ON A PRACTICAL DESIGN OF A LOW COMPLEXITY SPEECH RECOGNITION ENGINE

Marcel Vasilache, Juha Iso-Sipilä, Olli Viikki

Audio-Visual Systems Laboratory
Nokia Research Center, Tampere, Finland

ABSTRACT

In this paper we outline the main design features of a low complexity speech recognition engine targeted for mobile devices. Although major parts have already been presented, new features and important refinements of the original ideas, which were omitted, are now described. We also show how these techniques can be successfully combined in order to achieve various design targets with minimized impact on the recognition performance.

1. INTRODUCTION

During the recent years the field of automatic speech recognition (ASR) has witnessed an intense activity in the area of complexity reduction. The main target of such research is of making possible increasingly more complex recognition tasks using only the limited capabilities of portable devices. Even though the hardware capabilities of such mobile devices are on a constant increase, the importance of low complexity systems will continue to remain very high. A main driver for this is the need of increasing the battery time by minimizing the energy consumption of the underlying algorithms.

With this paper we aim to illustrate a few key complexity reduction techniques and their influence in the design process for a practical speech recognition engine. The paper is organized as follows: in the next three sections we briefly review the main blocks of a speech recognition engine, namely: acoustic modeling, decoding and speaker adaptation. For each of these sections low complexity design paradigms are described. In the experimental section, with the aim of selecting effective design targets for a practical system, we present how these techniques are evaluated and can be combined. Finally, two design alternatives are isolated and conclusions are drawn.

2. ACOUSTIC MODELING

Hidden Markov Models (HMM) are a very successful tool in ASR and among them the continuous density HMMs are the most widely used. In addition to their good modeling capacity they have proved to be highly robust regarding the parameter values and, as consequence, highly compressible. Various compression options have been applied, the vast majority being focused on quantization procedures for the mean and variance vectors of state densities [6].

As previously presented in [1], joint scalar quantization is a very effective and, possibly, one of the simplest methods. Similarly to the alternative approaches, it is capable of achieving both a reduced memory representation as well as a reduced computational cost.

3. DECODING

3.1. State emission likelihoods

The decoder complexity is heavily influenced by the cost of computing state emission likelihoods (which we will refer shortly as B-probs). This part can consume more than 60% of the total recognition costs even with the efficient approach allowed by qHMMs. For mixture of Gaussians likelihoods the most expensive part consists on the evaluation of the Mahalanobis distance for each density. For a single component the formula reduces to:

$$d_{ki} = \frac{(\mu_{ki} - o_i)^2}{\sigma_{ki}^2} \qquad (1)$$

where μ_{ki} and σ_{ki}^2 are the i^{th} mean respectively variance component of the density k and o_i is the i^{th} component of the observation vector. The sum of d_{ki} for all dimensions of the observation vector (the feature space) gives the required distance value for the density k. With a quantized representation for means and variances it is readily apparent that given the observation vector the set of possible values for d_{ki} has the size of the product of

mean and variance quantizer levels. With low rate quantization this results in a small set which makes pre-computing these values advantageous. By storing them into tables, one for each feature vector dimension, the d_{ki} values are obtained by indexing with the joint mean and variance quantizers index. This reduces B-prob computation to table indexing and summations. In [1] estimates are given of the number of floating point operations required by the original and by the indexed approach. It is shown that, for a certain critical level of densities, it is more effective to use the helper tables. Since, in practical systems, the arithmetic operations alone are not uniquely responsible of processor cycles both methods need to be evaluated in order to determine the fastest approach.

If in formula (1) the observation vectors are also quantized, the computation of the helper tables at each frame can be completely avoided, provided that the product quantization levels are manageable in terms of the required storage. Since, usually, the quantization rate of the mean parameters is low, is it not required to use a high accuracy representation of the observation vectors because the induced error is already determined by the precision of the mean parameter storage. A comparable quantization rate or even identical quantizers can be considered.

3.2. The Viterbi algorithm

Once B-probs have been obtained the computation of the best alignment of the HMM states to the input sequence of feature vectors is done using the Vitebi algorithm. A token passing approach [7] is used. For a phoneme based isolated word recognition task, as for instance name dialing, it is advantageous to minimize the Viterbi token passing structure with the help of prefix sharing. By this, a tree structured recognition grammar is established. It is known [8] that optimal structures can be derived in terms of the total number of phonemes required. However, since usually the optimal structures are less regular and much more expensive to construct dynamically, the tree provides a good trade-off between compactness and representation regularity. Its major advantages are that it requires minimal additional resources to keep its structure (as low as one bit per phoneme instance), offers the possibility of memory localized decoding and hence good processor cache utilization and, not the least, in comparison with a linear grammar organization, it has better efficiency when a beam search Viterbi is used.

4. SPEAKER ADAPTATION

For maximizing the recognition performance speaker adaptation is a mandatory procedure. Among the available alternatives, Bayesian adaptation was chosen due to good performance, simplicity and minimal complexity overhead. A more in-depth description of the adaptation algorithm and its performance is given in [3]. In the following we focus only on the complexity aspects. Two complexity reduction methods are described next.

4.1. Single utterance adaptation

The simple means adaptation formula

$$\mu_k' = \frac{\tau\mu_k + \sum_{t=1}^{T}\gamma(t,k)\mathbf{x}_t}{\tau + \sum_{t=1}^{T}\gamma(t,k)} \quad (2)$$

requires the computation and storage of a set of accumulator parameters (for the sums in (2)). Once enough adaptation utterances are observed this set is used to compute the replacement for the original parameters. However, doing this will also substantially increase the memory requirement for acoustic model storage. In a supervised framework it was observed that adaptation after each utterance produces good results. Therefore, with careful programming of (2), the memory requirements for accumulators are reduced to the equivalent required for adapting a single Gaussian density. Furthermore, no long-term storage of the accumulators is needed.

4.2. Quantization of the features

Following 4.1 the major memory requirement is imposed by the storage of the feature vectors for the most recent utterance. To minimize this cost feature vector quantization can be used. Since the HMM parameters are already heavily quantized, it is to be expected that the quantization of the feature vectors with a higher or comparable rate will not significantly influence the adaptation performance. In this respect, the existing scalar quantizers for the mean parameters of qHMMs can provide a simple and efficient solution.

5. EXPERIMENTS

The evaluation was carried out using a multi-lingual, speaker independent, name recognition task. Feature extraction was performed using FFT derived Mel cepstral coefficients. The 1st and 2nd order time derivatives were included which resulted in 39 dimensional feature vectors. For noise robustness a normalization scheme [5] was enabled as the final front-end stage. The three state phoneme models had a left-to-right structure with no skips. ML training was performed on a diverse set of languages. Although the system was trained to simultaneously handle a large language set, we are presenting here results for only two out of the 27

languages covered. The selected languages are German and English. The recognition grammar consisted of names composed of one or several parts. Slightly more than 100 name entries were active for each language and a total of 11000 utterances were selected for testing. Two testing environments were used; "clean" which contains the original waveform recordings and "noise" which was created by mixing various noise types with SNR ranges from 5 to 20 dB.

In the first set of experiments, using the original set of models, we evaluated the effect of switching the engine from floating point mode ("float") into fix point mode ("fxp"). Typically, 16 bit integer representations were used. Both the speaker-independent ("SI") and the speaker-adapted ("SA") rates are presented below.

	SI fxp	SI float	SA fxp	SA float
Clean	95.05	95.14	98.02	97.97
Noise	84.83	85.09	92.40	93.07

Table 1 Fixed point vs. floating point performance

The negative impact is most visible in the more challenging "noise" environment. Since floating point computation is prohibitively expensive in an embedded environment both because of memory as well as computational complexity (e.g. emulation may be required) this design step and performance impact are unavoidable.

In the following step we focused on the problem of HMM quantization. A large range of quantization rates for means and variances was evaluated, as illustrated in Table 2 and Table 3. The quantization rate for means is placed at the start of each line and the rates for variances are on top of the columns. All the quantizers were non-linear, Lloyd-Max trained on the parameters of the original models.

		Clean	SI		
m \ v	0	1	2	3	4
1	51.00	72.49	77.48	n/a	n/a
2	91.51	93.00	93.74	93.84	n/a
3	92.75	94.40	95.03	95.14	n/a
4	92.91	94.51	95.01	94.93	94.94
5	93.08	94.56	95.08	95.01	95.05
6	92.99	94.45	95.14	94.99	94.95
7	93.03	94.48	95.11	94.79	94.87
		Noise	**SI**		
m \ v	0	1	2	3	4
1	22.39	38.01	39.88	n/a	n/a
2	81.01	82.97	83.37	83.71	n/a
3	82.03	83.77	84.23	84.58	n/a
4	82.34	84.10	84.46	84.65	84.82
5	82.52	84.30	84.50	84.76	84.65
6	82.59	84.33	84.53	84.80	84.80
7	82.48	84.50	84.44	84.79	84.76

Table 2 Recognition rates with HMM quantization

		Clean	SA		
m \ v	0	1	2	3	4
1	56.60	76.20	81.78	n/a	n/a
2	95.63	96.37	96.52	96.36	n/a
3	97.05	97.51	97.77	97.66	n/a
4	97.37	97.79	97.97	97.95	98.05
5	97.45	97.86	98.17	97.97	98.01
6	97.53	97.92	98.03	97.96	98.06
7	97.54	97.82	98.04	97.97	97.98
		Noise	**SA**		
m \ v	0	1	2	3	4
1	24.15	39.26	41.27	n/a	n/a
2	87.13	88.00	88.07	87.93	n/a
3	90.24	90.86	91.34	91.30	n/a
4	91.08	92.23	92.71	92.43	92.36
5	91.11	92.40	92.50	92.82	92.56
6	90.81	92.10	92.72	92.70	92.48
7	90.61	92.16	92.80	92.62	92.51

Table 3 Recognition rates with HMM quantization and speaker adaptation

From Table 2 and Table 3 we observe that for higher rates the performance saturates close to the value of the original models. Due to practical considerations (an even packing of mean-variance pairs into bytes) 5m3v and 3m1v are the most interesting design points. If pressed by hard complexity limits, even extreme quantization rates such as a 2-bit rate for the means with a global variance, can be considered.

Although the required storage for the quantizers is small (e.g. only 40 values for a 5-bit + 3-bit pair) we also investigated the performance of optimal linear quantizers. In comparison with the non-linear ones, they had similar performance except for the lower rates where a higher degradation was noticeable. Only the "Clean SI" results are shown in Table 4.[1]

		Clean	SI		
m \ v	0	1	2	3	4
1	51.00	72.49	73.59	n/a	n/a
2	90.46	92.52	93.34	93.53	n/a
3	92.44	94.03	94.38	94.32	n/a
4	92.79	94.45	94.72	94.92	94.69
5	93.04	94.59	95.00	95.14	95.18
6	93.10	94.57	95.05	94.93	94.90
7	93.10	94.52	94.96	95.08	94.94

Table 4 Recognition rates with uniform HMM quantization

As presented in section 3.1 it is very attractive to avoid computing the B-prob helper tables at each frame. In order to do this we are quantizing the features only for the

[1] This is due to lack of space and a similar observed behavior also in the other three testing cases.

purpose of an optimized B-prob computation. The results of these experiments are visible in the "SI" and "SA" columns of Table 5. The naming convention encodes the quantization rates for means, variances and features (i.e. "3m1v4f" states rate 3 for mean, 1 for variances and 4 for the feature components). For reference, the original models "orig" and quantized models with accurate B-prob computation are also included (i.e. the table entries which are not ending in "f").

Based on these results this method has negligible impact on the recognition performance. Surprisingly, in a few cases even better performance is measured.

Clean	SI	SA	qSA
3m1v3f	94.14	97.52	97.48
3m1v4f	94.52	97.70	97.55
3m1v	94.40	97.51	97.58
5m3v3f	94.35	97.74	97.87
5m3v4f	95.02	97.97	97.95
5m3v5f	95.09	97.91	98.00
5m3v	95.01	97.97	97.99
orig	95.05	98.02	97.96
Noise	**SI**	**SA**	**qSA**
3m1v3f	83.98	91.15	91.20
3m1v4f	83.90	90.83	90.76
3m1v	83.77	90.86	90.69
5m3v3f	84.34	92.41	92.51
5m3v4f	84.66	92.76	92.78
5m3v5f	84.71	92.80	92.60
5m3v	84.76	92.82	92.69
orig	84.83	92.40	92.25

Table 5 Results of feature quantization for B-prob computation and adaptation

As discussed in Section 4.2, storing quantized features can substantially reduce the memory requirements for adaptation. The impact on the adaptation performance is visible in the "qSA" column of Table 5 where a 4-bit quantizer is used in all cases. This quantizer is identical to the one used in the B-prob acceleration for "3m1v4f" and "5m3v4f". A minimal effect is observed for all cases.

Finally, following the ideas in [4] we tested the influence of halving the frame frequency of the B-prob computations. We have used the two most interesting design targets.

	SI	SI/2	qSA	qSA/2	
3m1v4f	94.52	94.38	97.55	97.51	Clean
5m3v4f	95.02	94.87	97.95	97.82	
3m1v4f	83.90	83.47	90.76	90.34	Noise
5m3v4f	84.66	84.25	92.78	92.04	

Table 6 Results with computation of B-probs every 2nd frame

As shown in Table 6 in columns "SI/2" and "qSI/2", for both cases the impact is only marginal in comparison to the reference rates in columns "SI" and "qSA". Nevertheless, the complexity reduction is substantial.

Finally, after all these steps, we can conclude that in practice, for this design example, "5m3v4f" provides a good solution, with substantial complexity reductions, when high performance is desired. For stronger complexity constrains, "3m1v4f" gives a very good compromise.

6. CONCLUSION

In this paper we addressed several topics related with the practical design of low complexity speech recognition engines. The structure of such engines allows a large degree of freedom in choosing the target operating points. Several low complexity techniques were presented and evaluated. In the context of a speaker independent name-dialing task we have shown that these techniques can be successfully combined in order to achieve various design targets with minimized impact on the recognition performance.

7. REFERENCES

[1] Vasilache, M., "Speech recognition using HMMs with quantized parameters", *Proc. of International Conference on Spoken Language Processing*, vol.1, pp. 441-443, Beijing, China, 2000.

[2] Riis, S. K., Viikki, O., "Low complexity speaker-independent command word recognition in car environments", *Proc. of International Conference on Acoustics, Speech, and Signal Processing*, Istanbul, Turkey, 2000.

[3] Vasilache, M., Viikki O., "Speaker adaptation of quantized parameter HMMs", *Proc.Eurospeech-Scandinavia*, vol. 2, pp. 1265-1268, Aalborg, Denmark, 2001.

[4] Kiss, I., Vasilache, M., "Low complexity techniques for embedded ASR systems", *Proc.ICSLP 2002*, pp. 1593-1596, Denver, USA, 2002.

[5] Viikki O., Bye D., and Laurila K., "A recursive feature vector normalization approach for robust speech recognition in noise", *Proc. of International Conference on Acoustics, Speech, and Signal Processing*, pp 733-736, Seattle, WA, USA, 1998.

[6] Filali, K., Li, X., and Bilmes J., "Data-driven vector clustering for low-memory footprint ASR", *Proc. ICSLP 2002*, pp. 1601-1604, Denver, USA.

[7] Young, S.J., Russel, N.H. and Thornton, J.H.S., "Token Passing: a Simple Conceptual Model for Connected Speech Recognition Systems", Cambridge University Engineering Department, July, 1989.

[8] Mohri, M., Pereira, F., and Riley M., "Weighted finite state transducers in speech recognition", *Computer Speech and Language,* 16(1) pp 69-88, January 2002.