

Robust Joint Huffman and Convolutional Decoding

Ksenija Lakovic, John Villasenor and Rick Wesel
 Electrical Engineering Department
 University of California, Los Angeles

Abstract: We introduce a joint decoding algorithm for a Huffman source code followed by a convolutional channel code. The algorithm incorporates a restriction to valid Huffman codewords into the convolutional code's trellis. Performance is improved by this restriction. Furthermore, soft Huffman decoding is implicitly included, and *a priori* probabilities of input symbols can be easily exploited, if available. The joint decoding algorithm demonstrates significantly improved robustness relative to a system in which source and channel decoding are performed separately.

1. Introduction

Source and channel coding are often performed independently in communication systems. In theory, this is justified by the separation theorem of Shannon [1], which states that the two-stage system achieves optimal performance for sufficiently large block lengths. Practical coding schemes, however, use finite and often short block lengths, thus creating an opportunity to perform joint source-channel coding. In this paper we consider the combination of Huffman source coding followed by convolutional channel coding. We introduce a joint decoding algorithm that incorporates information about the source coding, and that demonstrates significantly improved performance relative to a system that performs source and channel decoding separately.

Joint source channel coding has received increased attention in the literature recently. Previous work includes the theoretical bounds established by Hochwald and Zeger [2] and the work on soft decoding of Huffman codes [3]. The work of Demir and Sayood [4] involves joint consideration of Huffman and channel coding based on extended Huffman coding, and is best suited to small Huffman symbol alphabets. In contrast with [4], our approach is based on modified convolutional decoding, providing closer connection to general source and channel coding methods and involving less sensitivity to the source alphabet size.

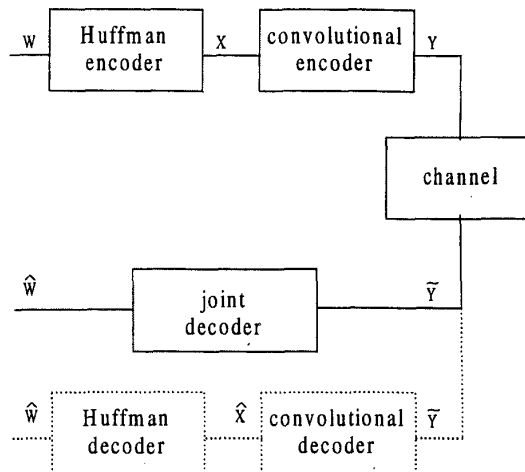


Figure 1: The proposed joint decoding system. The dotted lower branch represents a standard system that employs the channel and the source decoding separately.

2. Modified Trellis for Joint Huffman and Convolutional Soft Decoding

In a system employing separate source and channel decoding, the received bits are first subject to convolutional decoding, usually using the Viterbi algorithm [5]. The output of the channel decoder is then fed as a sequence of hard decision bits to a Huffman decoder that applies a state machine to produce a sequence of decoded symbols. By contrast, the joint decoding we introduce performs the channel decoding and Huffman decoding simultaneously. Soft Huffman decoding is implicitly included, which offers advantages over hard decision Huffman decoding [3].

Consider a two-stage coding system, illustrated in Figure 1, and let the source alphabet be denoted by $\mathcal{A} = \{a_0, a_1, \dots, a_N\}$, and the source coder alphabet by $\mathcal{B} = \{b_0, b_1, \dots, b_D\}$. Assume that an information sequence $W = (w_1, w_2, \dots)$, $w_i \in \mathcal{A}$, is Huffman encoded into $X = (x_1, x_2, \dots)$, $x_i \in \mathcal{B}$, which is further encoded for error-protection into $Y = (y_1, y_2, \dots)$, and that a sequence $\tilde{Y} = (\tilde{y}_1, \tilde{y}_2, \dots)$ is received at the output of the channel.

In order to perform *maximum likelihood decoding*, both the joint decoder and the standard convolutional decoder produce an estimate \hat{Y} of the codeword Y , based on the received sequence \tilde{Y} , by maximizing:

$$P(\tilde{Y}|Y) = \prod P(\tilde{y}_j|y_j)$$

or equivalently:

$$\log P(\tilde{Y}|Y) = \sum \log P(\tilde{y}_j|y_j)$$

The standard convolutional decoder maximizes $P(\tilde{Y}|Y)$ over all valid convolutional output sequences Y while the joint decoder restricts attention to the sequences Y produced by valid outputs of the Huffman source coder.

The log-likelihood function $\log P(\tilde{Y}|Y)$ is often called path metric $M(\tilde{Y}|Y)$, while the terms $\log P(\tilde{y}_j|y_j)$ are usually called branch metrics. Both for the standard convolutional trellis, and for the modified joint decoding trellis, we define the trellis states in the usual manner, by the outputs of the shift registers in the convolutional encoder, and denote the state S_m after j time units by $S_{m,j}$, with associated sequence $\hat{Y}_{m,j}$ and metric $M_{m,j}$.

In the standard convolutional trellis, state to state transitions are associated with symbols from the source coder alphabet \mathcal{B} (usually it is simply a binary alphabet, i.e. $\mathcal{B} = \{0,1\}$). Assuming that $c_{m',m}^{b_i}$ represents the sequence that corresponds to the transition from state $S_{m'}$ to state S_m for the input b_i , the decoding procedure at each step can be written as:

$$\begin{aligned} M_{m,j} &= \max_{m'} M_{m',j-1} + \log P(\tilde{y}_j | c_{m',m}^{b_i}) = \\ &= M_{m^*,j-1} + \log P(\tilde{y}_j | c_{m^*,m}^{b^*}) \\ \hat{Y}_{m,j} &= (\hat{Y}_{m^*,j-1}, c_{m^*,m}^{b^*}) \end{aligned}$$

These equations illustrate the well-known Viterbi decoding algorithm - computation of path metrics

for all the paths entering the state (by adding the branch metric entering that state to the metric of the connecting path at the previous time unit), after which the path with the largest metric is stored, together with its metric, and all other paths are pruned.

In the joint decoding procedure we introduce an alternative definition of trellis branches, and follow the same decoding strategy. Branches of the modified trellis are associated with symbols from the alphabet $\mathcal{A} = \{a_0, a_1, \dots, a_N\}$, created as collections of branches $c_{m',m}^{b_i}$ and can be labeled by $C_{m',m}^{a_i}$. They are determined by the Huffman code, i.e. if a sequence $(\beta_{1,1}, \beta_{1,2}, \dots, \beta_{1,L_1})$, $\beta_{1,k} \in \mathcal{B}$, represents a Huffman coded symbol a_i , then:

$$C_{m',m}^{a_i} = (c_{m',m_1}^{\beta_{1,1}}, c_{m_1,m_2}^{\beta_{1,2}}, \dots, c_{m_{L_1-1},m}^{\beta_{1,L_1}})$$

Therefore, the decoding procedure considers variable length state to state transitions, involving branch metrics and received bits not only from the preceding time unit. It can be described by the following equations:

$$\begin{aligned} M_{m,j} &= \max_{m',l} M_{m',j-L_1} + \log P(\tilde{Y}_{j-L_1+1}^j | C_{m',m}^{a_i}) = \\ &= M_{m^*,j} + \log P(\tilde{Y}_{j^*+1}^j | C_{m^*,m}^{a^*}) \end{aligned}$$

$$\hat{Y}_{m,j} = (\hat{Y}_{m^*,j^*}, C_{m^*,m}^{a^*})$$

where $\tilde{Y}_{j-L_1+1}^j$ represents an extended sequence of received symbols:

$$\tilde{Y}_{j-L_1+1}^j = (\tilde{y}_{j-L_1+1}, \tilde{y}_{j-L_1+2}, \dots, \tilde{y}_j).$$

At each step of the joint decoding process the sequence of decoded input symbols can be found directly, as $\hat{W}_{m,j} = (\hat{W}_{m^*,j^*}, a^*)$, while the standard convolutional decoding procedure obtains the sequence $\hat{X}_{m,j} = (\hat{X}_{m^*,j-1}, b^*)$, which requires additional processing by the Huffman decoder. Furthermore, the convolutional decoding procedure does not obtain $\hat{X}_{m,j}$ as a necessarily valid Huffman coded sequence.

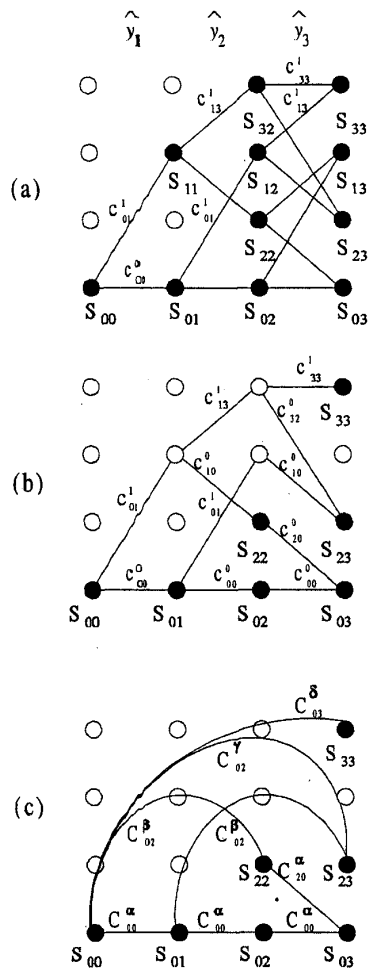


Figure 2: (a) Standard convolutional decoding trellis (b) Subset of trellis branches that correspond to valid bit sequences, which is used to construct variable length branches for the joint decoding (c) Variable length branches in the modified trellis

As an example consider the situation illustrated in Figure 2, where $A = \{\alpha, \beta, \gamma, \delta\}$, $B = \{0, 1\}$ and the Huffman code is:

$$\alpha : 0, \quad \beta : 10, \quad \gamma : 110, \quad \delta : 111.$$

The standard convolutional trellis diagram is shown in Figure 2(a). The branches that correspond to valid bit sequences after 3 time units are shown in Figure 2(b). They form the variable length branches:

$$\begin{aligned} C_{0,0}^\alpha &= c_{0,0}^0, & C_{0,2}^\beta &= (c_{0,1}^1, c_{1,2}^0), \\ C_{0,2}^\gamma &= (c_{0,1}^1, c_{1,3}^1, c_{3,2}^0), & C_{0,3}^\delta &= (c_{0,1}^1, c_{1,3}^1, c_{3,3}^1), \\ C_{2,0}^\alpha &= c_{2,0}^0 \end{aligned}$$

Those branches are illustrated in Figure 2(c) and they will be involved in the joint decoding procedure.

For instance, consider the decoding process at the state $S_{3,3}$. In the standard convolutional decoding procedure the possible associated sequences are:

$$\hat{Y}_{3,3} = \begin{cases} (\hat{Y}_{3,2}, c_{3,3}^1) = (\hat{Y}_{1,1}, c_{1,3}^1, c_{3,3}^1) = (c_{0,1}^1, c_{1,3}^1, c_{3,3}^1) \\ (\hat{Y}_{1,2}, c_{1,3}^1) = (\hat{Y}_{0,1}, c_{0,1}^1, c_{1,3}^1) = (c_{0,0}^0, c_{0,1}^1, c_{1,3}^1) \end{cases}$$

In the case that:

$$M_{1,2} + \log P(\hat{y}_3 | c_{1,3}^1) > M_{3,2} + \log P(\hat{y}_3 | c_{3,3}^1)$$

which is equivalent to:

$$\begin{aligned} \log P(\hat{y}_1 | c_{0,0}^0) + \log P(\hat{y}_2 | c_{0,1}^1) + \log P(\hat{y}_3 | c_{1,3}^1) > \\ > \log P(\hat{y}_1 | c_{0,1}^1) + \log P(\hat{y}_2 | c_{1,3}^1) + \log P(\hat{y}_3 | c_{3,3}^1) \end{aligned}$$

the output sequence is $\hat{X}_{3,3} = (0, 1, 1)$, which is an invalid input of the Huffman decoder, and directly results into a decoding error.

In the joint decoding procedure the only possible decoded sequence is, however:

$$\hat{Y}_{3,3} = (\hat{Y}_{0,0}, C_{0,3}^\delta) = C_{0,3}^\delta$$

that corresponds to $\hat{W}_{3,3} = (\hat{W}_{0,0}, \delta) = \delta$.

This example illustrates that the probability of decoding error is reduced by restricting consideration during decoding to the subset of transitions associated with valid bit sequences.

The joint decoding procedure, however, involves disadvantages due to increased decoding complexity. For a source alphabet of size N , a binary Huffman code, and a $(1, n, m)$ convolutional code, the standard convolutional algorithm involves the computation of 2^{m+1} branch metrics at each step, for the branches of length 1. Since the joint decoding procedure considers at most $2^m N$ variable length branches at each step, the complexity is increased.

In the case that *a priori* probabilities of input symbols are known, the optimum decoding procedure involves a *maximum a posteriori decoder*, that maximizes:

$$P(Y|\tilde{Y}) = \frac{P(\tilde{Y}|Y)P(Y)}{P(\tilde{Y})}$$

While for fixed length codes $P(\tilde{Y})$ is irrelevant to the decoders's operation, this is not necessarily true for the variable length case. In order to simplify the decoding procedure, however, the effect of $P(\tilde{Y})$ can be disregarded, and performance gain can still be achieved, as shown by Demir and Sayood [4].

In the general case, then, for the random sequence $\{y_i\}$, the equivalent maximization criterion becomes:

$$\log P(\tilde{Y}|Y)P(Y) = \sum \log P(\tilde{y}_i|y_i)P(y_i)$$

In our case, *maximum a posteriori* decoding can be obtained by including into the previously described procedure the probabilities of symbols from the original alphabet $a_i \in \mathcal{A}$ in the case of the joint decoding, or the probabilities of binary symbols from the Huffman coded sequence $b_i \in \mathcal{B}$, in the case of the two-stage decoding. Since the source statistics affect the symbols from the original alphabet directly, which is not the case for binary symbols, the consideration of *a priori* probabilities additionally improves the performance of a joint decoding algorithm, relative to an algorithm that performs source and channel decoding separately.

3. Simulation Results

Simulations were performed for a random source with an alphabet of size 5, a Huffman code with codewords $\{0,10,110,1110,1111\}$, and symbol probabilities ideally matched to codeword lengths, i.e. $\{1/2, 1/4, 1/8, 1/16, 1/16\}$. The convolutional code was a (2,1,2) code with the octal generator sequences $g^{(1)} = 5$ and $g^{(2)} = 7$. Blocks containing 100 symbols were Huffman coded, convolutionally encoded, and then corrupted by additive white Gaussian noise. Decoding was performed using the joint algorithm as well as with a traditional approach combining a hard-decision Viterbi decoder with a Huffman decoder.

There are a number of interesting criteria that can be used to assess the results. One possibility is to examine the number of blocks that contain the correct number of decoded symbols (100 symbols in the simulations described here). The fraction of such blocks was consistently higher (for example by 17 % at an SNR of 5 dB) for the joint decoding

algorithm. Another criterion concerns the probability that a block produced by the decoder has the correct number of symbols but contains one or more symbol errors (block decoding error). In Figure 3, these two criteria are combined i.e. a decoded block was considered in error if it contained the wrong number of symbols, or if it contained the correct number of symbols but with one or more symbol errors. It illustrates that the joint algorithm offers superior performance for all values of SNR, with a larger degree of improvement for higher SNR.

Additional simulations were performed for the random source with symbol probabilities $\{0.9, 0.09, 0.009, 0.0009, 0.0001\}$. The Huffman code was the same. One set of simulations included *a priori* probabilities both in the joint decoding procedure, and in the standard two-stage procedure, while the other one was based on maximum likelihood decoding. The results are shown in Figure 4. Consideration of *a priori* information improves both the joint and the standard method, particularly for low values of SNR. However, for all values of SNR, the level of improvement is higher for the joint method.

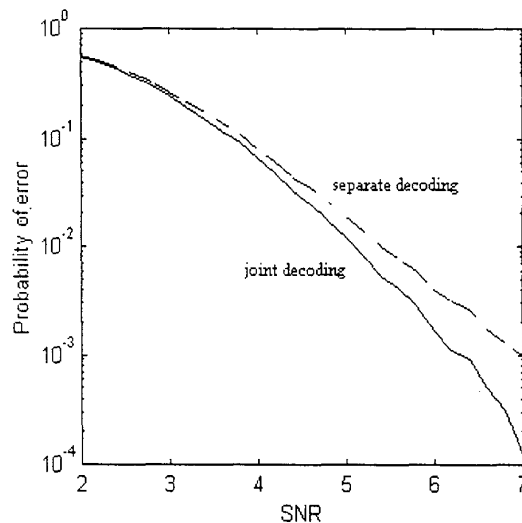


Figure 3: Probability of error as a function of the signal to noise ratio. This figure shows the performance improvement offered by the joint source/channel decoding algorithm over separate decoding. In generating this figure, a decoded block was considered in error if it contained the wrong number of symbols, or if it contained the correct number of symbols but with one or more symbol errors.

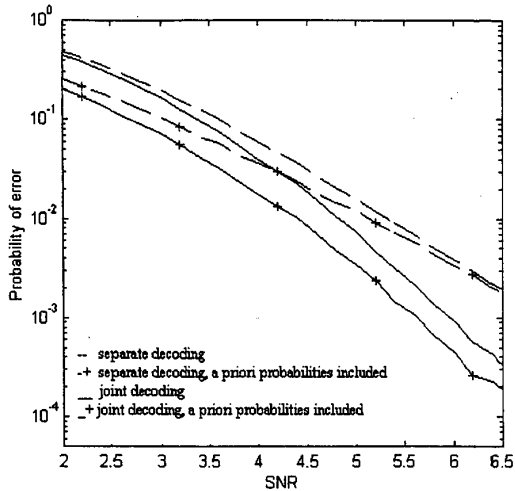


Figure 4: Probability of error as a function of the signal to noise ratio, for a non-uniform source

4. Conclusions

We have presented an algorithm for jointly decoding a Huffman and convolutionally encoded data sequence. This algorithm relies on a modified channel decoding trellis that incorporates information about the structure of the Huffman source code. Simulation results show significant improvement in error rates relative to a traditional system in which source and channel decoding are performed separately.

5. References

- [1] C.E. Shannon "A mathematical theory of communication," *Bell Syst. Technol.* Vol. 27, 1948.
- [2] B.Hochward, K. Zeger "Tradeoff between source and channel coding," *IEEE Transactions on Information Theory*, vol. 43, no.5, pp. 1412-24, Sept. 1997.
- [3] J. Wen, J.D. Villasenor "Utilizing Soft Information in Decoding of Variable Length Codes," *Proc. 1999 IEEE Data Compression Conference*, Snowbird, UT, March 1999.
- [4] N. Demir, K. Sayood, "Joint source/channel coding for variable length codes," *Proc. 1998 IEEE Data Compression Conference*, Snowbird, UT, March 1998.
- [5.] G.D. Forney "The Viterbi Algorithm," *Proceedings of IEEE*, vol.61, no.3, March 1973.