UNCERTAINTY DECODING WITH SPLICE FOR NOISE ROBUST SPEECH RECOGNITION

Jasha Droppo, Alex Acero, and Li Deng

Microsoft Research, One Microsoft Way, Redmond, Washington, USA

ABSTRACT

Speech recognition front end noise removal algorithms have, in the past, estimated clean speech features from corrupted speech features. The accuracy of the noise removal process varies from frame to frame, and from dimension to dimension in the feature stream, due in part to the instantaneous SR of the input. In this paper, we show that localized knowledge of the accuracy of the noise removal process can be directly incorporated into the Gaussian evaluation within the decoder, to produce higher recognition accuracies. To prove this concept, we modify the SPLICE algorithm to output uncertainty information, and show that the combination of SPLICE with uncertainty decoding can remove 74.2% of the errors in a subset of the Aurora2 task.

1. INTRODUCTION

As soon as speech recognition systems moved out of pristine laboratory environments and into more mainstream use, it became clear that noise robustness would become a necessary component of any application. It is no longer safe to assume that speech input comes from a known microphone through a channel with high signal to noise ratio. Consequently, systems must be modified to deal with these harsher environments.

Research continues into both feature- and model-domain techniques to improve the robustness of speech recognition systems. It was shown in [1] that a feature-domain technique can achieve higher recognition accuracy than using matched noisy training and testing conditions. Since this matched condition is the limit that any model-domain technique strives for, we focus on techniques in the feature domain that allow us to beat the limit.

One general method for feature-domain cepstral de-noising is to design a module that pre-processes cepstra before they are fed into the speech recognition system. This includes parametric feature space transformations [2, 3], spectral subtraction, vector Taylor series, CDCN, stereo piecewise linear compensation for environment (SPLICE) [4], and cepstral smoothing techniques such as RASTA and CMN. The advantage of all of these techniques is that they can be seamlessly integrated into existing systems, without a complete overhaul of existing code.

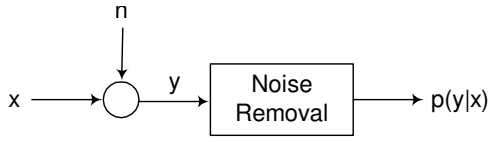

Fig. 1. Generic Noise Removal Framework

Many of these algorithms can be simply modified to produce estimates of the uncertainty of the noise removal process in addition to the cleaned features. Figure 1 represents the proposed system. Noise $\mathbf{n}$ corrupts the clean speech signal $\mathbf{x}$, producing the noisy signal $\mathbf{y}$. This is followed by the noise removal algorithm, which outputs a conditional PDF $p(\mathbf{y}|\mathbf{x})$.

This paper describes how the uncertainty from the noise removal system can be directly integrated into the decoding process. The uncertainty is generated by augmenting the SPLICE algorithm, which recently had the best noise removal performance at a special Aurora session in Eurospeech 2001 [4]. Uncertainty decoding improves the performance even further.

Our work generalizes the missing feature techniques in [5], and integrates more easily into existing decoders. It has a continuous, soft weighting of data rather than discarding regions below a set threshold.

In Section 2, we describe how the Gaussian evaluation in the decoder is modified to incorporate the new information from the front end. Section 3 contains a brief overview of SPLICE and a description of how the uncertainty estimation can be integrated into the noise removal process. In Section 4, we provide experimental results that show how uncertainty decoding improves recognition accuracy in the Aurora task.

2. UNCERTAINTY DECODING

Here we present a framework whereby estimates of the error, or uncertainty, associated with noise removal can be incorporated into the recognition process.

2.1. Uncertainty Modifies Gaussian Evaluation

At the heart of the speech recognition engine, many Gaussian mixture components are evaluated.

When recognizing uncorrupted speech cepstra, the purpose of these evaluations is to discover the probability of each clean observation vector, conditioned on the mixture index, $p_{\mathbf{x}|m}(\mathbf{x}|m)$ for the individual Gaussian in the speech model used by the recognizer.

If the training and testing conditions do not match, as is the case in noise-corrupted speech recognition, one option is to ignore the imperfections of the noise removal, and evaluate $p_{\mathbf{x}|m}(\hat{\mathbf{x}}|m)$. This is the classic case of passing the output of the noise removal algorithm directly to the recognizer.

A more rigorous approach, uncertainty decoding, is to generate the joint conditional PDF $p(\mathbf{x}, \mathbf{y}|m)$ and marginalize over all possible unseen clean speech cepstra:

$$p(\mathbf{y}|m) \quad = \quad \int_{-\infty}^{\infty} p(\mathbf{y}, \mathbf{x}|m)\, d\mathbf{x}$$

Under this framework, instead of just providing cleaned cepstra, the noise removal process estimates the conditional distribution $p(\mathbf{y}|\mathbf{x}, m)$, as a function of $\mathbf{x}$. For ease of implementation, we assume[1] that $p(\mathbf{y}|\mathbf{x})$ is independent of m:

$$p(\mathbf{y}|\mathbf{x}, m) \approx p(\mathbf{y}|\mathbf{x}) = \alpha N(\mathbf{x}; \hat{\mathbf{x}}, \sigma^2_{\hat{\mathbf{x}}})$$

Finally, the probability for the observation $\mathbf{y}$, conditioned on each acoustic model Gaussian mixture component m, can be calculated.

$$\begin{aligned} p(\mathbf{y}|m) &= \int_{-\infty}^{\infty} p(\mathbf{y}|\mathbf{x}, m)p(\mathbf{x}|m)\, d\mathbf{x} \\ &= \alpha \int_{-\infty}^{\infty} N(\hat{\mathbf{x}}; \mathbf{x}, \sigma^2_{\hat{\mathbf{x}}}) N(\mathbf{x}; \mu_m, \sigma^2_m)\, d\mathbf{x} \\ &= \alpha N(\hat{\mathbf{x}}; \mu_m, \sigma^2_m + \sigma^2_{\hat{\mathbf{x}}}) \end{aligned} \quad (1)$$

This formula is evaluated for each Gaussian mixture component in the decoder, $p(\mathbf{x}|m) = N(\mathbf{x}, \mu_m, \sigma^2_m)$.

The uncertainty output from the front end increases the variance of the Gaussian mixture component, producing an effective smoothing in cases where the front end is uncertain of the true value of the cleaned cepstra.

2.2. Special Cases

Two special cases exist for uncertainty decoding. In the absence of uncertainty information from the noise removal process, we can either assume that there is no uncertainty or that there is complete uncertainty.

If there were no uncertainty, then $\sigma^2_{\hat{\mathbf{x}}} = 0$. The probability of the observation $\mathbf{y}$, for each acoustic model Gaussian mixture component m, simplifies to:

$$p(\mathbf{y}|m) = p(\hat{\mathbf{x}}|m) = N(\hat{\mathbf{x}}; \mu_m, \sigma^2_m). \quad (2)$$

This is the traditional method of passing features directly from the noise removal algorithm to the decoder.

If there were complete uncertainty of any of the cepstral coefficients, the corresponding $\sigma^2_{\hat{\mathbf{x}}}$ would approach infinity. That coefficient would have no effect on the calculation of $p(\mathbf{y}|m)$. This is desirable behavior, under the assumption that the coefficient could not contribute to discrimination.

Both of these extreme cases are similar to the computations performed when using hard thresholds with missing feature techniques [5]. There has been some success in incorporating heuristic soft thresholds with missing feature techniques[6], but we believe that uncertainty decoding benefits from a rigorous probabilistic framework.

3. SPLICE NOISE REMOVAL AND UNCERTAINTY

SPLICE [4] is an algorithm that learns a probabilistic model of distortion from a clean cepstral vector, $\mathbf{x}$, into a noisy one, $\mathbf{y}$. Using this model, SPLICE can produce an approximation of the PDF $p(\mathbf{y}|\mathbf{x})$ for any distorted input $\mathbf{y}$.

[1]The mixture index m effectively partitions the acoustic space and a complete treatment would include this effect in the calculations.

3.1. A Model of Speech and its Degradation

SPLICE makes two fundamental assumptions about the form of the joint probability of $\mathbf{x}$ and $\mathbf{y}$. The first assumption is that the noisy speech cepstral vector follows the distribution of mixture of Gaussians:

$$p(\mathbf{y}) = \sum_s N(\mathbf{y}; \mu_s, \sigma_s)p(s)$$

One distribution $p(\mathbf{y})$ is trained for each separate distortion condition (not indexed for clarity), and can be thought as a "codebook" with a total of N codewords (means) and their variances. Each codebook is implicitly conditioned on a specific noise type and level. To select the appropriate codebook at runtime, we developed an effective on-line environmental selection method, which has been described in detail in [7].

The second assumption made by the SPLICE is that the conditional probability density function (PDF) for the clean vector $\mathbf{x}$ given the noisy speech vector, $\mathbf{y}$, and the region index, s, is Gaussian whose mean vector is a linear transformation of the noisy speech vector $\mathbf{y}$. In this paper, we take a simplified form of this linear transformation by making the rotation matrix to be the identity matrix, leaving only the bias or correction vector. Thus, the conditional PDF is assumed to have the form,

$$p(\mathbf{x}|\mathbf{y}, s) = N(\mathbf{x}; \mathbf{y} + \mathbf{r}_s, \Gamma_s). \quad (3)$$

3.2. SPLICE Training

Since the noisy speech PDF $p(\mathbf{y})$ is assumed to be a mixture of Gaussians, the standard EM algorithm can be used to train μ_s and σ_s on noisy speech. Initial values of the parameters are determined by a VQ clustering algorithm.

If stereo data is available, the parameters $\mathbf{r}_s$ and Γ_s of the conditional PDF $p(\mathbf{x}|\mathbf{y}, s)$ can be trained using the maximum likelihood criterion:

$$\mathbf{r}_s = \frac{\sum_n p(s|\mathbf{y}_n)(\mathbf{x}_n - \mathbf{y}_n)}{\sum_n p(s|\mathbf{y}_n)} \quad (4)$$

$$\mathbf{\Gamma}_s = \frac{\sum_n p(s|\mathbf{y}_n)(\mathbf{x}_n - \mathbf{y}_n)^2}{\sum_n p(s|\mathbf{y}_n)} - \mathbf{r}_s^2 \quad (5)$$

$$p(s|\mathbf{y}_n) = \frac{p(\mathbf{y}_n|s)p(s)}{\sum_s p(\mathbf{y}_n|s)p(s)} \quad (6)$$

This training procedure requires a set of stereo (two channel) data. One channel contains the clean utterance, and the other contains the same utterance with distortion. The two-channel data can be collected, for example, by simultaneously recording on one close-talk and one far-field microphone.

For the Aurora work reported in this paper, the SPLICE parameters were trained using identical utterances from the clean training set and the multi-style training set.

3.3. Complete SPLICE

In the past, SPLICE has been applied to the 13-dimensional static cepstral coefficients only, ignoring the fact that delta and acceleration coefficients are also central to the recognition process. Since SPLICE processes each frame independently, one could argue that the SPLICE mapping is incomplete. The delta and acceleration features computed on-line during recognition, from these cleaned static features, are not optimal.

Alternatively, the static feature vector can be completed with delta and acceleration components before passing the vector to SPLICE for processing. Under this improved scenario, SPLICE maps a 39-dimensional noisy input vector to a 39-dimensional cleaned speech output vector. The advantage of this approach is that it is consistent across the entire vector being modeled by the recognizer. Of course, the delta and acceleration parameters no longer correspond to a linear filtering of the static parameters.

3.4. Estimating $p(\mathbf{y}|\mathbf{x})$

For uncertainty decoding, the front end must provide an estimate of the conditional probability density function

$$p(\mathbf{y}|\mathbf{x}) \quad = \quad \frac{\sum_s p(\mathbf{x}|\mathbf{y},s)p(\mathbf{y}|s)p(s)}{p(\mathbf{x})}.$$

as a function of $\mathbf{x}$. We do this by leveraging the probabilistic framework of SPLICE.

Each term of the numerator is directly computable from the SPLICE parameters. It is somewhat more difficult to derive the prior $p(\mathbf{x})$.

First, note that the joint conditional probability $p(\mathbf{x},\mathbf{y}|s)$ can be re-written as,

$$\begin{aligned} p(\mathbf{x},\mathbf{y}|s) &= p(\mathbf{x}|\mathbf{y},s)p(\mathbf{y}|s) \\ &= N(\mathbf{x};\mathbf{y}+\mathbf{r}_s,\Gamma_s^2)N(\mathbf{y};\mu_s,\sigma_s^2) \\ &= N(\mathbf{y};\frac{\sigma_s^2(\mathbf{x}-\mathbf{r}_s)+\Gamma_s^2\mu_s}{\Gamma_s^2+\sigma_s^2},\frac{\Gamma_s^2\sigma_s^2}{\Gamma_s^2+\sigma_s^2}) \\ & \quad N(\mathbf{x};\mu_s+\mathbf{r}_s,\Gamma_s^2+\sigma_s^2) \end{aligned}$$

So the prior for $\mathbf{x}$ is simply,

$$\begin{aligned} p(\mathbf{x}) &= \sum_s p(\mathbf{x}|s)p(s) \\ &= \sum_s \int_{-\infty}^{\infty} p(\mathbf{x},\mathbf{y}|s)p(s)\,d\mathbf{y} \\ &= \sum_s N(\mathbf{x};\mu_s+\mathbf{r}_s,\Gamma_s^2+\sigma_s^2)p(s) \end{aligned}$$

In order to simplify $p(\mathbf{y}|\mathbf{x})$ for use in Eq. 1, we approximate this mixture of Gaussians as a single Gaussian,

$$\begin{aligned} p(\mathbf{x}) &\approx N(\mathbf{x};\mu_\mathbf{x},\sigma_\mathbf{x}^2), \text{ where} \\ \mu_\mathbf{x} &= \sum_s (\mathbf{r}_s+\mu_s)\,p(s), \text{ and} \\ \sigma_x^2 &= \sum_s \left((\mathbf{r}_s+\mu_s)^2+\sigma_s^2+\Gamma_s^2\right)p(s)-\mu_\mathbf{x}^2. \end{aligned}$$

We then use this approximation to simplify $p(\mathbf{y}|\mathbf{x})$.

$$\begin{aligned} p(\mathbf{y}|\mathbf{x}) &= \frac{\sum_s p(\mathbf{x}|\mathbf{y},s)p(\mathbf{y}|s)p(s)}{p(\mathbf{x})} \\ &= \frac{\sum_s N(\mathbf{x};\mathbf{y}+\mathbf{r}_s,\Gamma_s^2)p(\mathbf{y}|s)p(s)}{N(\mathbf{x};\mu_\mathbf{x},\sigma_\mathbf{x}^2)} \\ &= \sum_s N(\mathbf{x};\hat{\mathbf{x}}_s,\sigma_{\hat{\mathbf{x}}_s}^2)p(\mathbf{y}|s)p(s), \text{ where} \qquad (7) \\ \hat{\mathbf{x}}_s &= \frac{\sigma_\mathbf{x}^2(\mathbf{y}+\mathbf{r}_s)-\Gamma_s^2\mu_\mathbf{x}}{\sigma_\mathbf{x}^2-\Gamma_s^2}, \text{ and} \qquad (8) \\ \sigma_{\hat{\mathbf{x}}_s}^2 &= \frac{\sigma_\mathbf{x}^2\Gamma_s^2}{\sigma_\mathbf{x}^2-\Gamma_s^2}. \qquad (9) \end{aligned}$$

Of course this derivation only makes sense when

$$\sigma_\mathbf{x}^2 \geq \Gamma_s^2. \qquad (10)$$

Recall that $\sigma_\mathbf{x}^2$ is the global variance of the clean speech prior, and that Γ_s^2 is the expected squared error of the noise removal process. There are two cases where we might expect Eq. 10 to be violated. Either the noise removal process is fundamentally flawed, and expects itself to be doing worse than outputting the prior mean for speech, or our approximation of $p(\mathbf{x})$ is causing mischief. In the handful cases that don't obey Eq. 10, we assume the latter to be the case, and simply force $\sigma_{\hat{\mathbf{x}}}^2 \geq \Gamma_s^2+\epsilon$, where $\epsilon = 0.1$. In practice, this occurs on less than 5% of cepstral coefficients.

Ideally, the conditional distribution we seek would be given by the sum in Eq. 7, but the fast implementation we use the assumption that $p(\mathbf{y}|s)p(s)$ is zero for all but one value of s.

$$\hat{p}(\mathbf{y}|s)p(s) \cong \begin{cases} 1 & s = \text{argmax}_s p(\mathbf{y}|s)p(s) \\ 0 & \text{otherwise} \end{cases}$$

SPLICE processing then consists of two sequential operations. First, finding optimal codeword s using the VQ codebook based on the parameters (μ_s,σ_s), and then finding $\hat{\mathbf{x}}$ and $\sigma_{\hat{\mathbf{x}}}^2$ according to Eq. 8 and Eq. 9.

4. RESULTS

4.1. Qualitative

Figure 2 illustrates that SPLICE is producing reasonable outputs. The upper half of the figure shows c_0 as a function of time, for both uncorrupted and corrupted speech. The lower half of the figure shows the $\hat{\mathbf{x}}$ parameter output by SPLICE, as well as the range of variation ($\pm\sigma_{\hat{\mathbf{x}}}^2$).

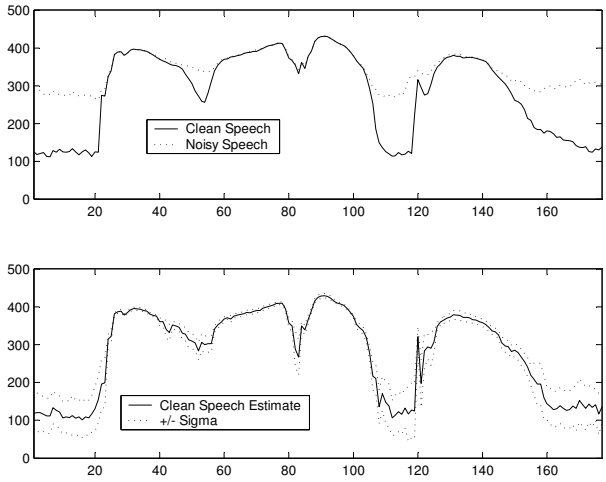

Fig. 2. C_0 for clean speech (top), together with $p(y|x)$ for a corresponding noisy utterance, plotted as a function of x.

The cepstral coefficient c_0 is roughly related to frame energy. In speech regions, c_0 is large and masks the contribution of noise. The value of $\hat{\mathbf{x}}$ is accurate, and the margin of error is small. In non-speech regions, c_0 from the noise masks the speech value. The value of $\hat{\mathbf{x}}$ is less accurate, and is accompanied by a larger variance. We observe that the c_0 for clean speech is consistent with the bounds described by the dashed lines in the lower half of Figure 2.

4.2. Quantitative

Several connected digit experiments were run using the framework provided in the Aurora2[8] corpus. The acoustic model training data consists of 8440 clean utterances and the same utterances in groups of 422, corrupted 20 different ways. These 20 sets consist of four noise types (subway, babble, car, and exhibition) artificially added at five different signal to noise ratios (infinite, 20, 15, 10, and 5). All of our tests were performed with an acoustic model trained on clean data with scripts provided with the Aurora2 framework.

Only results on set A, which consists of 28028 files are reported in this paper. This set is partitioned in to the same noise types found in the training data, artificially added at seven different signal to noise ratios (infinite, 20, 15, 10, 5, 0, and -5).

SPLICE, as represented in this paper, does not generalize well to unseen noise types (contained in set B), or unseen convolutional channels (contained in set C), although it has been shown that both of these limitations can be easily overcome [4] with noise mean normalization (NMN) and cepstral mean normalization (CMN). Neither technique is used in this paper; in theory the results on set A should be unchanged by their omission.

Table 1 shows digit recognition accuracy for eight different experiments, consisting of three different front ends, and three different decoding strategies.

Table 1. A comparison of the digit accuracy of eight front end configurations on Aurora test set A, with a clean acoustic model.

	Uncertainty Source		
Front End	None	SPLICE	True
Standard MFCC	63.66%	N/A	85.94%
SPLICE	87.22%	87.47%	89.52%
Complete SPLICE	88.21%	90.63%	92.81%

The front ends considered were a standard MFCC algorithm, SPLICE, and Complete SPLICE. The standard MFCC algorithm was identical to the Aurora reference in all respects but one: we modified it to use magnitude-squared spectra internally instead of magnitude spectra. SPLICE and Complete SPLICE post-process this data as described in Section 3.

We considered decoders without uncertainty, with uncertainty generated by SPLICE, and true uncertainty generated from oracle data. The SPLICE uncertainties were of course unavailable for the reference MFCC front end, and computed as described above for the other front ends. The true uncertainties were derived by computing the magnitude-squared error between the front-end output and the true clean speech cepstra, which are available in the Aurora2 data.

The experiments with true uncertainty are indicative of how much improvement we can expect by performing uncertainty decoding. For the standard front end, it is possible to eliminate over 60% of the word error rate just by adding perfect knowledge of the magnitude of the cepstral errors. For the front ends containing SPLICE, the possible improvement is smaller, but not negligible.

Estimating the uncertainty using SPLICE yields improvement in both cases. The best realizable system, the Complete SPLICE front end, including uncertainty decoding, reduces the word error rate by 74.2% with respect to the front end without SPLICE.

5. DISCUSSION

We have described a systematic process for incorporating uncertainty from the noise removal process into a speech recognition system.

Uncertainty decoding carries two major benefits. First, including the conditional probability $p(\mathbf{y}|\mathbf{x})$ effectively accounts for the residual corruption from the noise removal process. And second, because uncertainty decoding is based on a comprehensive probabilistic framework, we avoid any heuristic tuning.

It should be simple to modify most noise removal algorithms to produce uncertainty estimates. We have demonstrated this using SPLICE. Introducing uncertainty decoding reduced the average word error rate of our state of the art system by over 20% relative.

6. REFERENCES

[1] L. Deng, A. Acero, M. Plumpe, and X. D. Huang, "Large vocabulary speech recognition under adverse acoustic environments," in *Proc. 2000 ICSLP*, Beijing, China, October 2000, pp. 806–809.

[2] Y. Ephraim and M. Rahim, "On second-order statistics and linear estimation of cepstral coefficients," *IEEE Trans. Speech and Audio Proc.*, vol. 7, no. 2, pp. 162–176, March 1999.

[3] Y. Zhao, "Frequency-domain maximum likelihood estimation for automatic speech recognition in additive and convolutive noises," *IEEE Trans. Speech and Audio Proc.*, vol. 8, no. 3, pp. 255–266, May 2000.

[4] J. Droppo, L. Deng, and A. Acero, "Evaluation of the SPLICE algorithm on the Aurora2 database (web update)," in *Proc. Eurospeech 2001*, Aalborg, Denmark, September 2001, pp. 217–220.

[5] M. Cooke, P. Green, L. Josifovski, and A. Vizinho, "Robust automatic speech recognition with missing and unreliable acoustic data," *Speech Communication*, vol. 34, no. 3, pp. 267–285, June 2001.

[6] P. Green J. P. Barker, M. Cooke, "Robust ASR based on clean speech models: An evaluation of missing data techniques for connected digit recognition in noise," in *Proc. Eurospeech 2001*, Aalborg, Denmark, September 2001, pp. 213–216.

[7] J. Droppo, A. Acero, and L. Deng, "Efficient on-line acoustic environment estimation for FCDCN in a continuous speech recognition system," in *Int. Conf. On Acoustics, Speech and Signal Processing*, Salt Lake City, Utah, May 2001.

[8] H. G. Hirsch and D. Pearce, "The AURORA experimental framework for the performance evaluations of speech recognition systems under noisy condidions," in *ISCA ITRW ASR2000 "Automatic Speech Recognition: Challenges for the Next Millennium"*, Paris, France, September 2000.