An Accuracy Optimization of a Dialog ASR System Utilizing Evolutional Strategies

Juraj Kacur
Department of Telecommunication, Slovak Technical University, Bratislava, Slovakia
E-mail: kacur@ktl.elf.stuba.sk

Jan Korosi
Department of Telecommunication, Slovak Technical University, Bratislava, Slovakia

Abstract

This article deals with an accuracy optimization of a dialog ASR system based on ATK tool, whereas the optimization is performed by the means of evolutional strategies.

First an introductory overview of the basic capabilities of ATK system is given followed by the founding principles and variations of the evolutional strategies applied to the nonlinear optimization problems. Then the process of ASR optimization is presented and supported by series of experiments.

By applying stochastic optimization methods to the mutual adjustment of Viterbi decoder, speech detection, and grammar related settings, we were able to achieve improved overall performance (WER 18.72%) compared to the manual settings chosen by tests and gained "experience" (generally WER above 23%).

1. Introduction

A successful construction of dialog systems is a very appealing issue in the presence and with the growing knowledge, technological advances and consumer demands it will became even more important. However, their commercial and wide spread usage for the general public is stile somehow awkward because of the lack of "intelligence" and reliability and that there are still limited in their vocabulary and supported grammars.

Nowadays a great effort is spent in reducing the abovementioned drawbacks. The reliability can be improved both in the data preparation and training phases. The speech feature extraction plays an important role in the noise suppression, channel equalization and in making different speech units to be more distinct in the transformed domain. The training process however, aims to get more precise and more robust HMM models from the limited training sets e.g. [2].

On the other hand the proper setting of the recognition process is also very important for the overall performance. In practical realisations this involves: speech detection which controls starting and stopping of the recognition, speech decoding process (variations of the Viterbi decoding algorithm), grammar related issues, reduction of computational and memory loads, etc. Usually these actions are controlled by sets of variables or predefined constants, whose number is often more than several dozens [1]. Unfortunately these are related to each other in nonlinear way either positively or negatively correlated to the overall recognition error. Almost always these are set according to the "common sense" and verified by few experiments, but their optimality is rarely achieved.

Thus in the following, issues related to the speech decoding and its optimization for a dialog application based on the wide spread ATK system will be discussed, however similar findings and concepts are applicable to other systems like SPHIX.

2. A dialog system based on the ATK package

The ATK system [1] is a real time application layer residing above the well known HTK package, thus its main refinement is in the speech decoding process and related auxiliary functions.

The ATK defines 3 system components that perform complex but well separated tasks: the audio component (ASource) is used to input the incoming speech, the speech coding component (ACode) process available data, and the recognition component (ARec) encompasses the Viterbi algorithm with the backtracking. These components are run asynchronously in separated threads and communicate to each other by packages sent to adjacent buffers. Nevertheless these components utilize the HTK functions organized in modules [2]. The exact function of each is controlled by a set of variables listed in the configuration file. The ATK supports 4 modes of operation of an ASR and defines 5 states. The states are: WAIT- system waits for a start signal otherwise is idle, PRIME- an initialisation is performed (grammars, dictionaries and models are loaded and are getting ready for the use), FLUSH- flushing the incoming packets and waiting for the speech segments or the start signal, RUN- the recognition process is running, ANS- backtracking of

the Viterbi algorithm and providing results to the output buffer. The modes of operation can be adjusted as follows: CYCLE- determines if there is only a single recognition pass or the process of recognition is continuous, FLUSH- the system may start the recognition immediately after the initialization phase is completed or it waits for the incoming speech signal or it may even wait for the start signal, STOP- determines whether the recognition stops immediately, or waits for the stop signal, or searches for the detected pause, RESULT- supports following options: provides full result in the ANS state, outputs the best guess in the RUN state, return the unambiguously recognized parts in the RUN state, or output as much information as possible.

A classical ASR system enabling controlled dialog like in [7], which we aim for, can be usually run in these modes: CYCLE- single recognition, i.e. after each recognition (dialog item) a new grammar is initialized, FLUSH- discards all samples till the speech is detected, STOP- stops when silence is recognized, RESULT- returns the unambiguous part of the recognized sentence.

Other settings that are also vital for the decoding process, but are independent of the training phase need to be adjusted ahead of the ASR employment. These belong to 3 ATK components: ARec, Acode and ASource that control following ATK and HTK modules: HParam, ARec, HRec and HNet. A list of features that are most important for the optimization process will be discussed in the paragraph 4 together with their physical meaning.

3. Evolutional strategies for the stochastic non-linear optimization

As it was noted so far there are many features to be optimally set for the real recognition process. Generally these are non-linearly tied into an analytically unknown recognition error function which is to be minimized. All features have some physical boundaries with strong dependencies upon each other and can be either real numbers or integers. Thus one has to solve a multivariable, constrained and non-linear optimization problem. In mathematics there are many effective methods which most of them require an analytical knowledge of the objective function. They are based on the gradient of the error function or its estimation, which in our case (unknown function, for which some dimensions are defined only on integer points), are either useless or rather ineffective. Less sophisticated methods but more suitable for the discussed optimization problem can be hill climbing [3] or direct search algorithms. However these are sensitive to the initial conditions (may stop at the local minima) and are not guaranteed to search through the whole space. Following this course of deliberation we decided to apply the evolutional strategies (ES) – a branch of evolutional algorithms (EA), which was originally designed for such kind of optimization problems [4].

Evolutional strategies belong to a group of stochastic nonlinear optimization methods which are based on the natural process of evolution [3]. They can be applied to variety of optimization problems that are not well suited for standard optimization algorithms, including problems in which the objective function is discontinuous, nondifferentiable, stochastic, or highly nonlinear. All notions like mutation, recombination, selection, fitness of an individual, generations, parents, and offspring are closely related to the process of natural selection.

There are more variations of the basic strategy [4] denoted like: (λ, η) which means that from a population of λ individuals, η offspring will be selected into a new generation- no parent survive, whereas $(\lambda+ \eta)$ symbolize that η individuals for the new generation will be selected from the set of parents and their offspring, etc. Basic operations are defined as follows:

A mutation represents a process of modification of an individual that slightly differs from its "original version", i.e. its features- optimized data, are slightly changed by a stochastic process like in (1):

$$x' = x + N(0, \delta) \qquad (1),$$

where x' is a mutation of the original data vector x that is subject to the optimization process and $N(0,\delta)$ represents Gaussian distribution with zero mean and standard deviation δ.

Fitness is a measure of eligibility or capability of an individual, i.e. how good it is to survive. This measure is task specific, so a method should be provided to classify each individual based on its data that is subject of the optimization.

A recombination is a process of combining several parents who will produce a new child. First a proper pair (or group) of parents must be chosen which can be done in several ways. This process can be completely random or only the best parents according to their fitness are chosen (roulette game) etc. After selecting pairs or groups of parents for each child a method of their mutual combination must be decided. Again there are more manners to do so. Most common are: intermediate – an offspring will be a weighted average of their parents, or discrete - some randomly chosen features will be copied from one parent and the remaining ones will be supplied by the other one, etc.

So far the subject of the evolution was only the optimized vector x. It was assumed that settings controlling the evolution like: standard deviation in the mutation phase, the method for selecting parents and the algorithm for passing attributes from parents to the offspring, were constant during all generations. However, these may evolve as well and presumably better respond to the structure and dynamics of the solved problem. Thus in more sophisticated systems they become optimized in

the same way as the original features. An example of the mutation of the standard deviation is in (2):

$$\delta' = \delta e^{N(0,\delta_0)} \qquad (2),$$

where δ_0 is a constant measure of standard deviation for the whole process, δ is the original deviation and δ' is the newly mutated one. Next, each optimized feature may have its own dispersion unless of course they are normalized. Then in a place of only one value of δ a vector of standard deviations must be used. Furthermore, optimized elements may be correlated. Thus instead of vectors of standard deviations the covariance matrix should be used. For practical reasons rather than mutual correlations of elements the vector of angles is used (rotation of the diagonal covariance matrix). Then the process of evolution from one generation of individuals to the next one can be formally expressed in general terms as follows (3):

$$P^{(t+1)} = sel_{\mu}^{\mu+\lambda}(\bigcup_{i=1}^{\lambda}\{mut(rec(P^{(t)}))\} \cup P^{(t)}) \qquad (3),$$

where P(t) represents individuals in generation t, $sel_{\mu}^{\mu+\lambda}$ determines the operation of selecting μ individuals for the new generation from μ+λ individuals in the current generation. Each individual A is then defined by the original data to be optimized x, vectors of dispersions σ and angles α.

The mutation of individuals is performed after the recombination phase in the following steps: mutation of angles, where the standard deviation is empirically about 5 degrees, mutation of dispersions (diagonal covariance matrix), and finally mutation of the original data vector x based on the newly derived standard deviations and angles. Generally each part of an individual A can be recombined in the different way, thus $\vec{\omega} = (\omega_x, \omega_\sigma, \omega_\alpha)$ represents a vector of possible recombination methods for each part (data, dispersions and angles), and vector $\vec{\rho} = (\rho_x, \rho_\sigma, \rho_\alpha)$ determines the number of parents needed to produce a child.

As the optimization process proceeds the actual solution is supposed to get to some vicinity to the global extreme, so the mutations (dispersions) should be naturally decreasing in the time to produce a stable result. However if the solving problem is hardly to be described, but easily verified it may be better to let the evolution algorithm itself to control the standard deviations and methods of recombination as was previously outlined for the general case.

4. The optimization process of the speech decoding stage

The optimization process and all experiments were accomplished using the MOBILDAT-SK database [6] and its test portion in which 200 speakers are present. As we aimed for a simple dialog application we evaluated records containing looped digits denoted as B1 that does the best approximation of the pursued task (the unknown number of repetition of words from a dictionary). This ensures the proper setting of the speech detection algorithm as well as of some auxiliary grammar parameters. In the tests we used context dependent models of phonemes with 3 states and 8 Gaussian mixtures. As the speech features the MFCC, delta and acceleration coefficients were used (39 elements). The models were trained using the train section of the MOBILDAT-SK database (800 speakers) and the MASPER training procedure.

Table 1. Selected features for the accuracy optimization of the speech decoding process realized in the ATK system.

feature	value	Description
SILSEQCOUNT	109	Number of frames classified as silence to detect the silence
SPCSEQCOUNT	22	Number of frames classified as speech to detect the beginning of speech
SPEECHTHRESH	9.51	Energy threshold for the speech detection. It is relative to the estimated silence level.
SILENERGY	10	Mean energy of silence (background noise) in dB
SPCGLCHCOUNT	3	Number of frames below the threshold that can be ignored when counting SPCSEQCOUNT
SILGLCHCOUNT	10	Number of frames above the threshold that can be ignored when counting SILSEQCOUNT
SILMARGIN	17	Shift the beginning and end of the detected speech given the number of frames.
CMNTCONST	0.9984	The time constant for cepstral mean adaptation
CMNMINFRAMES	26	Minimum number of frames after which a cepstral mean adaptation can take place
WORDPEN	-81	Log probability penalisations for a word insertion error

In the next stage we had to choose some of the most important parameters that we believed can play an important role for the recognition process employed in a

real dialog application. It is vital to limit their number in order to speed up the convergence. It should be noted that playing all eligible records takes 30 minutes, so to assess the fitness for a single individual in a given generation takes about 30 minutes. The selected parameters for the optimization process mainly govern the speech detection algorithm the ATK system [1], [2] uses and adjust some grammar related issues. In table 1 these features are listed together with their description and "optimally" set values.

As all of these parameters have their physical interpretation and different ranges of acceptable values they were transformed to the unified range <0, 1> according to (4).

$$fe' = \frac{fe - \text{lower\_limit}}{\text{upper\_limit} - \text{lower\_limit}} \qquad (4),$$

where *fe* is the original value of the feature, *fe'* is the transformed feature, *lower_limit* and *upper_limit* are the limit values of a feasible range.

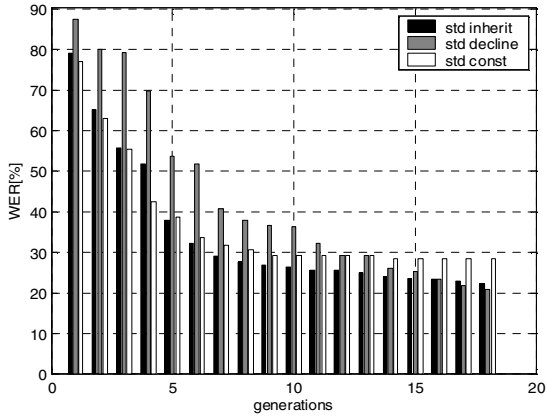

Figure 1. The mean word error rates per generations for gradually declining, inherited and constant standard deviation.

Then in the process of recombination or mutation this range had to be checked. If for some features it was not met the upper or lower limit was substituted, to ensure the physical meaning. Such normalization enabled us to use only one standard deviation for the vector as a whole because all features had approximately the same dispersion. Furthermore we decided to automatically decrease the dispersion with the increasing number of generations to reach a stable solution, however tests with fixed and inherited (optimized) dispersion were executed as well. Because of the time consumption we tried to limit the number of generations and the number of individuals in each generation. To do so we neglected some possible correlations among optimized features.

As we were not completely sure with the "ideal" settings of the optimization process we tested more versions like: constant, decreasing and optimized deviation, various methods for parents' selection and recombination, number of generations etc. In fig. 1 the mean error mistakes in each generation are depicted for the constant deviation (0.3), gradually declining deviation ranging from 0.35 to 0.02 and the optimized deviation by the evolution process, where in all cases the parents were selected stochastically according to their fitness.

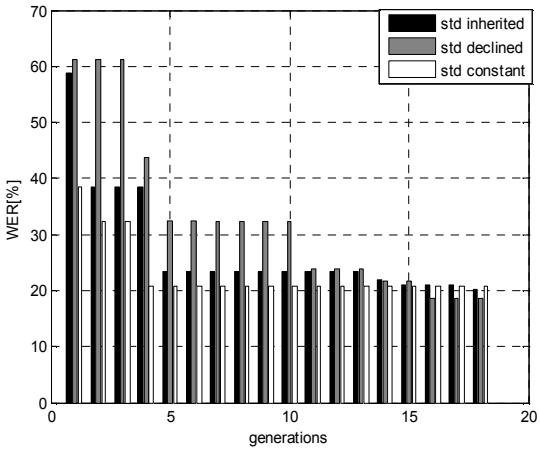

Figure 2. The word error rates of best individuals in each generation for gradually declining, inherited and constant standard deviation.

In all cases an average error per generation is depicted and as it can be expected it is declining (exponentially). This is due to the so called crowding phenomenon, where all individuals tend to exhibit similar features during the evolutional process. If this is undesirable a life time control can be introduced. Another interesting view is provided in fig. 2 showing the error for the best individuals in each generation. As it can be seen, the course of decline is rather abrupt and unequally spread, however the major changes are more likely to happen at the beginning as the whole process is converging.

5. Conclusions

Although the model training phase is vital for a successful recognition system the proper adjustment of the decoding process plays an important role as well. It is relevant not only to achieve the declared accuracy usually observed in the offline experiments which provide better results, but also to speed up the decoding process and to spare the time and computational recourses. As there are many parameters to be optimized together, which are mutually related either positively or negatively to each other, usually a common sense approach applied to the process of their manual adjustment may be far from optimum.

By using the abovementioned method we achieved 18.72% WER (offline tests using the standard evaluation procedure defined in REFREC or MASPER for the looped digits test showed only 1.17% WER), which is about 5% lower than those reached when adjusted

manually (generally above 23%). As presented, the utilization of evolutional strategies brought promising results at low cost (we tested only 98 individuals through 10 dimensional space, which is less than 1.6 search points per dimension) similarly as it is reported in other areas e.g. [8].

From the tested settings of evolutional strategies (recombination methods, dispersion control) it is shown in figure 1, that the gradual decrease of the dispersion has brought better final results within the range of tested generations, which follows theoretic expectations. However, this approach showed slightly worse results in the early stages, which may be caused by the higher dispersion which aimed to cover wider area at the beginning. In the case of "unlimited" evolution time the deviation should be kept optimized as well and some kind of life time control introduced, in order no to get stuck in the local minima.

Features like WORDBEAM, MAXBEAM and GENBEAM that were involved in the pruning of the decoding process were not set during the optimization process. The fitness of each individual was based solely on the recognition accuracy (WER). Abovementioned parameters introduce some approximations in order to speed up the computation time, but could deteriorate the accuracy. To hit the balance a new fitness must be used utilizing the computational load as well. This, however, would be highly dependent on the perplexity of the actual dialog system.

Usually the incorporation of Lamarckian evolutional strategy [3] can bring even better results, that is, some individuals are additionally trained during the evolution, to reach for a local minimum in their vicinity. Usually this is done by hill climbing algorithm and such "learned" information is passed to the next generation (this opposes the Darwin theory). Unfortunately, in the case of costly evaluation of the fitness function, this option looses its benefits as more individuals must be examined, which can be made up for by allowing more generations in the classical evolutional strategy optimization.

Acknowledgement

The authors hereby declare that they work has been supported by the project: VEGA 1/3110/06

6. References

[1] S. Young, *An Application Toolkit for HTK,v. 1.4.1*, Cambridge University, 2004

[2] S. Young, G. Evermann, T. Hain, *The HTK Book V.3.2.1*, Cambridge University Engineering Department, Dec. 2002

[3] L. D. Chambers, *Practical Handbook of Genetic Algorithms, Complex Coding Systems, Volume III*, CRC Press, ISBN 0-8493-2539-0, 1999

[4] M. Dianati, I. Song, M. Treiber, *An Introduction to genetic Algorithms and Evolution Strategies*, Technical report, University of Waterloo, Ontario, Canada, 2002.

[5] Thomas M. Mitchell, *Machine Learning*, McGraw-Hill Higher Education, ISBN:0070428077, 1997

[6] S. Darjaa, M. Rusko and M. Trnka, *MobilDat-SK - a Mobile Telephone Extension to the SpeechDat-E SK Telephone Speech Database in Slovak*, Proceedings of the 11-th International Conference Speech and Computer (SPECOM'2006),"Anatolya" publishers, St. Petersburg 2006, pp. 449-454, ISBN 5-7452-0074-x

[7] J. Juhar, S. Ondas, A. Cizmar, M. Rusko, G. Rozinaj, R. Jarina: *Development of Slovak GALAXY/VoiceXML Based Spoken Language Dialogue System to Retrieve Information from the Internet*, The Ninth International Conference on Spoken Language Processing (Interspeech 2006 — ICSLP), Sept 17-21, 2006 in Pittsburgh, Pennsylvania, 4 pp. ISSN 1990-9772

[8] J. Filanova, *Optimisation of Multipoint Routing in Telecommunication Networks*, In: Proc. of the Int. Conf. Research in Telecommunication Technology RTT 2004, Cesky Raj, Czech Republic, 15.-17. Sept. 2004, CD ROM.