On A N-GRAM MODEL APPROACH FOR PACKET LOSS CONCEALMENT

Minkyu Lee, Imed Zitouni, Qiru Zhou
Bell Labs, Lucent Technologies,
600 Mountain Avenue, Murray Hill, NJ 07974, USA
{minkyul,qzhou}@research.bell-labs.com

Abstract
In this paper, we investigate the possibility of predicting lost packets for packet loss concealment using n-gram predictive models. Unlike the conventional repetition-based algorithms, the proposed algorithm is based on the Shannon game, which serves as a principle for predicting the speech parameters of lost packets using the previously received parameters. During training phase, we construct statistical backoff n-gram models. In test phase, the models are used to predict the speech parameters of lost packets. Experiments were performed on switchboard telephone speech database and the proposed algorithm is compared with the conventional repetition-based algorithm. The performance is evaluated in terms of the spectral distortion between the original and the predicted (or repeated) speech. The algorithm based on the backoff n-gram models reduces the spectral distortion by 8.7% over the conventional repetition-based algorithm for the first lost packet after receiving one. Further it maintains about 6.2% improvement up to six consecutive lost packets. In terms of perplexity of the predictive models, backoff n-gram approach outperforms the repetition-based algorithm by 8.65%, which is very close to the improvement rate obtained from the spectral distortion measurement.

1. INTRODUCTION
Networking environment is migrating toward a single converged IP network that delivers voice, video and data traffic. A critical component for successful convergence is the transmission of voice packets over IP network. IP network is originally designed for transmitting data traffic that consists of relatively large-sized packets and does not require reliable real-time delivery. Packets can be dropped if necessary in order to alleviate network congestion. Also, subsequent packets can be routed through different paths. As a result, each packet can undergo different transmission delay. These network characteristics are very difficult to predict, if not impossible. But they are perfectly fine for data transmission because dropped packets can be retransmitted, and delay jitter does not affect packet transmission very much.

Voice transmission, however, requires real-time and reliable delivery of smaller-sized packets. The receiving end needs to get a steady stream of voice packets for playout. When a voice packet is dropped, there is simply no time for retransmitting the dropped packet. Also when a voice packet takes a longer route and fails to arrive on time for playback, the voice packet is useless. In Voice over IP (VoIP), a voice packet is regarded as a lost packet whether the packet fails to arrive on time or it simply fail to be delivered at all. Such problems are found in all IP networks no matter how well managed or over-provisioned they may be; these problems are certainly not limited to the public Internet or mismanaged networks.

Various schemes are available to recover or conceal the effects of the lost packets. Without such efforts, even the best designed and managed IP networks would fail to deliver toll quality. Many practical VoIP systems rely on the receiver-based Packet-Loss Concealment (PLC) schemes [1]. They are classified into insertion based, interpolation based and regeneration based methods. Insertion-based PLC methods include silence insertion, noise insertion and packet repetition. Silence insertion fills the gap with silence. Although widely used, its performance is very poor because packet loss results in periods of silence, causing unpleasant clipped-speech distortion. Noise insertion produces slightly better voice quality and intelligibility than silence insertion. In packet repetition, the most recently received packet is used to replace lost packets. Packet repetition performs the best among insertion-based methods, but still results in audible distortions in the speech signal. Interpolation-based PLC methods, such as G.711 PLC [2], provide higher concealment performance at the expense of computation. Another interpolation-based method is time scale modification technique [3], which stretches the good speech frame across the gap to hide the lost packets. Regeneration based PLC methods are the most sophisticated but producing highest quality. Imbedded PLC algorithms in CELP-based speech codecs [4] such as G.723.1, G.728 and G.729 belong to this category.

All PLC algorithms described above run at the receiving end (or decoder). When the decoder finds out that the receiving buffer is empty, meaning that the next packets are either lost or getting delayed, it starts PLC processing using the previous received packet. This is based on the assumption that speech is quasi-stationary, i.e., the current missing packet will probably possess similar characteristics as the previous received packet.

In this paper, instead of relying on previously received packet, we attempt to predict the lost packet based on the fact that the entropy of a language in acoustic space is limited. We use statistical n-gram models, which are widely used in speech recognition field for language modeling. Based on the n-gram model, we propose a new predictive PLC algorithm. First, speech utterances are segmented into frames. For each frame, a set of speech parameter vector is extracted and the vectors from all training data are clustered using vector quantization. Each cluster is represented by its codebook index, or kernel. In training phase, statistical n-gram predictive models are constructed based on the sequence of kernels. In testing phase, when there is a lost packet, the kernel (or index) of the lost packet is predicted using the n previous received packets. The performance of the proposed method is compared with the most common approach in PLC, which is to repeat the previous kernel (we denote this as repetition-based approach).

2. N-GRAM PREDICTIVE MODELS

The general idea of predicting the coming event was proposed by several researchers. C.E. Shannon [5] describes an approach for estimating the entropy of a language: a person is asked to predict
first letter of a text by proposing successively some candidates within 27 possibilities (the 26 English alphabets and the character space), until success. Once the candidate correctly predicts the first letter, he is asked to predict the second one, and so on. He relates the statistics of the number of trials required to find the right answer (namely the frequency distribution of the rank of the correct answer), to upper and lower bounds of the entropy. Cover and King [6] show that the Shannon game can be generalized into a gambling approach. Jelinek [7] describes variants of the Shannon game that compare the prediction accuracy of a statistical model and a human being. O’Boyle et al. [8] use the Shannon game for assessing a given rank. In this paper, we apply these principles to the problem of predicting lost packets based on the n previous received packets for voice over IP applications.

Firstly, we describe the details of the construction of n-gram predictive models. The models estimate the conditional probability of a kernel s given a history h, denoted as $p(s|h)$, according to the n-gram method in:\n
$$P(s_i | h) = P(s_i | s_{i-n+1}^{i-1}) .$$

(1)

We denote this approach a frequency n-gram model. Using the event frequency in the training data only, the conditional probability $P(s_i | s_{i-n+1}^{i-1})$ is estimated as follows:

$$P(s_i | s_{i-n+1}^{i-1}) = \frac{N(s_i^{i-1})}{N(s_{i-n+1}^{i-1})} ,$$

(2)

where $N(.)$ denotes the frequency of the argument in the training data. Using this approach, however, the probability of an event that does not appear in the training data $N(s_{i-n+1})$ is equal to zero. Since the training corpus cannot be large enough to represent the complete behavior of the source that emits packets, the estimation of unseen event probabilities using this method cannot be accurate. In fact, sparseness of data is a generic problem of frequency statistics. A new approach to estimate the probability of unseen events in the training data is to use the backoff method [9]. The main idea is to discount unreliable probability estimates obtained from the observed frequency and “redistribute” the “freed” probability among n-grams which never occurred in the training corpus. Using this approach, the probability of an unseen n-gram, $s_{i-n+1}^{i-1}$, is estimated according to a more general context, which is the $(n-1)$-gram, $s_{i-n+2}^{i-1}$:

$$P(s_i | s_{i-n+1}^{i-1}) = \begin{cases} \tilde{P}(s_i | s_{i-n+1}^{i-1}) & \text{if } N(s_{i-n+1}^{i-1}) > 0, \\ \alpha(s_{i-n+1}^{i-1})P(s_i | s_{i-n+2}^{i-1}) & \text{otherwise} \end{cases}$$

(3)

where $\alpha(.)$ is a normalizing constant [9], and $\tilde{P}(.)$ is estimated as follows:

$$\tilde{P}(s_i | s_{i-n+1}^{i-1}) = d N(s_i^{i-1}) N(s_{i-n+1}^{i-1}) .$$

(4)

The term $d_{s_i}$ denotes the Turing’s discount coefficient [9]. The normalizing constant $\alpha$ is derived according to the following equation:

$$\alpha(s_{i-n+1}^{i-1}) = \frac{1 - \sum_{s_i: N(s_{i-n+1}^{i-1})>0} \tilde{P}(s_i | s_{i-n+1}^{i-1})}{1 - \sum_{s_i: N(s_{i-n+1}^{i-1})>0} P(s_i | s_{i-n+2}^{i-1})} .$$

(5)

Backoff n-gram models shall leave intact the estimate count for the probability of all unseen n-grams. It shall also not discount high values of counts $r>k$, considering them to be reliable. To achieve this, the discount coefficient $d_r$ is set to one for $r>k$ and

$$d_r = \frac{r^*}{r - (k + 1)n_{k+1}/n_i} \quad \text{for } 1 \leq r \leq k ,$$

(6)

where $r^* = (r + 1)n_{r+1}/n_r$. The term $n_r$ denotes the number of n-grams, which occur exactly r times in the training set. As for the parameter $k$ in practice, a value close to $k=10$ is a good empirical choice.

3. STATISTICAL MODEL ACCURACY

Having built the statistical n-gram predictive models from a training corpus, one may ask how well these models will perform. This can be answered based on the concept of source of information in information theory [10]. To provide such a measure of performance, several concepts including entropy, estimated entropy, and perplexity have been introduced [5].

Consider an information source that puts out a sequence of symbols $s_1, s_2, ..., s_n$, each of which is chosen from a symbol set K with size $|K|$, according to a certain stochastic law. Using n-gram models, if the source is ergodic (i.e., its statistical properties can be completely characterized in a sufficiently long sequence that the source puts out), we often compute $H_p$ based on a finite set but sufficiently large $Q$:

$$H_p = -\frac{1}{Q} \sum_{i=1}^{Q} \log P(s_i | s_{i-1}, s_{i-2}, ..., s_{i-n+1}) .$$

(7)

An interesting interpretation of $H_p$ is that it is the degree of difficulty that the predictive model encounters, on average, when it is to predict a symbol from the same source [11].

The quantity $H_p$ is estimated entropy as it is calculated from a sufficiently long sequence based on n-gram models. Associated with $H_p$ is a quantity called perplexity (often called the average symbol branching factor of the predictive model) defined as $pp=2^{H_p}$. Another way to view perplexity is to consider it as the average number of possible symbols following any sequence of n-1 symbols. Perplexity is an important parameter in specifying the degree of sophistication in a prediction task, from the source uncertainty to the quality of the predictive model. Consequently, the ideal predictive model will have a perplexity equal to one. This means that the model is almost always able to predict the lost symbol exactly.

4. DATABASE

The switchboard telephone speech (SWB) corpus is used to evaluate the performance of the n-gram method. The training corpus is roughly 25 hours of speech. Speech analysis is performed for each 20 msec speech frame, to estimate 10th-order lsf parameters. In total, for training, we have collected about 4 million sets of lsf parameters from the SWB corpus. Then, the training parameter vector set is
quantized into a finite number of clusters by vector quantization (VQ) [11]. Each cluster is represented by a kernel or codebook index. Kernels are stored for further processing to build n-gram models. We use a codebook of 512 kernels.

For testing, a number of consecutive packet loss are inserted into the test utterances from the switchboard corpus. They are located randomly within active speech regions. There are a total of 888 locations where up to six consecutive packets are lost. The lsf parameters of the previous received packets are quantized using the same codebook obtained during the training phase. The resulting kernel history is inserted into the n-gram predictive models in order to predict the kernel for the lsf parameters of the lost packet.

5. EXPERIMENTAL RESULTS

Backoff bigram (n=2) and trigram (n=3) predictive models are constructed using the set of kernels, which are visualized in Fig. 1 and 2, respectively. Not surprisingly, the backoff bigram model tells that given a previous kernel \( s \) at time \( t-1 \), the best prediction for the kernel at time \( t \) is to repeat the previous one, i.e., \( s \). This interpretation is consistent with the popular repetition-based packet loss concealment scheme. Due to the quasi-stationary characteristics of human speech, it is probably the best strategy if only the previous packet is to be used for prediction.

In the backoff trigram model, repeating the previous kernel is still a good strategy. The flat plane in Fig. 2 represents cases of repeating the previous kernel. However, there are many other points in the figure that are off the plane. Those points represent cases where the past two kernels lead to a new kernel. We believe that this happens especially when speech is in transient region. We believe the proposed predictive algorithm will outperform the repetition-based algorithm due to the off-plane points in Fig. 2.

Next, we compare the performance of the backoff trigram predictive model with that of the conventional repetition-based PLC scheme and the frequency trigram model. The performance is measured in terms of spectral distortion between the original and synthetic (either by prediction or repetition) parameters. Fig. 3 shows the average Euclidian distortion for the three different methods: distances (1) between the true kernel and the kernel predicted by backoff trigram, (2) between the true kernel and the kernel predicted by frequency trigram and (3) between the true kernel and repeated kernel. Average distortions are calculated for each group, which is categorized based on the status of the previous packet. Group 1 represents instances when the previous packet is received but the current packet is lost. Likewise, group \( i \) is for the cases when \( i \) consecutive packets are lost after a packet is received.

Fig. 3 shows that the backoff-based trigram predictive model consistently outperforms the repetition-based method. The frequency-based trigram model slightly increases the distortion but still it is better than the repetition-based method. Table 1 illustrates the percent improvement over the repetition-based method. For example, the backoff trigram model shows more than 8.69% improvement over the repetition-based method in terms of spectral distortion measure for the first lost packet (i.e., about 0.19 for the backoff trigram model vs. about 0.21 for the repetition-based model in Fig. 3 Group 1). For the cases for a single packet loss (group 1), the frequency trigram model performs better than the backoff trigram model. But, for the instances of two or more consecutive losses (group 2 through group 6), the backoff trigram model shows slightly better performance over the frequency trigram model. On the average, the percent improvements of the backoff and frequency trigram models over the repetition-based algorithm are 6.2% and 5.9%, respectively.

Another method for performance evaluation is to use perplexity measurement. Perplexity is commonly used in speech recognition to evaluate the performance of n-gram models. A low perplexity measurement means that the model will have a better
prediction performance. Since the repetition-based model is not a statistical approach, one cannot calculate perplexity. However, since it has similar behavior to the backoff bigram model (cf. Fig. 1), we estimate its perplexity to be equal to that of the backoff bigram model. From the test data, the perplexity of the backoff bigram model is 85.69. The backoff trigram model outperforms the bigram model (and consequently the repetition-based model) by 8.65%. Interestingly, this improvement rate is similar to the rate calculated using distortion measure. It is important to notice that when computing the perplexity, the true previous kernels are to be used to estimate the probability $H_p$ in (7). Hence, the perplexity values shown above are for the cases of single packet losses (i.e., group 1 in Fig. 3 or the first row of Table 1).

Table 1: Improvement rate (%) of the two predictive models over repetition-based method

<table>
<thead>
<tr>
<th>Group</th>
<th>Frequency Trigram</th>
<th>Backoff Trigram</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>9.2123</td>
<td>8.6990</td>
</tr>
<tr>
<td>2</td>
<td>10.1906</td>
<td>10.4024</td>
</tr>
<tr>
<td>3</td>
<td>5.4356</td>
<td>5.9497</td>
</tr>
<tr>
<td>4</td>
<td>4.3394</td>
<td>4.8861</td>
</tr>
<tr>
<td>5</td>
<td>3.6785</td>
<td>4.0251</td>
</tr>
<tr>
<td>6</td>
<td>2.8340</td>
<td>3.2404</td>
</tr>
</tbody>
</table>

6. DISCUSSION AND FUTURE WORK

Statistical n-gram models are commonly used in automatic speech recognition systems. They are able to estimate the likelihood (probability) of an event ($n$ words or $n$ phonemes) in the language. In this paper, our goal is to estimate the likelihood of successive kernels, which may lead to a good prediction of the lost packet. However, it is important to notice that a kernel represents 20 msec of speech. Using the trigram model, only 60 msec can be handled, which is often smaller than the duration of one phoneme. In order to improve the performance of the predictive approach, we can expand the idea to use larger history, such as 4-gram, 5-gram, etc. An interesting approach extending this idea is to use phrase $n$-gram predictive models. For this approach, we first tag kernels into kernel phrases. Then, we build $n$-gram models on these phrases. Kernel phrases can be extracted automatically using mutual information technique. This allows us to handle large variable length history, which may lead to a better prediction [12].

Another approach is to predict the kernel using a function of candidate kernels. Let's denote $\gamma$ the perplexity of the statistical predictive model. According to the interpretation of perplexity, $\gamma$ is the average number of possible kernels that may follow the previous kernels in the history. One possible approach is to replace the lost kernel with one built from the $\gamma$ best candidate kernels. For example, we may consider the lost kernel as the mean of the $\gamma$ best candidates. Another possibility will be to build a tree, where leafs represent the lsf parameters and nodes represent clusters. A kernel models each cluster. Each cluster contains all the kernels of descendent nodes. The kernel is the mean of all lsf parameter vectors that belong to its cluster. Then, the predicted kernel will be the one that includes the $\gamma$ best candidates.

7. CONCLUSION

In this paper, we presented a new algorithm for packet loss concealment. The algorithm is based on statistical backoff $n$-gram models. The lsf parameter codebook index of the lost packet is predicted using the $n$-gram predictive approach. The simulation result shows that the proposed methods can predict lsf parameters that are closer to the original than simply repeating the previous parameters. Using switchboard telephone speech database for evaluation, the backoff trigram model shows more than 8.7% improvement over the repetition-based method in terms of spectral distortion measure for the first lost packet. Further the backoff trigram model maintains about 6.2% improvement up to six consecutive lost packets over the repetition-based method. We also used the perplexity to evaluate the performance of these methods. Our approach outperforms the repetition-based method by more than 8.65% (i.e., 85.69 for the backoff trigram model vs. 93.80 for the repetition-based model).

8. REFERENCES