Truncation Error and Dynamics in Very Low Latency Phonetic Recognition

Giampiero Salvi

Dept. of Speech, Music and Hearing,
KTH (Royal Institute of Technology)
Drottning Kristinasv. 31, 10044 Stockholm, Sweden
giampi@speech.kth.se

***Abstract*— The truncation error for a two-pass decoder is analyzed in a problem of phonetic speech recognition for very demanding latency constraints (look-ahead length $<$ 100ms) and for applications where successive refinements of the hypotheses are not allowed. This is done empirically in the framework of hybrid MLP/HMM models. The ability of recurrent MLPs, as *a posteriori* probability estimators, to model time variations is also considered, and its interaction with the dynamic modeling in the decoding phase is shown in the simulations.**

I. Introduction

In real-time applications conventional two-pass decoders based on different flavors of the Viterbi algorithm [1], can only be used in an approximate fashion. The approximation lies in the need for incremental results that limits the length of look-ahead, or equivalently requires the back-tracking phase to be truncated to a certain number of frames. Truncation error in the Viterbi and Best-Path (BP) algorithms has been extensively studied for convolutional codes in the area of speech coding [2], [3]. There, given the relatively simple nature of the problem, error bounds could be found analytically and confirmed empirically.

In speech recognition, few empirical studies dealing with this problem can be found (e.g. [4], [5]). In [5] a system based on incremental hypothesis correction was shown to asymptotically reach the optimal MAP solution. These studies are restricted to the (large vocabulary) word recognition case and deal with look-ahead lengths of the order of 200ms.

The aim of the current study is to analyze the effect of truncation errors at very low latencies (look-ahead < 100ms) in phonetic recognition. In the application we have in mind [6] the resulting phone string is fed into a rule system that in turns creates articulatory parameters for a synthetic face. In this conditions, and since the delay between the incoming speech and the resulting face movements must be short, successive refinement of the hypothesis is not allowed.

II. Problem definition and notation

A. Speech production

The process of speech production could be seen as the one of encoding a sequence of symbols $X_1^M = (x_1, \cdots, x_M)$ into a sequence of states $S_1^N = (s_1, \cdots, s_N)$ with an associated output sequence $U_1^T = (u_1, \cdots, u_T)$. In our oversimplified description, X_1^M could represent phonemic intentions, or as in our case phonetic classes, S_1^N are motivated by the dynamics introduced by articulatory gestures that in turn generate the speech signal U_1^T. Phonetic speech recognition is then the process of recovering the right sequence X_1^M on the base of some features $Y_1^N = (y_1, \cdots, y_N)$ extracted from U_1^T. When the feature extraction procedure is assumed to be given, as in the current study, the distinction between U and Y is not essential. Speech production is then a (stochastic) function of the kind: $P : X \to Y$. The natural choice for characterizing this function is a Markov model Θ where the states s_i are assumed to vary synchronously with the features y_j, which explains why we indicated the length of S and Y with the same symbol N. Besides an *a priori* term, Θ is then fully specified by the distribution of state transition probabilities $a_{ij} = P(s_j|s_i)$ and the distribution of data generation given a certain state $b_i(Y_h^k) = P(Y_h^k|s_i)$. Usually the dynamics of the process are considered to be represented uniquely by the a_{ij}, being the $b_i(Y_h^k)$ static models ($Y_h^k \equiv y_h$). As the notation indicates, we are interested in the case in which each state influences the output at different time steps.

B. State to output probability estimators

Robinson [7] has shown how recurrent Multi Layer Perceptrons (MLPs) can be efficient estimators for the *a posteriori* probabilities $P(x_i|Y_1^n)$. A particularly efficient training scheme uses Back Propagation through time [8] with a cross entropy error measure [9]. If the nonlinearity in the units is in the $\tanh$ form, we can write for the state to output probabilities:

$$P(Y_h^k|x_j) = \frac{P(x_j|Y_h^k)P(Y_h^k)}{P(x_j)} \simeq \frac{a_j + 1}{2} \; \frac{P(Y_h^k)}{P(x_j)} \qquad (1)$$

Where x_j is a phonetic class and a_j the activity at the output node corresponding to that class. Y_h^k is the sequence of feature vectors spanning a window of time steps that depends on the dynamic properties of the MLP. In the case of simple feed-forward nets Y_h^k reduces to the current frame vector y_k, while for strict recurrent topologies, $h = 1$ and k is the current frame. In [10] a mixture of time delayed and recurrent connections was proposed. In this model the input layer received contributions both from the past and the future frames thanks to time delayed connections with possibly negative delays. Ström showed that the network took advantage mostly

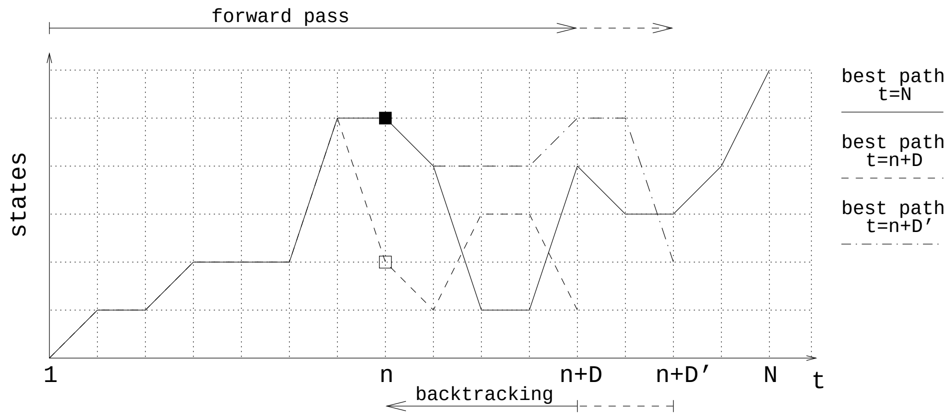

Fig. 1. Trellis plot in different conditions: the continuous line shows the best Viterbi path over the whole observation sequence. The dashed and dashed-dotted lines are obtained with a forward pass up to times $n+D$ and $n+D'$. Depending on the length of the backtracking phase, the solution at time n can differ or coincide (open and filled squares respectively) with the standard Viterbi solution

from future contributions, as information about the past was successfully coded by the recurrent connections.

C. Problem 1: Modeling dynamics

Given the probabilistic model Θ, the *maximum a posteriori* (MAP) solution to the speech recognition problem is the sequence X_1^M that maximizes

$$P(X_1^M|Y_1^N,\Theta) = P(x_1,\cdots,x_M|y_1,\cdots,y_N,\Theta)$$

A more pragmatic solution, provided by the Best-Path and Viterbi algorithms, approximates the sum over all possible state sequences, implicit in the formula above, with a maximum operation. Since in our model X_1^M is fully determined by S_1^N, the latter is equivalent to finding the sequence S_1^N for which $P(S_1^N|Y_1^N,\Theta)$ is maximum. This can be done iteratively according to the formula

$$\delta_t(j) = \max_i[\delta_{t-1}(i)a_{ij}]b_j(y_t)$$

Where $b_j(y_t) = P(y_t|x_j)$. In practice we substitute to the last the estimate of $P(Y_h^k|x_j)$ given by Equation. 1. In the case of recurrent MLPs, $P(Y_h^k|x_j) = P(Y_1^t|x_j)$ and the information contained by $\delta_{t-1}(i)$ and $b_j(Y_1^t)$ become widely overlapping. As a result we can expect a reduction in effectiveness of the Viterbi algorithm as compared to the case in which those two terms were based on independent information.

D. Problem 2: Truncation in the Viterbi algorithm

When truncation is considered, the optimal solution at time step n is the state s_n extracted from the sequence $S_1^{n+D} = (s_1,\cdots,s_n,\cdots,s_{n+D})$ that maximizes $P(S_1^{n+D}|Y_1^{n+D},\Theta)$, where D denotes the look-ahead length in time steps. The difference between the two approaches is exemplified in Figure 1. The grid displays the states as a function of time (trellis). The continuous line shows the Viterbi solution, while the dashed and dashed-dotted lines refer to the best path obtained using the partial information up to $t = n+D$ and $t = n+D'$, respectively. The Figure also shows a phenomenon that is common in practice: the influence of an observation at time t_1 over the result at time t_2 decays with the distance $D = |t_1 - t_2|$.

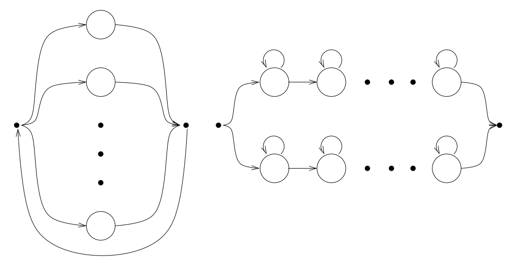

Fig. 2. Two extreme examples of transition topologies. Left: the net defines short time dependencies, at any time step any state is reachable. Right: long time dependences, the state at time t is dependent on the state at time $t+n$ for any value of n

In the example the observations in the interval $[n{+}D{+}1, n{+}D']$ influence the result at time n (open and filled squares), while the ones in $[n+D'+1, N]$ don't. As a result the truncated solution will in general asymptotically approach the standard Viterbi solution (filled square in this case) as D increases. The value D^* at which the two solutions become indistinguishable depends on the dynamic characteristics of the problem at hand, that is on the time correlations in Y and especially on those imposed by Θ. Extreme examples of the last are shown in Figure 2. On the left side the transition probabilities define a network in which the maximum of the minimum lengths of independent paths is 1: at every time step, any state in the network can be reached. On the right side a network is depicted in which the minimum length is the length of the whole observation sequence (this corresponds for example to sentence recognition with only two alternative hypothesis): at any time step only one (or two) states are reachable depending on the sentence. In the first case we can expect D^* to be small, while in the second much larger: evidence at the end of the utterance could cause an hypothesis change that would affect results in the beginning of that utterance.

III. Computer Simulation

Given the difficulty to derive an analytical solution to the problems defined above, computer simulations were run in order to empirically estimate their effect.

A. Data

The experiments were performed on the Swedish SpeechDat corpus, containing recordings of 5000 speakers over the telephone line. The official training and test sets defined in SpeechDat and containing respectively 4500 and 500 speakers, were used in the experiments. Mel frequency cepstrum features were extracted at 10ms spaced frames. When using Gaussian mixture models (GMMs), the delta and delta-delta coefficients were computed on a window of 3 frames in both directions. The GMM models were trained on the full corpus defined in SpeechDat, while the training material for the neural networks (MLPs), and the test material were restricted to the phonetically rich sentences ("s" codes). The training material for the MLPs was further divided into training and validation sets of 33062 and 500 utterances respectively. One problem with the SpeechDat database, that is important when training the MLPs and for testing at the frame level, is the unbalanced amount of silence frames if compared with the amount of material for any phonetic class. As the silence frames are mostly concentrated in the beginning and end of each utterance, their partial removal was facilitated.

B. Phonetic Transcriptions

Since the dataset lacks phonetic transcriptions, some preprocessing was necessary. The time-aligned reference, necessary for both training the MLP models and testing, was obtained with forced alignment based on word level transcriptions, a lexicon and context dependent Gaussian mixture models (CDGMMs). This method is strongly dependent on the quality of the lexicon: a poor lexicon (i.e. with limited pronunciation variation) will force mistakes into the transcriptions that are difficult to detect automatically. The solutions ofter proposed to this problem aim at increasing the degrees of freedom in the alignment network, giving to the alignment process greater flexibility without affecting the effectiveness of the method. This in general requires linguistic-phonetic resources, that allow defining pronunciation variations based on rules or a lexicon. Given the unavailability of such resources, we adopted a simple alignment network with optional silence or noise between words. This method proved to be reasonably accurate, but a more detailed inspection is required to guarantee the quality of the result. To be noted here is the fact that this reference will in general favor recognition methods based on Viterbi decoding and Gaussian mixture models that were used to obtain it.

C. Acoustic Models

As described above, the acoustic models used in this study are Gaussian Mixture Models (GMMs) and Multy Layer Perceptrons (MLPs). The first were used both in the data processing phase, as already mentioned, and in the recognition experiments as an example of static models, i.e. models that don't retain an internal representation of time variations. This in spite of the use of dynamic features, the span of whose is infact limited to a few frames. The GMMs were trained using the procedure defined in the RefRec scripts [11], that produces a set of monophones and triphones of varying complexity. The best models, consisting of a set of triphones with 32 Gaussian components (GCs) for each state, were used for forced alignment, A set of monophones with 32 GCs per state (`GMM`) was used for recognition.

model	# param.	# hidd.u.	# hidd.l.	RMLP?	f-by-f MAP
`GMM`	379050	-	-	-	35.4%
`ANN`	186050	800	2	no	31.5%
`RNN1`	185650	400	1	yes	49.6%
`RNN2`	541250	400	1	yes	54.2%

TABLE I
DETAILS ON THE ACOUSTIC MODELS

The neural networks used in the experiments were a feed-forward perceptron (`ANN`) and two recurrent perceptrons (`RNN1` and `RNN2`) trained as described in section II-B.

Details on the acoustic models are reported in Table I, which shows the overall number of parameters, and, in the case of perceptrons, the number of hidden layers and hidden units and the dynamic characteristics. The last column reports the frame correct rate when the *maximum a posteriori* class (MAP) was selected frame-by-frame. This will be considered as a baseline for Problem 1 where we are interested in the increase in performance introduced by the Viterbi processing.

Note that the topology in `ANN` was chosen in order to obtain a network that could compare with `RNN1` in terms of complexity (number of free parameters).

D. Scoring method

The scoring method chosen for this study is frame-by-frame correct rate simply computed as the ratio between number of correctly classified and total number of frames. This method was preferred to the more common minimum edit distance at the symbolic level, because the application we have in mind requires not only correct classification of the speech sounds, but also correct segment alignment.

E. Implementation note

GMM training was performed using the HTK Toolkit [12]. The MLP training algorithm was implemented in the NICO Toolkit [13], the two-pass algorithm, and the other tools used in the experiments were implemented by the author. All experiments were performed on a GNU-Linux platform running on standard hardware (PC).

IV. Results

Results are shown in Table II and Figure 3 for Problem 1 and Problem 2 respectively, and will be discussed separately.

A. Problem 1

Here the question is to verify whether probability estimators that model time variations could benefit from the use of the Viterbi decoder to the same extent as static models. As can be seen in Table II, this is not the case for the recurrent perceptrons. If the percentage of correct classification for the GMMs has $\sim$19% relative improvement, when using the

decoder	static		dynamic	
	`GMM`	`ANN`	`RNN1`	`RNN2`
f-by-f MAP	35.4	31.5	49.6	54.2
Viterbi	42.1	32.8	50.7	55.3
rel. impr.	19%	4%	2%	2%

TABLE II

CORRECT FRAME RATE AND RELATIVE IMPROVEMENT

Viterbi decoder, this improvement is much lower in the case of RNNs (∼2%). This seems to validate what intuitively suggested in Section II-C, i.e. the information contained in the recognition network, used in the Viterbi decoder, is not independent from that learned by the perceptron weights. However, the feed-forward perceptron (static model) shows an improvement (∼4%) that is similar to what obtained with dynamic models suggesting that the problem might lie elsewhere.

Note that the Viterbi results in this case correspond to a Viterbi decoder truncated to 30 frames (300ms). The goodness of this approximation, already discussed in Section II-D, is supported by [14].

B. Problem 2

Problem 2 aims at verifying how truncation in the Viterbi decoder affects results in the case of phonetic speech recognition. The results are shown in Figure 3, which displays the frame correct rate as a function of the look-ahead length used in the truncated Viterbi decoding. This in the case of Gaussian models (`vit-GMM`) and for the two recurrent perceptrons (`vit-RNN1` and `vit-RNN2`). For completeness the frame-by-frame MAP solutions are also reported in the different cases as horizontal lines (`map-GMM`, `map-RNN1` and `map-RNN2`).

As the same GMM models and experiment settings developed by the author, have been used in [14] to test another two-pass decoder, the results there obtained are reported as a reference (`dec-gmm`, X marks). These closely agree with our results, where the small differences are probably due to the fact that the test material is in our case pushed into the decoder as a continuous stream to avoid boundary effects.

The curves in Figure 3 show that both in the GMM and RNN case, results are stable and equivalent to the standard Viterbi, when the look-ahead, or truncation length, is greater then 10 frames (100ms). For shorter lengths, the performance drops, first slowly and then abruptly, even below the frame-by-frame MAP solution. This behavior seems to depend more strongly on the dynamic characteristics of the recognition network, then on those of the probability estimators, although the slope of the curve seems to be steeper in the RNN case.

V. Conclusions

The truncation error for a two-pass decoder has been shown to be negligible for truncation lengths greater then 100ms in the case of phonetic recognition. This both for Gaussian mixture models (GMMs) and recurrent perceptrons (RNNs). The advantage in using a two-pass decoder in the place of a simple maximum *a posteriori* frame-by-frame decision, results to be grater in the case of Gaussian probability estimators.

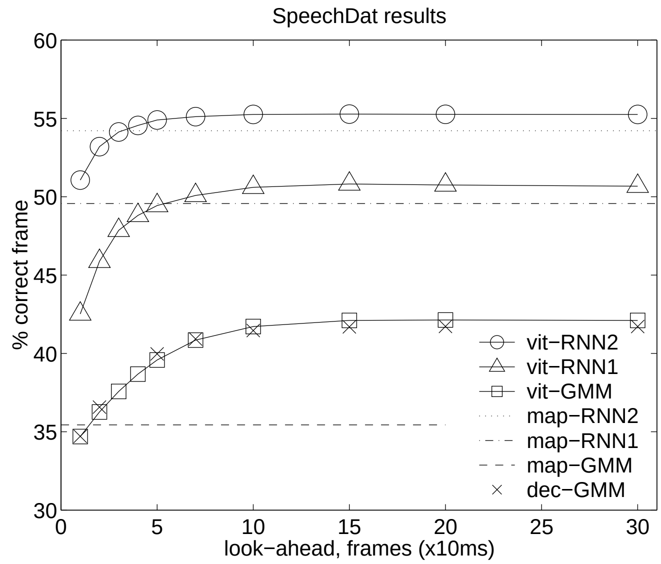

Fig. 3. Degradation in recognition accuracy due to truncation error in the Viterbi decoder. The x-axis shows the look ahead length in number of frames (10ms), while the y-axis the percentage of correct frames.

Acknowledgments

This research was funded by the Synface European project IST-2001-33327 and carried out at the Centre for Speech Technology supported by Vinnova (The Swedish Agency for Innovation Systems), KTH and participating Swedish companies and organizations.

References

[1] A. J. Viterbi, “Error bounds for convolutional codes and an asymtotically optimum decoding algorithm,” *IEEE Trans. Inform. Theory*, vol. IT-13, pp. 260–269, Apr. 1967.

[2] D. Kwan and S. Kallel, “A truncated best-path algorithm,” *IEEE Trans. Commun.*, vol. 46, no. 5, pp. 565–567, May 1998.

[3] A. D. Weathers, “An analysis of the truncated Viterby algorithm for PRML channels,” in *Proc. IEEE Intern. Conf. on Comm.*, vol. 3, 1999, pp. 1951–1956.

[4] T. Imai, A. Kobayashi, S. Sato, H. Tanaka, and A. Ando, “Progressive 2-pass decoder for real-time broadcast news captioning,” in *ICASSP*, 2000, pp. 1937–1940.

[5] A. Ljolje, D. M. Hindle, M. D. Riley, and R. W. Sproat, “The AT&T LVCSR-2000 system,” in *Speech Transcription Workshop*. University of Maryland, May 2000.

[6] B. Granström, I. Karlsson, and K.-E. Spens, “Synface - a project presentation,” in *Fonetik 2002, TMH-QPSR*, vol. 44, 2002, pp. 93–96.

[7] A. J. Robinson, “An application of recurrent nets to phone probability estimation,” *IEEE Trans. Neural Networks*, vol. 5, no. 2, pp. 298–304, Mar. 1994.

[8] P. J. Werbos, “Backpropagation through time: what it does and how to do it,” *Proc. of the IEEE*, vol. 78, no. 10, pp. 1550–1560, Oct. 1990.

[9] H. Bourlard and N. Morgan, “Continuous speech recognition by connectionist statistical methods,” *IEEE Trans. Neural Networks*, vol. 4, no. 6, pp. 893–909, Nov. 1993.

[10] N. Ström, “Development of a recurrent time-delay neural net speech recognition system,” *TMH-QPSR*, no. 4, pp. 1–15, Oct.-Dec. 1992.

[11] B. Lindberg, F. T. Johansen, N. Warakagoda, G. Lehtinen, Z. Kačič, A. Žgank, K. Elenius, and G. Salvi, “A noise robust multilingual reference recogniser based on SpeechDat(II),” in *6th Intern. Conf. on Spoken Language Processing*, vol. III, 2000, pp. 370–373.

[12] S. Young, G. Evermann, D. Kershaw, G. Moore, J. J. Odell, D. Ollason, V. Valtchev, and P. P. C. Woodland, *The HTK Book, Version 3.2*. Cambridge University Engineering Department, 2002.

[13] N. Ström, “The NICO toolkit for artificial neural networks,” http://www.speech.kth.se/NICO.

[14] E. Pihl, “Bottlenecks in the Synface telephone,” Bachelor of Science th., Dept. of Speech, Music and Hearing, KTH, Stockholm, Sweden, 2003.