior for small n-tuple sizes n, but the advantage tends

[1] It often goes unnoticed that it turns out to be highly restrictive to demand both of these conditions together, a difficulty we presume to be dwarfed by the inaccuracy of each assumption individually.

to disappear as n is increased, due to worsening probability estimates based on diminishing tallies in each of the increasingly numerous memory locations [12]. It would be desirable to modify the frequentist system in such a way as to retain its robustness for any tuple size n.

2 Weakness of the Maximum Likelihood Estimate (MLE)

The maximum likelihood estimate (MLE) has been routinely [3] [11] [13] applied for the frequentist n-tuple system. In this approach, estimate $\hat{p}$ of the true probability p of an event is approximated as the ratio of the event's tally r to the sample size N; $\hat{p} = \frac{r}{N}$. Under the assumption that each tally value is binomially distributed (with unknown probability p that the feature is present in a pattern of class c and $1-p$ that it is not) the ratio $\frac{r}{N}$ is the maximum likelihood estimate of p. The uncertainty of the tally can be defined as its standard deviation, which can be estimated as

$$\delta r = \sqrt{Np(1-p)} \approx \sqrt{N\hat{p}(1-p)} = \sqrt{r(1-p)}. \tag{2}$$

In an n-tuple with n inputs, p is one of $2^n - 1$ other multinomial parameters which sum to 1. Therefore, p is typically much less than 1, so

$$\delta r \approx \sqrt{r}. \tag{3}$$

Equation 3 shows that the accuracy of MLE is limited for the events with small tallies. The relative tally uncertainty $\frac{\delta r}{r}$, grows with diminishing tallies and becomes undefined for zero tally.

It should be noted that the fact that a tally $r = 0$ for some event doesn't imply that the probability of the event is also zero. It merely states that the event has not taken place in a finite sample of size N. This problem is known in the literature [14] as the "Zero Frequency Problem". Various unprincipled, *ad hoc* techniques exist which try to rectify it. The most common one is to add an arbitrary small constant to each zero tally. However, the choice of a particular constant is difficult to justify formally. We make some experimental observations concerning this Maximum Likelihood system with zero tally correction (MLZ) in section 5.

3 Good-Turing Estimate (GTE)

An alternative method of density estimation has been originally proposed by Turing and researched in detail by Good [4] in the context of species frequencies in a mixed population. It has also been applied in linguistics for n-gram probability estimation [5] and statistical text compression [14]. The advantage of GTE over MLE is improvement of the accuracy of the probability estimates derived from non-zero tallies. Moreover, an estimate for objects not present in the sample can also be provided.

Suppose we draw a random sample of size N from the population of objects. We record n_r, the number of distinct objects that were represented exactly r times in the sample, so that

$$N = \sum_{r=1}^{\infty} r n_r. \tag{4}$$

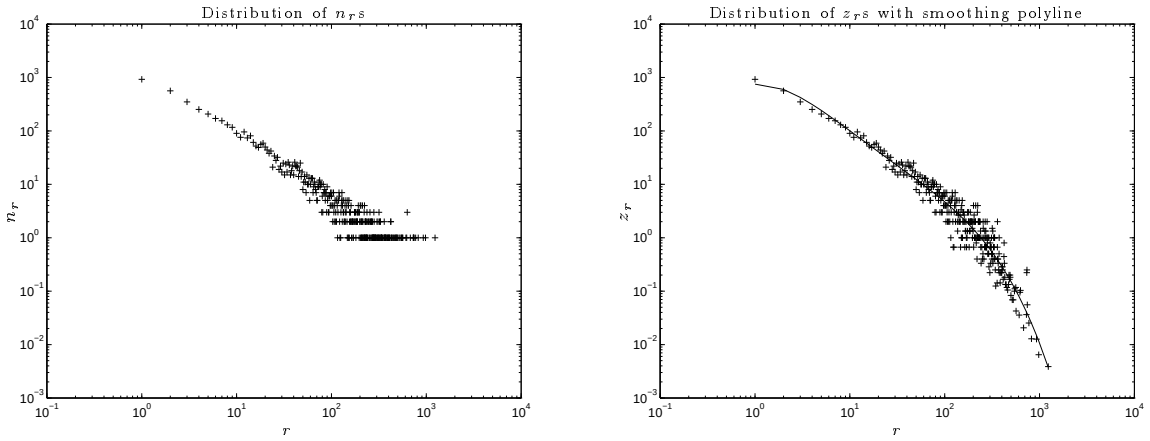

Figure 1: A) Original distribution of frequencies of tallies in class 0 of "tsetse" database. B) distribution after averaging transform has been applied. The solid line denotes the polynomial curve fitted.

Let $\hat{p}_r^{GT}$ denote the Good–Turing estimate of the population probability of an arbitrary object that occured r times in the sample. This entails the assumption that all events which occured r times have the same probability p_r. The Good–Turing theorem states that the expected value of p_r for an event with tally r in one particular sample is r^{GT}/N where the smoothed tally r^{GT} can be approximated as

$$r^{GT} \approx (r+1)\frac{n_{r+1}}{n_r} \qquad r \geq 0. \tag{5}$$

Various derivations [7] of this theorem exist. The values of n_r are most accurate for small values of r and become increasingly noisy for larger tallies. In this respect GTE complements MLE which, as mentioned earlier, becomes less precise for smaller tallies.

4 Smoothing GTEs

The major problem with the Good–Turing theorem is that the distribution $\{n_0, n_1, n_2, \ldots\}$ tends to be sparse and requires smoothing. Moreover, for large values of r there are "gaps" in the distribution of n_r. This suggests that we should average a non-zero n_r value with the zero n_r values surrounding it. We use the transform proposed by Church and Gale [5]

$$z_r = \frac{2n_r}{t-q} \tag{6}$$

where t, r, q are the successive indices of non-zero n_r. Averaging occurs for larger values of r only, because if there are no "gaps" the transformation has no effect.

After averaging we still have to smooth the z_r. This is accomplished by fitting a log polynomial onto the data. Unlike Church and Gale who used polynomial of order one (a straight line) we found that polynomials of higher orders are required to obtain a satisfactory fit to the data. Consequently, we smoothed tally frequency distributions z_r with polynomials of order 4, giving a new smoothed tally r^{SGT},

$$r^{SGT} = (r+1)e^{\sum_{i=1}^{n} a_i \ln^i \frac{r+1}{r}} \qquad r \geq 1 \tag{7}$$

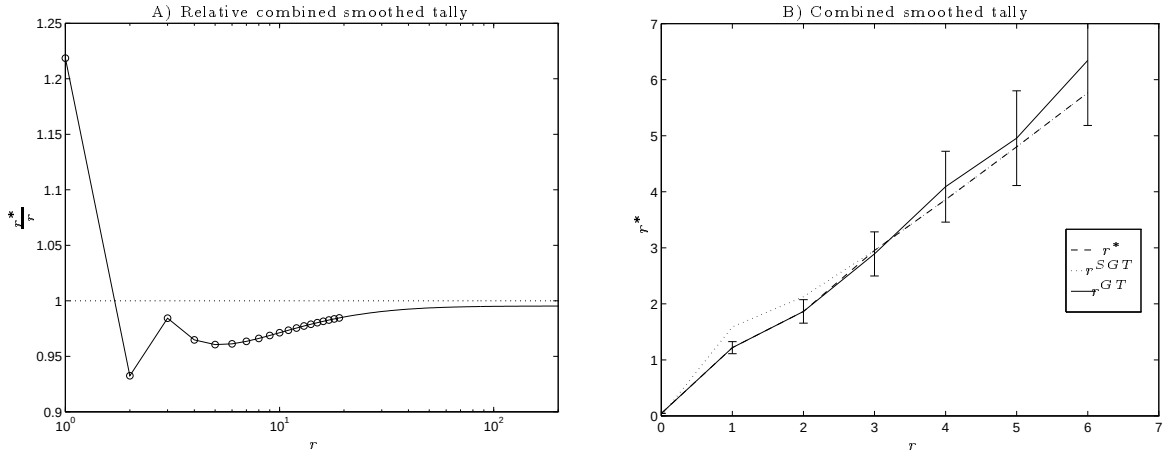

Figure 2: A) Relative adjusted tally r^* for class 0 of the "tsetse" dataset. B) Illustration of the smoothed tally r^* combined from tallies r^{GT} and r^{SGT}. The error-bars on r^{GT} are $1.65 \times \sigma(r^{GT})$. The switch from GTE to SGTE takes place at $r = 3$.

with parameters $a_1, a_2, \ldots, a_4$ determined from the data. Figure 1 shows the original n_r, and averaged z_r distributions with the fitted polynomial curve.

The smoothed Good–Turing estimate (SGTE) may be quite different from the original Good–Turing estimate (GTE) for small values of r. We would therefore prefer to use GTE for small r and then switch to SGTE and keep on using this estimate for the remaining tally values. The new, composite smoothed tally r^* is equal to r^{GT} if $|r^{SGT} - r^{GT}| > 1.65 \times \sigma(r^{GT})$. When the difference becomes insignificant we use SGTE for the remaining tallies. Gale gives the approximation of the variance of r^{GT} as

$$\sigma^2(r^{GT}) \approx (r+1)^2 \frac{n_{r+1}}{n_r^2}\left(1 + \frac{n_{r+1}}{n_r}\right) \tag{8}$$

The probability estimates computed using the corrected tallies have to be normalised because two different methods (GTE and SGTE) of estimation are employed. We compute the probability $\hat{p}_r^{norm}$ for the tally r using unnormalised probabilities $\hat{p}_r^* = \frac{r^*}{N}$ as

$$\begin{cases} n_0 \neq 0 \quad \hat{p}_r^{norm} = (1 - \frac{n_1}{N}) \frac{\hat{p}_r^*}{\sum\limits_{r' \geq 1} n_{r'} \hat{p}_{r'}^*} \qquad r \geq 1, \qquad \hat{p}_0^{norm} = \frac{n_1}{n_0 N} \\ n_0 = 0 \quad \hat{p}_r^{norm} = \frac{\hat{p}_r^*}{\sum\limits_{r'} n_{r'} \hat{p}_{r'}^*} \qquad r \geq 1 \end{cases} \tag{9}$$

5 Application of GTE for the Frequentist n-tuple System

We used several real–world datasets which have been used in the European Community StatLog project [10]. The attributes of the data are in most cases real-valued and pre-processing techniques have been applied [1, 9, 8], providing binary input for the classifier.

In order to obtain probabilities $p(\alpha_i|c)$ normalised within a tuple node one would have to apply Good–Turing estimation for each tuple in each class c separately. This is hardly possible because the distribution n_r is very sparse, especially for small tuple sizes n. Therefore, the estimation has

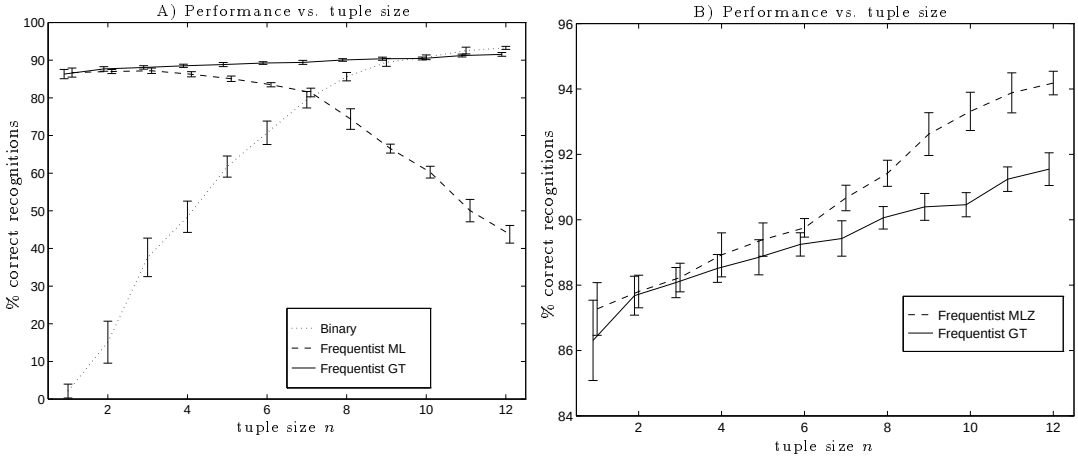

Figure 3: A) Performance of binary and frequentist systems with 100 tuples on the "tsetse.tst". The test set comprises of 1499 patterns. Systems were trained using 3500 training samples. The error-bars are of size one standard deviation computed for 10 random tuple mappings. B) Performance of frequentist Good–Turing system compared with MLZ system using zero tally correction $\epsilon = 10^{-150}$.

been carried out collectively for all T tuples within a class c, i.e., for the population of $T2^n$ features. Consequently, the probabilities $p(\alpha_i|c)$ are normalised within each discriminator c and each zero tally is smoothed by the same amount regardless of the tuple which generated it.

Figure 2A shows the relative composite smoothed tally r^*/r computed for the first discriminator of the n-tuple system trained on the "tsetse" dataset [10]. The construction of the combined smoothed tally r^* is given on figure 2B. We observe that for the first three tallies GTE was chosen whereas SGT was used for the remaining tallies. The adjusted zero tally was $r_0^* = 0.0452$.

We measured the performance of the binary, frequentist ML and GT n-tuple systems against each other. The benchmarking studies were carried out for several STATLOG databases [10]. Figure 3A shows a representative plot for a run on the "tsetse" database. Both frequentist systems perform better than the binary version for small values of n, because they do not suffer from the saturation effect. Unlike the frequentist system with MLE, the GT version retains the performance with increasing n. However, it eventually becomes inferior to binary system. It seems that for n large enough any technique other than zero tally counting (which is equivalent to setting the tally truncation threshold θ to one) is less effective.

We also compared the performance of the GT system to that of MLZ which is technically an ML system with zero tallies substituted by arbitrarily chosen constant ϵ. Preliminary experimental results plotted on Figure 3B suggest that if ϵ is small enough then MLZ will outperform GT system, especially for large n. This can be explained by observing that MLZ with $\epsilon \to 0$ will make exactly the same classification decision as the binary system, except for the patterns that are tied (have the same score) in the binary version. For large n, the saturation is very low, as is the probability of a tie. Consequently, the performance of MLZ must be equal to the performance of a binary system within a margin $\pm\frac{D_{tied}}{D}$ where D is test set size and D^{tied} number of tied patterns.

6 Conclusions

We have demonstrated that a major weakness of the frequentist tuple system using MLE is inadequate probability estimation for small tallies. A principled approach to tally smoothing using Good-Turing formula leads to an improved system performance for larger values of n. However, experiments suggest that replacing zero tallies with a small constant and using a maximum likelihood estimate yields even better results.

7 Acknowledgements

The authors are grateful to Trevor Booth of the Australian CSIRO Division of Forestry for permission to report results on the Tsetse data set, and William Gale of AT&T Bell Laboratories for private communication.

References

[1] N.M. Allinson and A. Kołcz. Application of the cmac input encoding scheme in the n-tuple approximation network. *IEE Proceedings on Comput. Digit. Tech.*, 141(3):177–183, 1994.

[2] Ardeshir Badr. N-tuple classifier for ecg signals. In N. Allinson, editor, *Proceedings of the Weightless Neural Network Workshop '93, Computing with Logical Neurons*, pages 29 – 32, University of York, 1993.

[3] W.W. Bledsoe and C.L. Bisson. Improved memory matrices for the n-tuple recognition method. *IRE Joint Computer Conference*, 11:414–415, 1962.

[4] I.J. Good. The population frequencies of species and the estimation of population parameters. *Biometrika*, 40(3):237–263, 1953.

[5] W.A. Gale K.W. Church. A comparison of the enhanced Good-Turing and deleted estimation methods for estimating probabilities of english bigrams. *Computer Speech and Language*, 5:55–64, 1991.

[6] C.N. Liu. A programmed algorithm for designing multifont character recognition logics. *IEEE Transactions on Electronic Computers*, 139(2):586–593, 1964.

[7] A. Nadás. On Turing's formula for word probabilities. *IEEE Transactions on Acoustics, Speech and Signal Processing*, 33(6):1414–1416, 1985.

[8] R. Rohwer and M. Morciniec. Benchmarking the n-tuple classifier with Statlog datasets. This volume.

[9] R. Rohwer and M. Morciniec. The theoretical and experimental status of the n-tuple classifier. Technical Report NCGR/4347, Aston University Neural Computing Research Group, Aston Triangle Brimingham B4 7ET UK, 1995.

[10] D. Michie D.J. Spiegelhalter and C.C. Taylor, editors. *Machine Learning, Neural and Statistical Classification.* Prentice–Hall, 1994.

[11] M.J. Sixsmith G.D. Tattersall and J.M. Rollett. Speech recognition using n-tuple techniques. *British Telecom Technology Journal*, 8(2):50–60, 1990.

[12] J.R. Ullmann. Experiments with the n-tuple method of pattern recognition. *IEEE Transactions on Computers*, 18(12):1135–1137, 1969.

[13] J.R. Ullmann and P.A. Kidd. Recognition experiments with typed numerals from envelopes in the mail. *Pattern Recognition*, 1:273–289, 1969.

[14] I.H. Witten and T.C. Bell. The zero–frequency problem: Estimating the probabilities of novel events in adaptive text compression. *IEEE Transactions on Information Theory*, 37(4):1085–1094, 1991.