A Unified Theory of Density Models and Auto-Encoders*

S P Luttrell
Defence Evaluation and Research Agency
St Andrews Road, Malvern
Worcs, WR14 3PS, United Kingdom
email: luttrell@signal.dera.gov.uk
tel: +44 (0) 1684 894046
fax: +44 (0) 1684 894384

31 October 1997

Abstract

This report introduces an objective function for simultaneously optimising the density model and transition matrices of a Markov source. The chosen objective function seeks to minimise the average total number of bits that is required to encode the joint state of the Markov source. This may be applied to the problem of optimising the bottom-up (recognition model) and top-down (generative model) connections in a multilayer neural network. This approach unifies many previous results on the optimisation of multilayer unsupervised neural networks.

1 Introduction

There is currently a great deal of interest in modelling probability density functions (PDF). This research is motivated by the fact that the joint PDF of a set of variables can be used to deduce any conditional PDF which involves these variables alone, which thus allows all inference problems in the space of these variables to be addressed quantitatively. The only limitation of this approach to solving inference problems is that a *model* of the PDF is used, rather than the *actual* PDF itself, which can lead to inaccurate inferences.

*© 1997 British Crown Copyright/DERA

The objective function for optimising a PDF model is usually to maximise the log-likelihood that it could generate the training set: i.e. maximise $\langle \log(\text{model probability}) \rangle_{\text{training set}}$.

There is also a great deal of interest in the design of optimal autoencoders, for encoding input vectors with the intention of reconstructing them with minimum average error. This would allow data to be transmitted along a limited bandwidth communication link, for instance. The objective function for optimising an autoencoder is usually to minimise the average squared reconstruction error over the training set: i.e. minimise $\left\langle \|\text{vector-reconstruction}\|^2 \right\rangle_{\text{training set}}$.

These two optimisation criteria are different from each other. For instance, autoencoding requires that information be retained about the input *itself*, whereas density modelling requires only that information be retained about the *PDF* of the input; these are different requirements. However, there is a way of expressing the autoencoder objective function which turns out to be equivalent to a density modelling objective function, although it turns out that the corresponding density model is not of the type that was referred to above.

The purpose of this report is to derive the theory that relates density modelling and autoencoding, and to show how many of the key results obtained by the author (during the past decade) may be derived from this theory. This unified theory may then be used to ensure that future results are backwardly compatible with past results.

In section 2 the standard Shannon theory of information is summarised, and its application to coding various types of source is derived; in particular, the Markov source is discussed, because it is central to the topic of this report. In section 3 the application of Markov source coding to unsupervised neural networks is discussed in detail, and the connection with folded Markov chains (FMC) [17] is derived. In section 4 density modelling of Markov sources is compared with standard density modelling using a Helmholtz machine [2, 3], which demonstrates a close connection and important differences between these two problems. The last three sections deal with particular applications of density modelling of Markov sources: section 5 deals with the Kohonen network [7], section 6 deals with partitioned mixture distributions (PMD) [18, 20], and section 7 deals with the adaptive cluster expansion (ACE) [14, 19, 21].

2 Coding Theory

Section 2.1 outlines the basic ideas of information theory, and section 2.2 describes in detail the process of using a model to code a source. In section 2.3 this is extended to the case of a Markov source, and in section 2.4 this is further extended (in outline only) to the case of a dynamical Markov source.

See [25] for a lucid introduction to information theory, and see [23, 24] for a discussion of the number of bits required to encode a source using a model.

2.1 Information Theory

A source of symbols (drawn from an alphabet of M distinct symbols) is modelled by a vector of probabilities $\mathbf{P}$

$$\mathbf{P} \equiv (P_1, P_2, \cdots, P_M) \tag{1}$$

which describes the relative frequency with which each symbol is drawn independently from the source $\mathbf{P}$. A trivial example is an unbiassed die, which has $M = 6$ and $P_i = \frac{1}{6}$ for $i = 1, 2, \cdots, 6$.

The ordered sequence of symbols drawn independently from a source may be partitioned into subsequences of N symbols, and each such subsequence will be called a message. If N is very large, then a message is *likely* if the relative frequency of occurrence of its symbols approximates $\mathbf{P}$, or *unlikely* if not. As $N \to \infty$ the set of messages that is likely is very sharply defined, so that there is a set of likely messages all with equal probability of occurring (because each likely message has the same relative frequency of occurrence of each symbol), and a set of unlikely messages (i.e. all the messages that are not likely messages) that have essentially zero probability of occurring. It is this separation of messages into a likely set (all with equal probability) and an unlikely set (all with zero probability) that underlies information theory.

A likely message from $\mathbf{P}$ will be called a likely $\mathbf{P}$-message. The number of times n_i that each symbol i occurs in a $\mathbf{P}$-message of length N is $n_i = NP_i$, where $\sum_{i=1}^{M} P_i = 1$ guarantees that the normalisation condition $\sum_{i=1}^{M} n_i = N$ is satisfied. The logarithm of the number of different likely $\mathbf{P}$-messages is given by (using Stirling's approximation $\log x! \approx x \log x - x$ when x is large)

$$\log\left(\frac{N!}{n_1! n_2! \cdots n_M!}\right) \approx N \log N - N - \sum_{i=1}^{M} n_i \log n_i + \sum_{i=1}^{M} n_i$$

$$
\begin{aligned}
&= N \log N - \sum_{i=1}^{M} n_i \log n_i \\
&= N \log N - \sum_{i=1}^{M} N\, P_i \log (N\, P_i) \\
&= N \log N - \sum_{i=1}^{M} N\, P_i \log N - \sum_{i=1}^{M} N\, P_i \log P_i \\
&= -N \sum_{i=1}^{M} P_i \log P_i \qquad (2)
\end{aligned}
$$

Now define the entropy $H(\mathbf{P})$ of source $\mathbf{P}$ as the logarithm of the number of different likely $\mathbf{P}$-messages (measured per message symbol)**:**

$$H(\mathbf{P}) \equiv -\sum_{i=1}^{M} P_i \log P_i \geq 0 \qquad (3)$$

Thus $H(\mathbf{P})$ is the number of bits per symbol (on average) that are required to encode the source (assuming a perfect encoder), because the only messages that the source has a finite probability of producing are the likely $\mathbf{P}$-messages that are enumerated in equation 2. The base of the logarithm determines the base in which the "bits" are measured. Thus base 2 logarithms correspond to "bits" that each have 2 states (i.e. binary digits), whereas base 10 logarithms correspond to "bits" that each have 10 states (i.e. decimal digits). A common error is to assume that the base of the logarithm somehow implies a corresponding discretisation of $H(\mathbf{P})$. The logarithm is used only as a convention to control the dynamic range of the quantity that is called *information*; its effect can be removed by exponentiation using the same base as was used for the logarithm in the first place.

It is usually very difficult to encode the source $\mathbf{P}$ using $H(\mathbf{P})$ bits per symbol on average. This is because although the boundary between the set of likely $\mathbf{P}$-messages and the set of unlikely $\mathbf{P}$-messages is sharply defined in principle, in practice it is very hard to model mathematically. If this boundary is not precisely defined, then it is impossible to compute the value of $H(\mathbf{P})$ accurately. In order to ensure that *all* of the likely $\mathbf{P}$-messages are accounted for, it is necessary for the mathematical model of the boundary to lie *outside* the true boundary, which thus overestimates the value of $H(\mathbf{P})$. This demonstrates that $H(\mathbf{P})$ is in fact a lower bound on the true number of bits per symbol that must be used to encode the source $\mathbf{P}$.

2.2 Source Coding

The mathematical model of the boundary between the set of likely **P**-messages and the set of unlikely **P**-messages may be derived from a vector of probabilities **Q**, whose M elements model the probability of each symbol drawn from an alphabet of M distinct symbols. If $\mathbf{Q} = \mathbf{P}$ then the boundary is modelled perfectly, and hence in principle the lower bound $H(\mathbf{P})$ on the number of bits per symbol may be attained, although even this is difficult to realise constructively in practice. In practical situations $\mathbf{Q} \neq \mathbf{P}$ is invariably the case, so the problem of coding a source with an inaccurate model cannot be avoided.

Constructive coding of a source using a model **Q** requires that **Q** be used to generate messages (**Q**-messages) which can then be compared with **P**-messages. Since the only **P**-messages that occur are the likely **P**-messages (these all occur with equal probability because each likely message has the same relative frequency of occurrence of each symbol) all we need to do in order to calculate the number of bits per symbol that is required when using **Q** to encode **P** is to calculate the probability that a **Q**-message is one of the likely **P**-messages (the probability that **Q** can generate each of the likely **P**-messages is the same), which is sufficient information to deduce the total number of bits per symbol that is required.

The log-probability $\Pi_N(\mathbf{P}, \mathbf{Q})$ that a **Q**-message is a likely **P**-message is

$$\begin{aligned}
\Pi_N(\mathbf{P}, \mathbf{Q}) &= \log\left(\frac{N!}{n_1!n_2!\cdots n_M!}\, Q_1^{n_1} Q_2^{n_2} \cdots Q_M^{n_M}\right) \\
&\approx -N\sum_{i=1}^{M} P_i \log P_i + N\sum_{i=1}^{M} P_i \log Q_i \\
&= -N\sum_{i=1}^{M} P_i \log \frac{P_i}{Q_i} \\
&\leq 0 \qquad (4)
\end{aligned}$$

which is negative because the model **Q** generates likely **P**-messages with less than unit probability.

The model **Q** must be used to generate enough **Q**-messages to ensure that all of the likely **P**-messages are reproduced. This requires the basic $H(\mathbf{P})$ bits per symbol that would be required if $\mathbf{Q} = \mathbf{P}$, plus some extra bits to compensate for the less than 100% efficiency (because $\mathbf{Q} \neq \mathbf{P}$) with which **Q** generates likely **P**-messages. The number of extra bits per symbol

is the relative entropy $G(\mathbf{P},\mathbf{Q})$

$$\begin{aligned} G(\mathbf{P},\mathbf{Q}) &\equiv \sum_{i=1}^{M} P_i \log \frac{P_i}{Q_i} \\ &\geq 0 \end{aligned} \tag{5}$$

which is $-\frac{\Pi_N(\mathbf{P},\mathbf{Q})}{N}$, or minus the log-probability that a $\mathbf{Q}$-message is a likely $\mathbf{P}$-message. Thus $\mathbf{Q}$ is used to generate exactly the number of extra $\mathbf{Q}$-messages that is required to compensate for the fact that the probability that each $\mathbf{Q}$-message is a likely $\mathbf{P}$-message is less than unity (i.e. $\Pi_N(\mathbf{P},\mathbf{Q}) \leq 0$).

$G(\mathbf{P},\mathbf{Q})$ (i.e. relative entropy) is the amount by which the number of bits per symbol exceeds the lower bound $H(\mathbf{P})$ (i.e. source entropy). Note that both $G(\mathbf{P},\mathbf{Q})$ and $H(\mathbf{P})$ are quantities that are realisable only in principle (i.e. they are lower bounds on the number of bits that is required in practice), because of the well-known practical difficulties associated with using a model $\mathbf{Q}$ to design an encoder. Define the total number of bits per symbol $H(\mathbf{P}) + G(\mathbf{P},\mathbf{Q})$ as $L(\mathbf{P},\mathbf{Q})$

$$\begin{aligned} L(\mathbf{P},\mathbf{Q}) &\equiv H(\mathbf{P}) + G(\mathbf{P},\mathbf{Q}) \\ &= -\sum_{i=1}^{M} P_i \log P_i + \sum_{i=1}^{M} P_i \log \frac{P_i}{Q_i} \\ &= -\sum_{i=1}^{M} P_i \log Q_i \\ &\geq 0 \end{aligned} \tag{6}$$

This expression for $G(\mathbf{P},\mathbf{Q})$ provides a means of optimising the model $\mathbf{Q}$. Ideally the number of extra bits that is required to compensate for the model's inefficiency should be as small as possible, which requires that the optimum model $\mathbf{Q}_{opt}$ should minimise the objective function $G(\mathbf{P},\mathbf{Q})$ with respect to $\mathbf{Q}$, thus

$$\begin{aligned} \mathbf{Q}_{opt} &= \underset{\mathbf{Q}}{\arg\min}\ G(\mathbf{P},\mathbf{Q}) \\ &= \underset{\mathbf{Q}}{\arg\max}\ \sum_{i=1}^{M} P_i \log Q_i \\ &= \underset{\mathbf{Q}}{\arg\max}\ \log\left(Q_1^{n_1} Q_2^{n_2} \cdots Q_M^{n_M}\right) \end{aligned} \tag{7}$$

where $\log\left(Q_1^{n_1} Q_2^{n_2} \cdots Q_M^{n_M}\right)$ is the log-probability that a message of length N generated by $\mathbf{Q}$ is a likely $\mathbf{P}$-message. This criterion for optimising a model will not include the number of bits that is required to specify the model *itself*, such as is used in the minimum description length approach [23, 24].

$G(\mathbf{P}, \mathbf{Q})$ is frequently used as an objective function in density modelling, where the optimum model $\mathbf{Q}_{opt}$ is chosen as the one that maximises the log-probability of generating the observed data $(n_1, n_2, \cdots, n_M)$. Since $\mathbf{Q}_{opt}$ must, in some sense, be close to $\mathbf{P}$, this affords a practical way of ensuring that the optimum model probabilities $\mathbf{Q}_{opt}$ are similar to the source probabilities $\mathbf{P}$, which is the goal of density modelling.

2.3 Markov Source Coding

The above scheme for using a model $\mathbf{Q}$ to code symbols derived from a source $\mathbf{P}$ may be extended to the case where the source and the model are L-layer Markov chains. Thus split up each of $\mathbf{P}$ and $\mathbf{Q}$ into separate pieces associated with each layer, or pair of adjacent layers.

$$\begin{aligned}
\mathbf{P} &= \left(\mathbf{P}^0, \mathbf{P}^{1|0}, \cdots, \mathbf{P}^{L-1|L-2}, \mathbf{P}^{L|L-1}\right) \\
&= \left(\mathbf{P}^{0|1}, \mathbf{P}^{1|2}, \cdots, \mathbf{P}^{L-1|L}, \mathbf{P}^{L}\right) \\
\mathbf{Q} &= \left(\mathbf{Q}^0, \mathbf{Q}^{1|0}, \cdots, \mathbf{Q}^{L-1|L-2}, \mathbf{Q}^{L|L-1}\right) \\
&= \left(\mathbf{Q}^{0|1}, \mathbf{Q}^{1|2}, \cdots, \mathbf{Q}^{L-1|L}, \mathbf{Q}^{L}\right) \qquad (8)
\end{aligned}$$

where $\mathbf{P}^{k|l}$ ($\mathbf{Q}^{k|l}$) is the matrix of transition probabilities from layer l to layer k of the Markov chain of the source (model), $\mathbf{P}^0$ ($\mathbf{Q}^0$) is the vector of marginal probabilities in layer L, $\mathbf{P}^L$ ($\mathbf{Q}^L$) is the vector of marginal probabilities in layer 0. These two ways of decomposing $\mathbf{P}$ (and $\mathbf{Q}$) are equivalent, because a forward pass through a Markov chain may be converted into a backward pass through a different Markov chain, whose transition probabilities are uniquely determined by applying Bayes' theorem to the original Markov chain.

The number of extra bits per symbol that is required to encode the source $\mathbf{P}$ with the model $\mathbf{Q}$ is given by

$$G(\mathbf{P}, \mathbf{Q}) = \sum_{i_0=1}^{M_0} \cdots \sum_{i_L=1}^{M_L} P_{i_0,i_1}^{0|1} P_{i_1,i_2}^{1|2} \cdots P_{i_{L-1},i_L}^{L-1|L} P_{i_L}^{L} \log \left(\frac{P_{i_0,i_1}^{0|1} P_{i_1,i_2}^{1|2} \cdots P_{i_{L-1},i_L}^{L-1|L} P_{i_L}^{L}}{Q_{i_0,i_1}^{0|1} Q_{i_1,i_2}^{1|2} \cdots Q_{i_{L-1},i_L}^{L-1|L} Q_{i_L}^{L}} \right)$$

$$
\begin{aligned}
&= \sum_{i_1=1}^{M_1} P_{i_1}^1 \sum_{i_0=1}^{M_0} P_{i_0,i_1}^{0|1} \log \frac{P_{i_0,i_1}^{0|1}}{Q_{i_0,i_1}^{0|1}} + \sum_{i_2=1}^{M_2} P_{i_2}^2 \sum_{i_1=1}^{M_1} P_{i_1,i_2}^{1|2} \log \frac{P_{i_1,i_2}^{1|2}}{Q_{i_1,i_2}^{1|2}} \\
&\quad + \cdots + \sum_{i_L=1}^{M_L} P_{i_L}^L \sum_{i_{L-1}=1}^{M_{L-1}} P_{i_{L-1},i_L}^{L-1|L} \log \frac{P_{i_{L-1},i_L}^{L-1|L}}{Q_{i_{L-1},i_L}^{L-1|L}} \\
&\quad + \sum_{i_L=1}^{M_L} P_{i_L}^L \log \frac{P_{i_L}^L}{Q_{i_L}^L} \\
&= \sum_{i_1=1}^{M_1} P_{i_1}^1 G_{i_1}\left(P^{0|1}, Q^{0|1}\right) + \sum_{i_2=1}^{M_2} P_{i_2}^2 G_{i_2}\left(P^{1|2}, Q^{1|2}\right) \\
&\quad + \cdots + \sum_{i_L=1}^{M_L} P_{i_L}^L G_{i_L}\left(P^{L-1|L}, Q^{L-1|L}\right) \\
&\quad + G(P^L, Q^L)
\end{aligned} \tag{9}
$$

where the suffix i_l that appears on the $G_{i_l}\left(P^{l-1|l}, Q^{l-1|l}\right)$ indicates that the state of layer l is fixed during the evaluation of $G_{i_l}\left(P^{l-1|l}, Q^{l-1|l}\right)$ (i.e. it is the relative entropy of layer $l-1$, given that the state of layer l is known). This may be interpreted as the number of extra bits per symbol $G(P^L, Q^L)$ that is required to encode the L^{th} layer of the Markov chain, plus the sum over layers l (for $0 \leq l \leq L-1$) of the number of extra bits per symbol $G_{i_l}\left(P^{l-1|l}, Q^{l-1|l}\right)$ that is required to make the transition backwards from layer l to layer $l-1$ (for $1 \leq l \leq L$) of the Markov chain (averaged over all states of layer l using $\sum_{i_l=1}^{M_l} P_{i_l}^l G_{i_l}\left(P^{l-1|l}, Q^{l-1|l}\right)$). Using Bayes' theorem, this expression for $G(\mathbf{P}, \mathbf{Q})$ can be manipulated into a form which starts at layer 0, and then makes forwards transitions from layer to layer, to eventually arrive at layer L. However, in this report, only the backwards pass through the Markov chain will be used.

The total number of bits per symbol that is required to code the source $\mathbf{P}$ with the model $\mathbf{Q}$ is $L(\mathbf{P}, \mathbf{Q})$ (i.e. $H(\mathbf{P}) + G(\mathbf{P}, \mathbf{Q})$), which is given by

$$
\begin{aligned}
L(\mathbf{P}, \mathbf{Q}) &= -\sum_{i_1=1}^{M_1} P_{i_1}^1 \sum_{i_0=1}^{M_0} P_{i_0,i_1}^{0|1} \log Q_{i_0,i_1}^{0|1} - \sum_{i_2=1}^{M_2} P_{i_2}^2 \sum_{i_1=1}^{M_1} P_{i_1,i_2}^{1|2} \log Q_{i_1,i_2}^{1|2} \\
&\quad - \cdots - \sum_{i_L=1}^{M_L} P_{i_L}^L \sum_{i_{L-1}=1}^{M_{L-1}} P_{i_{L-1},i_L}^{L-1|L} \log Q_{i_{L-1},i_L}^{L-1|L}
\end{aligned}
$$

$$
\begin{aligned}
& -\sum_{i_L=1}^{M_L} P_{i_L}^{L} \log Q_{i_L}^{L} \\
= & -\sum_{l=0}^{L-1} \sum_{i_{l+1}=1}^{M_{l+1}} P_{i_{l+1}}^{l+1} \sum_{i_l=1}^{M_l} P_{i_l,i_{l+1}}^{l|l+1} \log Q_{i_l,i_{l+1}}^{l|l+1} - \sum_{i_L=1}^{M_L} P_{i_L}^{L} \log Q_{i_L}^{L} \\
= & \sum_{l=0}^{L-1} \sum_{i_{l+1}=1}^{M_{l+1}} P_{i_{l+1}}^{l+1} L_{i_{l+1}} \left(\mathbf{P}^{l|l+1}, \mathbf{Q}^{l|l+1} \right) + L \left(\mathbf{P}^{L}, \mathbf{Q}^{L} \right) \qquad (10)
\end{aligned}
$$

where the suffix i_{l+1} appears on the $L_{i_{l+1}} \left(\mathbf{P}^{l|l+1}, \mathbf{Q}^{l|l+1} \right)$ because the state of layer $l+1$ is fixed during the evaluation of $L_{i_{l+1}} \left(\mathbf{P}^{l|l+1}, \mathbf{Q}^{l|l+1} \right)$. This may be interpreted as the total number of bits per symbol $L \left(\mathbf{P}^{L}, \mathbf{Q}^{L} \right)$ that is required to encode the L^{th} layer of the Markov chain, plus the sum over layers l (for $0 \leq l \leq L-1$) of the total number of bits per symbol $L_{i_{l+1}} \left(\mathbf{P}^{l|l+1}, \mathbf{Q}^{l|l+1} \right)$ that is required to make the transition backwards from layer l to layer $l-1$ (for $1 \leq l \leq L$) of the Markov chain (averaged over all states of layer l using $\sum_{i_{l+1}=1}^{M_{l+1}} P_{i_{l+1}}^{l+1} L_{i_{l+1}} \left(\mathbf{P}^{l|l+1}, \mathbf{Q}^{l|l+1} \right)$).

This result has a very natural interpretation. Both the source $\mathbf{P}$ and the model $\mathbf{Q}$ are Markov chains, and corresponding parts of the model are matched up with corresponding parts of the source. First of all, the number of bits that is required to encode the L^{th} layer of the source is $L \left(\mathbf{P}^{L}, \mathbf{Q}^{L} \right)$. Having done that, the number of bits that is required to encode the $L-1^{th}$ layer of the source, given that the state of the L^{th} layer is already known, is $L \left(\mathbf{P}^{L-1|L}, \mathbf{Q}^{L-1|L} \right)$, which must then be averaged over the alternative possible states of the L^{th} layer to yield $\sum_{i_L=1}^{M_L} P_{i_L}^{L} L \left(\mathbf{P}^{L-1|L}, \mathbf{Q}^{L-1|L} \right)$.This process is then repeated to encode the $L-2^{th}$ layer of the source, given that the state of the $L-1^{th}$ layer is already known, and so on back to layer 0. This yields precisely the expression for $L(\mathbf{P}, \mathbf{Q})$ given above.

2.4 Dynamical Markov Source Encoding

The above theory of coding Markov sources, in which both the source $\mathbf{P}$ and the corresponding model $\mathbf{Q}$ are multilayer Markov chains, may be extended to the case where each layer has a memory of its own previous state; thus the static Markov source becomes a dynamical Markov source (usually with a discretised time index). In this case the source and the model are doubly

Markov, where, in the simplest case, there is one Markov chain linking together different layers at the same time slice (as above), and there is another Markov chain linking together the same layer at different time slices. There are many possible variations on this theme.

This dynamical source differs from the static source used previously only insofar as $\mathbf{P}^{l+1|l}$ is now modulated by a prior probability on the states of layer $l+1$ (in the simplest case), so that the Markov source $\mathbf{P}(t)$ at time slice t has a statistical structure that depends directly on $\mathbf{P}(t-1)$, and thus indirectly on all $\mathbf{P}(\tau)$ for $\tau < t-1$. The simplest model $\mathbf{Q}(t)$ that can be used for this dynamical source is the same as was used in the case of a static source, but this will not be as efficient a model as one which modelled the dynamics of the source. Apart from the introduction of a state-dependent prior probability, the entire theory of dynamical sources and models is the same as the static theory.

3 Application To Unsupervised Neural Networks

In section 3.1 the theory of Markov source coding (that was presented in section 2.3) is applied to a multilayer neural network. In section 3.2 this approach is applied to a 2-layer neural network to obtain a folded Markov chain (FMC) network [17], which is generalised to a multilayer neural network in section 3.3 to obtain a network of coupled 2-layer FMCs. In section 3.4 a crude "mean field" approach to optimising this type of multilayer network is presented, where the concept of probability leakage is introduced. Finally, the problem of coding the output layer of a multilayer network is addressed in section 3.5.

3.1 Source Model Of Layered Network

In this section the optimisation of the joint PDF of the states of all of the layers of an $(L+1)$-layer unsupervised neural network will be considered. It turns out that this leads to new insight into the optimisation of a multilayer encoder network.

The Markov chain source $\mathbf{P} = \left(\mathbf{P}^0, \mathbf{P}^{1|0}, \cdots, \mathbf{P}^{L-1|L-2}, \mathbf{P}^{L|L-1}\right)$ (or, equivalently, $\mathbf{P} = \left(\mathbf{P}^{0|1}, \mathbf{P}^{1|2}, \cdots, \mathbf{P}^{L-1|L}, \mathbf{P}^{L}\right)$) may be used to describe the true behaviour (i.e. not merely a model) of a layered neural network as follows

$$P^0_{i_0} \quad = \quad \text{true probability that layer 0 has state } i_0$$

$$
\begin{aligned}
P^L_{i_L} &= \text{true probability that layer } L \text{ has state } i_L \\
P^{l+1|l}_{i_{l+1},i_l} &= \text{true probability that layer } l+1 \text{ has state } i_{l+1} \\
&\quad \text{given that layer } l \text{ has state } i_l \\
P^{l|l+1}_{i_l,i_{l+1}} &= \text{true probability that layer } l \text{ has state } i_l \\
&\quad \text{given that layer } l+1 \text{ has state } i_{l+1}
\end{aligned} \tag{11}
$$

Thus $\mathbf{P}^0$ is an external source, and $\left(\mathbf{P}^{1|0}, \cdots, \mathbf{P}^{L-1|L-2}, \mathbf{P}^{L|L-1}\right)$ is an internal source, where external/internal describes whether the source is outside/inside the layered network, respectively. $\mathbf{P}^{l+1|l}$ is not part of the source itself (i.e. the external source), rather it is the way in which layer l of the neural network is connected to layer $l+1$. There is an analogous interpretation of $\mathbf{P}^L$ and the $\mathbf{P}^{l|l+1}$.

The Markov chain model $\mathbf{Q} = \left(\mathbf{Q}^0, \mathbf{Q}^{1|0}, \cdots, \mathbf{Q}^{L-1|L-2}, \mathbf{Q}^{L|L-1}\right)$ (or, equivalently, $\mathbf{Q} = \left(\mathbf{Q}^{0|1}, \mathbf{Q}^{1|2}, \cdots, \mathbf{Q}^{L-1|L}, \mathbf{Q}^L\right)$) may then be used as a model (i.e. not actually the true behaviour) of a layered neural network as follows

$$
\begin{aligned}
Q^0_{i_0} &= \text{model probability that layer 0 has state } i_0 \\
Q^L_{i_L} &= \text{model probability that layer } L \text{ has state } i_L \\
Q^{l+1|l}_{i_{l+1},i_l} &= \text{model probability that layer } l+1 \text{ has state } i_{l+1} \\
&\quad \text{given that layer } l \text{ has state } i_l \\
Q^{l|l+1}_{i_l,i_{l+1}} &= \text{model probability that layer } l \text{ has state } i_l \\
&\quad \text{given that layer } l+1 \text{ has state } i_{l+1}
\end{aligned} \tag{12}
$$

$\mathbf{Q}$ has an analogous interpretation to $\mathbf{P}$, except that it is a model of the source, rather than the true behaviour of the source.

It turns out to be useful for the true Markov behaviour (i.e. $\mathbf{P}$) and the model Markov behaviour (i.e. $\mathbf{Q}$) to run in opposite directions through the Markov chain. Thus $\mathbf{P} = \left(\mathbf{P}^0, \mathbf{P}^{1|0}, \cdots, \mathbf{P}^{L-1|L-2}, \mathbf{P}^{L|L-1}\right)$ (flow of influence from layer 0 to layer L of the Markov chain) and $\mathbf{Q} = \left(\mathbf{Q}^{0|1}, \mathbf{Q}^{1|2}, \cdots, \mathbf{Q}^{L-1|L}, \mathbf{Q}^L\right)$ (flow of influence from layer L to layer 0 of the Markov chain). In the conventional language of neural networks, $\mathbf{P}$ is a "recognition model" and $\mathbf{Q}$ is a "generative model". Note that the terminology "recognition model" is strictly speaking not accurate in this context, because $\mathbf{P}$ describes the true behaviour (i.e. it is not merely a model) of a multilayer source. The

effect of the $\mathbf{P}^{l+1|l}$ on the external source $\mathbf{P}^0$ is to compute (in a stochastic fashion) various functions of the state of the source, so the $\mathbf{P}^{l+1|l}$ can be interpreteted as computing "statistics" of the external source. A better terminology would be to say that $\mathbf{P}$ is a "multilayer statistic", rather than a "recognition model".

However, terminology depends on one's viewpoint. In Markov chain density modelling $\mathbf{P}$ is a source when viewed from the point of view of the model $\mathbf{Q}$. In conventional density modelling $\mathbf{P}^0$ is a source when viewed from the point of model $\mathbf{Q}^0$, in which case $\left(\mathbf{P}^{1|0}, \cdots, \mathbf{P}^{L-1|L-2}, \mathbf{P}^{L|L-1}\right)$ is a recognition model and $\left(\mathbf{Q}^{0|1}, \mathbf{Q}^{1|2}, \cdots, \mathbf{Q}^{L-1|L}, \mathbf{Q}^{L}\right)$ is a generative model. In this report, terminology will thus be used in a context-dependent way.

Now evaluate the expression for $L(\mathbf{P}, \mathbf{Q})$ in the case where $\mathbf{P}$ and $\mathbf{Q}$ run in opposite directions through the Markov chain. Thus use Bayes' theorem in the form

$$P_{i_{l+1}}^{l+1} P_{i_l, i_{l+1}}^{l|l+1} = P_{i_l}^{l} P_{i_{l+1}, i_l}^{l+1|l} \tag{13}$$

and define $K_{i_l}\left(\mathbf{P}^{l+1|l}, \mathbf{Q}^{l|l+1}\right)$ in such a way that it depends on $P_{i_{l+1}, i_l}^{l+1|l}$ (flow of influence from layer l to layer $l+1$) and $\log Q_{i_l, i_{l+1}}^{l|l+1}$ (flow of influence from layer $l+1$ to layer l)

$$K_{i_l}\left(\mathbf{P}^{l+1|l}, \mathbf{Q}^{l|l+1}\right) \equiv -\sum_{i_{l+1}=1}^{M_{l+1}} P_{i_{l+1}, i_l}^{l+1|l} \log Q_{i_l, i_{l+1}}^{l|l+1} \tag{14}$$

The $P_{i_{l+1}}^{l+1} L_{i_{l+1}}\left(\mathbf{P}^{l|l+1}, \mathbf{Q}^{l|l+1}\right)$ terms in $L(\mathbf{P}, \mathbf{Q})$ (see equation 10) may thus be rewritten as

$$\sum_{i_{l+1}=1}^{M_{l+1}} P_{i_{l+1}}^{l+1} L_{i_{l+1}}\left(\mathbf{P}^{l|l+1}, \mathbf{Q}^{l|l+1}\right) = \sum_{i_l=1}^{M_l} P_{i_l}^{l} K_{i_l}\left(\mathbf{P}^{l+1|l}, \mathbf{Q}^{l|l+1}\right) \tag{15}$$

whence $L(\mathbf{P}, \mathbf{Q})$ may finally be written as

$$L(\mathbf{P}, \mathbf{Q}) = \sum_{l=0}^{L-1} \sum_{i_l=1}^{M_l} P_{i_l}^{l} K_{i_l}\left(\mathbf{P}^{l+1|l}, \mathbf{Q}^{l|l+1}\right) + L\left(\mathbf{P}^{L}, \mathbf{Q}^{L}\right) \tag{16}$$

The various parts of the $\sum_{l=0}^{L-1} \sum_{i_l=1}^{M_l} P_{i_l}^{l} K_{i_l}\left(\mathbf{P}^{l+1|l}, \mathbf{Q}^{l|l+1}\right)$ term may be interpreted as follows. $P_{i_l}^{l}$ is the source probability that the state of layer l is i_l (after propagation of the external source from layer 0 via layers $1, 2, \cdots, l-$

1), $P^{l+1|l}_{i_{l+1},i_l}$ is the source probability that the state of layer $l+1$ is i_{l+1} given that the state of layer l is i_l, and $Q^{l|l+1}_{i_l,i_{l+1}}$ is the model probability that the state of layer l is i_l given that the state of layer $l+1$ is i_{l+1}. Finally, the term $L\left(\mathbf{P}^L, \mathbf{Q}^L\right)$ is the total number of bits that is required to code the L^{th} layer of the network (i.e. its output layer).

3.2 2-Layer Folded Markov Chain Network

The expression for $L(\mathbf{P}, \mathbf{Q})$ in equation 16 is rather complicated, but it has a simple internal structure which allows it to be systematically analysed. Thus apply equation 16 to a 2-layer network to obtain the objective function

$$L(\mathbf{P}, \mathbf{Q}) = \sum_{i_0=1}^{M_0} P^0_{i_0} K_{i_0}\left(\mathbf{P}^{1|0}, \mathbf{Q}^{0|1}\right) + L\left(\mathbf{P}^1, \mathbf{Q}^1\right) \tag{17}$$

Now change notation in order to make contact with previous results on vector quantisers (VQ) [9]

$$\begin{array}{lll}
i_0 \to \mathbf{x} & \sum_{i_0=1}^{M_0} \to \int d\mathbf{x} & \text{input vector} \\
i_1 \to y & \sum_{i_1=1}^{M_1} \to \sum_{y=1}^{M} & \text{output code index} \\
P^0_{i_0} \to \Pr(\mathbf{x}) & & \text{input PDF} \\
P^{1|0}_{i_1,i_0} \to \Pr(y|\mathbf{x}) & & \text{recognition model} \\
Q^{0|1}_{i_0,i_1} \to V \frac{1}{\sqrt{2\pi}\sigma} \exp\left(-\frac{\|\mathbf{x}-\mathbf{x}'(y)\|^2}{2\sigma^2}\right) & & \text{Gaussian generative model} \\
Q^1_{i_1} \to Q(y) & & \text{output prior}
\end{array} \tag{18}$$

where $\mathbf{x}$ is a continuous-valued input vector (e.g. the activity pattern in layer 0), σ is the (isotropic) variance of the Gaussian generative model, V is an infinitesimal volume element in input space, and y is a discrete-valued output index (e.g. the location of the next neuron to fire in layer 1). This allows $L(\mathbf{P}, \mathbf{Q})$ to be written as

$$\begin{aligned}
L(\mathbf{P}, \mathbf{Q}) &= -\int d\mathbf{x} \Pr(\mathbf{x}) \sum_{y=1}^{M} \Pr(y|\mathbf{x}) \log\left(V \frac{1}{\sqrt{2\pi}\sigma} \exp\left(-\frac{\|\mathbf{x}-\mathbf{x}'(y)\|^2}{2\sigma^2}\right)\right) + L\left(\mathbf{P}^1, \mathbf{Q}^1\right) \\
&= \frac{1}{2\sigma^2} \int d\mathbf{x} \Pr(\mathbf{x}) \sum_{y=1}^{M} \Pr(y|\mathbf{x}) \ \|\mathbf{x}-\mathbf{x}'(y)\|^2 - \log\frac{V}{\sqrt{2\pi}\sigma} + L\left(\mathbf{P}^1, \mathbf{Q}^1\right)
\end{aligned} \tag{19}$$

Now define the 2-layer "folded Markov chain" (FMC) objective function D_{FMC} as (see [17] for details of the FMC approach)

$$D_{FMC} \equiv \int d\mathbf{x} \Pr(\mathbf{x}) \sum_{y=1}^{M} \Pr(y|\mathbf{x}) \int d\mathbf{x}' \Pr(\mathbf{x}'|y) \left\|\mathbf{x} - \mathbf{x}'\right\|^2 \tag{20}$$

and use the symmetry of D_{FMC} to write it in the form

$$D_{FMC} = 2 \int d\mathbf{x} \Pr(\mathbf{x}) \sum_{y=1}^{M} \Pr(y|\mathbf{x}) \left\|\mathbf{x} - \mathbf{x}'(y)\right\|^2 \tag{21}$$

where $\mathbf{x}'(y)$ takes the value that minimises D_{FMC} (i.e. $\mathbf{x}'(y) = \int d\mathbf{x} \Pr(\mathbf{x}|y)\ \mathbf{x}$).

This allows $L(\mathbf{P}, \mathbf{Q})$ to be written as

$$L(\mathbf{P}, \mathbf{Q}) = \frac{1}{4\sigma^2} D_{FMC} - \log \frac{V}{\sqrt{2\pi}\sigma} + L\left(\mathbf{P}^1, \mathbf{Q}^1\right) \tag{22}$$

If the cost of coding the output layer (i.e. $L(\mathbf{P}^1, \mathbf{Q}^1)$) is ignored, then provided that V and σ are fixed quantities, the 2-layer Markov source coding objective function $L(\mathbf{P}, \mathbf{Q})$ can be minimised by minimising the 2-layer FMC objective function D_{FMC}. The basic FMC approach can be generalised by replacing the (isotropic) variance σ by a vector of (anisotropic) variances, or even a full covariance matrix if there is enough training data to permit this.

Equation 21 is also the objective function for a soft vector quantiser (VQ), where $\Pr(y|\mathbf{x})$ is a soft encoder, and $\mathbf{x}'(y)$ is reconstruction vector attached to code index y, and $\left\|\mathbf{x} - \mathbf{x}'(y)\right\|^2$ is the L^2 norm of the reconstruction error. A hard VQ (i.e. winner-take-all encoder) has $\Pr(y|\mathbf{x}) = \delta_{y,y(\mathbf{x})}$; this emerges as the optimal form when D_{FMC} is minimised w.r.t. $\Pr(y|\mathbf{x})$. This shows that the VQ objective function (equation 21) is closely related to the objective function for 2-layer Markov source encoding (equation 17), provided that the cost of coding the output layer $L(\mathbf{P}^1, \mathbf{Q}^1)$ is ignored.

The effect of the $L(\mathbf{P}^1, \mathbf{Q}^1)$ term in $L(\mathbf{P}, \mathbf{Q})$ is to encourage $P_i^1 \to \delta_{i,i_0}$ (only one state in layer 1 is used) and $\mathbf{Q}^1 \to \mathbf{P}^1$ (perfect model in layer 1). The behaviour $P_i^1 \to \delta_{i,i_0}$ is in conflict with the requirements of the D_{FMC} term in $L(\mathbf{P}, \mathbf{Q})$, which requires that more than one state in layer 1 is used, in order to minimise the reconstruction distortion.There is a tradeoff between increasing the number of active states in layer 1 in order to enable the Gaussian generative model ($\mathbf{Q}^0$ is a Gaussian mixture distribution) to make a good approximation to the external source $\mathbf{P}^0$, and decreasing the

number of active states in layer 1 in order to make the average total number of bits $L\left(\mathbf{P}^1, \mathbf{Q}^1\right)$ that are required to specify an output state as small as possible.

In this report the $L\left(\mathbf{P}^1, \mathbf{Q}^1\right)$ term will usually be omitted. The optimal network is then a hard VQ, where only 1 output state is active for a given input state, but different output states are used for input states that lie in different quantisation cells, so the net effect is that all output states are used.

3.3 Coupled FMC Networks

The results of section 3.2 will now be generalised to an $(L+1)$-layer network. The objective function for coding a Markov source (equation 16) can be written, using a notation which is analogous to that given in equation 18 (where the superscripts are layer indices) as

$$L\left(\mathbf{P}, \mathbf{Q}\right) = \sum_{l=0}^{L-1} \left(\frac{D_{FMC}^l}{4\left(\sigma^l\right)^2} - \log \frac{V^l}{\sqrt{2\pi}\sigma^l} \right) + L\left(\mathbf{P}^L, \mathbf{Q}^L\right) \tag{23}$$

which is a sum of 2-layer FMC objective functions (where each term is weighted by $\left(\sigma^l\right)^{-2}$), plus an output coding cost $L\left(\mathbf{P}^L, \mathbf{Q}^L\right)$. The layer index l identifies the input layer of each of the 2-layer FMCs, and the output layer of the l^{th} 2-layer FMC is identified with the input layer of the $(l+1)^{th}$ 2-layer FMC, which overall yields a chain of L coupled 2-layer FMCs. This will be called an *FMC-ladder*, or simply a ladder.

If the cost of coding the output layer is ignored, then the multilayer Markov source coding objective function $L\left(\mathbf{P}, \mathbf{Q}\right)$ is minimised by minimising the sum of 2-layer FMC objective functions $\sum_{l=0}^{L-1} \frac{D_{FMC}^l}{\left(\sigma^l\right)^2}$. As the number of network layers is increased, the effect of omitting $L\left(\mathbf{P}^L, \mathbf{Q}^L\right)$ has less and less effect on the overall optimisation, because its effect is swamped by the $\sum_{l=0}^{L-1} \frac{D_{FMC}^l}{\left(\sigma^l\right)^2}$ term.

Just as the objective function for a 2-layer FMC (equation 21) is equivalent to the objective function for a soft VQ, the objective function $\sum_{l=0}^{L-1} \frac{D_{FMC}^l}{\left(\sigma^l\right)^2}$ for an FMC-ladder is equivalent to the objective function for a chain of coupled soft VQs [13]. In the simplest case, the VQ connecting layer l to layer $l+1$ encodes the scalar code index y^l which is output by the VQ connecting layer $l-1$ to layer l. Clearly, a VQ is not necessarily a good way of

encoding y^l, because VQs are designed to encode continuous-valued vectors. This problem will be addressed in the section on probability leakage (section 3.4), where the properties of the output from a VQ are optimised in such a way as to approximate what the input of the next VQ expects to see.

3.4 Probability Leakage

The objective function $\sum_{l=0}^{L-1} \frac{D^l_{FMC}}{(\sigma^l)^2}$ for an FMC-ladder couples the optimisation of the individual 2-layer FMCs together. Because the output of FMC l is the input of FMC $l+1$ (for $l = 0, 1, \cdots, L-2$), the optimisation of FMC k has side effects on the optimisation of FMCs $k+1, k+2, \cdots, L-1$. This leads to the effect called self-supervision, in which top-down connections from higher to lower network layers are automatically generated, to allow the lower layers to process their input more effectively in the light of what the higher layers discover in the data [15, 16]. This can be made explicit in the objective function by grouping the terms as follows

$$\sum_{l=0}^{L-1} \frac{D^l_{FMC}}{(\sigma^l)^2} = \frac{D^0_{FMC}}{(\sigma^0)^2} + \left(\frac{D^1_{FMC}}{(\sigma^1)^2} + \left(\frac{D^2_{FMC}}{(\sigma^2)^2} + (\cdots(\cdots(\cdots))) \right) \right) \quad (24)$$

From the point of view of FMC k, the effect of FMCs $k+1, k+2, \cdots, L-1$ is to add an additional piece of objective function to the basic FMC objective function $\frac{D^k_{FMC}}{(\sigma^k)^2}$ thus

$$\frac{D^k_{FMC}}{(\sigma^k)^2} \rightarrow \frac{D^k_{FMC}}{(\sigma^k)^2} + \left(\frac{D^{k+1}_{FMC}}{(\sigma^{k+1})^2} + (\cdots(\cdots(\cdots))) \right) \quad (25)$$

This expression can be bounded above as follows

$$\frac{D^k_{FMC}}{(\sigma^k)^2} + \left(\frac{D^{k+1}_{FMC}}{(\sigma^{k+1})^2} + (\cdots(\cdots(\cdots))) \right) \leq \frac{D^k_{FMC}}{(\sigma^k)^2} + \left[\frac{D^{k+1}_{FMC}}{(\sigma^{k+1})^2} + (\cdots(\cdots(\cdots))) \right]_{\text{worst case}} \quad (26)$$

where the input to each of the FMCs $k+1, k+2, \cdots, L-1$ is assumed to be uniformly distributed in the worst case. Minimising D^k_{FMC} then locates a least upper bound on $\frac{D^k_{FMC}}{(\sigma^k)^2} + \left(\frac{D^{k+1}_{FMC}}{(\sigma^{k+1})^2} + (\cdots(\cdots(\cdots))) \right)$, as required.

A tighter upper bound can be obtained by combining FMC k and FMC $k+1$ into a single 3-layer FMC [17] whose input and output are layer k and layer $k+2$ respectively, and there is also a hidden layer $k+1$.

$$\frac{D^k_{FMC}}{(\sigma^k)^2} + \left(\frac{D^{k+1}_{FMC}}{(\sigma^{k+1})^2} + (\cdots(\cdots(\cdots))) \right) \leq \frac{D^{k,k+1}_{FMC}}{(\sigma^{k,k+1})^2} + \left[\frac{D^{k+2}_{FMC}}{(\sigma^{k+2})^2} + (\cdots(\cdots(\cdots))) \right]_{\text{worst case}} \quad (27)$$

where a self-explanatory notation has been used. The 3-layer FMC objective function (see equation 20 for the 2-layer case) is given by

$$\begin{aligned} D_{FMC}^{k,k+1} &= \sum_{y^k=1}^{M_k} \Pr\left(y^k\right) \sum_{y^{k+1}=1}^{M_{k+1}} \Pr\left(y^{k+1}|y^k\right) \sum_{y^{k+2}=1}^{M_{k+2}} \Pr\left(y^{k+2}|y^{k+1}\right) \\ &\quad \sum_{y'^k=1}^{M_k} \Pr\left(y'^k|y'^{k+1}\right) \sum_{y'^{k+1}=1}^{M_{k+1}} \Pr\left(y'^{k+1}|y^{k+2}\right) \\ &\quad \times \left\|y^k - y'^k\right\|^2 \end{aligned} \tag{28}$$

The summations $\sum_{y^{k+2}=1}^{M_{k+2}} (\cdots)$ can be evaluated thus

$$\Pr\left(y'^{k+1}|y^{k+1}\right) = \sum_{y^{k+2}=1}^{M_{k+2}} \Pr\left(y'^{k+1}|y^{k+2}\right) \Pr\left(y^{k+2}|y^{k+1}\right) \tag{29}$$

which yields

$$\begin{aligned} D_{FMC}^{k,k+1} &= \sum_{y^k=1}^{M_k} \Pr\left(y^k\right) \sum_{y'^{k+1}=1}^{M_{k+1}} \left(\sum_{y^{k+1}=1}^{M_{k+1}} \Pr\left(y'^{k+1}|y^{k+1}\right) \Pr\left(y^{k+1}|y^k\right) \right) \\ &\quad \times \sum_{y'^k=1}^{M_k} \Pr\left(y'^k|y'^{k+1}\right) \left\|y^k - y'^k\right\|^2 \end{aligned} \tag{30}$$

which may be rearranged (in the same way that equation 21 is obtained from equation 20) to obtain

$$D_{FMC}^{k,k+1} = 2 \sum_{y^k=1}^{M_k} \Pr\left(y^k\right) \sum_{y^{k+1}=1}^{M_{k+1}} \left(\sum_{y'^{k+1}=1}^{M_{k+1}} \Pr\left(y^{k+1}|y'^{k+1}\right) \Pr\left(y'^{k+1}|y^k\right) \right) \left\|y^k - y'^k\left(y^{k+1}\right)\right\|^2 \tag{31}$$

where the notation y^{k+1} and y'^{k+1} have been interchanged for convenience. If this result is compared with the expression for D_{FMC}^k, then it is seen that $D_{FMC}^{k,k+1}$ can be obtained from D_{FMC}^k by making the replacement

$$\Pr\left(y^{k+1}|y^k\right) \rightarrow \sum_{y'^{k+1}=1}^{M_{k+1}} \Pr\left(y^{k+1}|y'^{k+1}\right) \Pr\left(y'^{k+1}|y^k\right) \tag{32}$$

where the transition matrix element $\Pr\left(y^{k+1}|y'^{k+1}\right)$ is given in equation 29; it specifies the probability that code index y'^{k+1} is damaged by the 2-layer

FMC $k+1$ in such a way as to convert it to code index y^{k+1}. Numerical values for the $\Pr\left(y^{k+1}|y'^{k+1}\right)$ can be assigned assuming a manifestly suboptimal 2-layer FMC $k+1$, which will give an upper bound on $D_{FMC}^{k,k+1}$. For instance, $\Pr\left(y^{k+1}|y'^{k+1}\right)$ could be modelled by an additive Gaussian noise process, with a zero mean and a large enough variance that it guarantees an upper bound on $D_{FMC}^{k,k+1}$. Finally, minimising this upper bound on $D_{FMC}^{k,k+1}$ (i.e. D_{FMC}^{k} with the replacement in equation 32) then locates a least upper bound on $\frac{D_{FMC}^{k,k+1}}{\left(\sigma^{k,k+1}\right)^2} + \left(\frac{D_{FMC}^{k+2}}{\left(\sigma^{k+2}\right)^2} + (\cdots(\cdots(\cdots)))\right)$, as required.

The process in equation 32 is called "probability leakage", because the transition matrix $\Pr\left(y^{k+1}|y'^{k+1}\right)$ leaks probability from index y'^{k+1} to index y^{k+1}. The application of this idea to Kohonen self-organising maps will be discussed in 5.

The above least upper bound approach can be extended to minimising the overall objective function as follows:

1. Minimise the upper bound on $D_{FMC}^{0,1}$ by introducing probability leakage into D_{FMC}^{0}.
2. Then minimise $D_{FMC}^{1,2}$ by introducing probability leakage into D_{FMC}^{1}.
3. Etc.
4. Then minimise $D_{FMC}^{L-2,L-1}$ by introducing probability leakage into D_{FMC}^{L-2}.
5. Then minimise D_{FMC}^{L-1}. No probability leakage occurs in network layer L, because it is the final layer.

3.5 Coding The Output Layer

The general expression for $L\left(\mathbf{P},\mathbf{Q}\right)$ in equation 16 is the sum of two terms: an FMC-ladder $\sum_{l=0}^{L-1}\sum_{i_l=1}^{M_l} P_{i_l}^{l} K_{i_l}\left(\mathbf{P}^{l+1|l},\mathbf{Q}^{l|l+1}\right)$, plus the cost $L\left(\mathbf{P}^{L},\mathbf{Q}^{L}\right)$ of coding layer L. The $L\left(\mathbf{P}^{L},\mathbf{Q}^{L}\right)$ term has precisely the form that is commonly used in density modelling, so any convenient density model could be used to parameterise $\mathbf{Q}^{L}$ in layer L.

A typical implementation of the type of network that minimises $L\left(\mathbf{P},\mathbf{Q}\right)$ thus splits into two pieces corresponding to the two different types of term in the objective function. The input space (i.e. layer 0) is connected to the output space (i.e. layer L) by an FMC-ladder corresponding to the

$\sum_{l=0}^{L-1} \sum_{i_l=1}^{M_l} P_{i_l}^l K_{i_l} \left(\mathbf{P}^{l+1|l}, \mathbf{Q}^{l|l+1} \right)$ term. In the special case where $L = 0$ (i.e. no FMC-ladder is used) this approach reduces to standard input density modelling. There are various ways in which $\mathbf{Q}^L$ can be computed, all of which are special cases of the Markov random field (MRF) density modelling approach (all finite connectivity density models are MRFs). There are various possibilities for this MRF model:

1. Boltzmann machine (BM). This is the most general type of MRF model (assuming that the restriction of the classical BM to binary variables and quadratic interactions is not imposed), and can be computationally very expensive. This includes all the well-known image modelling MRF approaches, with or without hidden variables.

2. Hopfield network. This is a zero temperature BM, which is computationally cheaper than a finite temperature BM, but is correspondingly less powerful. This is discussed in section 4.

3. Helmholtz machine (HM). This approach defines an upper bound on $L\left(\mathbf{P}^L, \mathbf{Q}^L\right)$, which reduces the computational load that would otherwise occur if a BM were used.

4 Two Types of Density Model

This section discusses the relationship between two types of density model. The first type is the conventional density model that aims to approximate the input probability density (i.e. the objective function is $L\left(\mathbf{P}^0, \mathbf{Q}^0\right)$), and the second type is the one introduced here which aims to approximate the joint probability density of a Markov source (i.e. the objective function is $L\left(\mathbf{P}, \mathbf{Q}\right)$). In order to relate $L\left(\mathbf{P}^0, \mathbf{Q}^0\right)$ to $L\left(\mathbf{P}, \mathbf{Q}\right)$ it is necessary to introduce additional layers (i.e. layers $1, 2, \cdots, L$) into $L\left(\mathbf{P}^0, \mathbf{Q}^0\right)$ in an appropriate fashion. The Helmholtz machine (HM) [5, 6, 2, 3] does this by replacing $L\left(\mathbf{P}^0, \mathbf{Q}^0\right)$ by a different objective function (which has these additional layers present as hidden variables), and which is an upper bound on the original objective function $L\left(\mathbf{P}^0, \mathbf{Q}^0\right)$. It turns out that Helmholtz machine (HM) objective function (which is generally written as D_{HM}) and the folded Markov chain (FMC) objective function (which is generally written as D_{FMC}) introduced here are closely related. The essential difference between the two is that the Helmholtz machine objective function does does *not* include the cost of specifying the state of layers $1, 2, \cdots, L$ given that

the state of layer 0 is known (this is known as the "bits-back" term), which thus allows it to develop distributed codes (which are expensive to specify) more easily. It is not clear whether the Helmholtz machine objective function is the best approach to distributed codes, because there are other ways of encouraging distributed codes to develop.

4.1 FMC versus Helmholtz Machine

In the conventional density modelling approach to neural networks, there are two basic classes of model. In the case of both unsupervised and supervised neural networks the source is $\mathbf{P}^0$, which is the network input (unsupervised case) or the network output (supervised case). Additionally, in the case of supervised neural networks $\mathbf{P}^0$ is conditioned on the network input as $\mathbf{P}^{0|\text{input}}$. Thus in both cases there is only an external source (i.e. source layers $1, 2, \cdots, L$ are not present), which is modelled by $\mathbf{Q}^0$ (unsupervised case) or $\mathbf{Q}^{0|\text{input}}$ (supervised case). $\mathbf{Q}^0$ or $\mathbf{Q}^{0|\text{input}}$ can be modelled in any way that is convenient. Frequently a multilayer generative model of the form

$$Q^0_{i_0} = \sum_{i_1, i_2, \cdots, i_L} Q^{0|1}_{i_0, i_1} \cdots Q^{l|l+1}_{i_l, i_{l+1}} \cdots Q^L_{i_L} \tag{33}$$

is used, where the i_l (for $1 \leq l \leq L$) are hidden variables, which need to be summed over in order to calculate the required marginal probability $Q^0_{i_0}$, and the notation is deliberately chosen to be the same as is used in the Markov chain model

$$Q_{i_0, i_1, \cdots, i_L} = Q^{0|1}_{i_0, i_1} \cdots Q^{l|l+1}_{i_l, i_{l+1}} \cdots Q^L_{i_L} \tag{34}$$

Helmholtz machines and FMCs are related to each other. Thus the $L\left(\mathbf{P}^0, \mathbf{Q}^0\right)$ that is minimised in conventional density modelling can be manipulated in order to derive D_{HM}

$$\begin{aligned} L\left(\mathbf{P}^0, \mathbf{Q}^0\right) &\leq L\left(\mathbf{P}^0, \mathbf{Q}^0\right) + \sum_{i_0=1}^{M_0} P^0_{i_0} G_{i_0}\left(\mathbf{P}^{1|0}, \mathbf{Q}^{1|0}\right) \\ &= -\sum_{i_0=1}^{M_0} P^0_{i_0} \sum_{i_1=1}^{M_1} P^{1|0}_{i_1, i_0} \log\left(Q^0_{i_0} Q^{1|0}_{i_1, i_0}\right) + \sum_{i_0=1}^{M_0} P^0_{i_0} \sum_{i_1=1}^{M_1} P^{1|0}_{i_1, i_0} \log P^{1|0}_{i_1, i_0} \\ &= L\left(\left(\mathbf{P}^0, \mathbf{P}^{1|0}\right), \left(\mathbf{Q}^0, \mathbf{Q}^{1|0}\right)\right) - \sum_{i_0=1}^{M_0} P^0_{i_0} H_{i_0}\left(\mathbf{P}^{1|0}\right) \end{aligned}$$

$$
\begin{aligned}
&= L\left(\mathbf{P},\mathbf{Q}\right) - \sum_{i_0=1}^{M_0} P_{i_0}^0 H_{i_0}\left(\mathbf{P}^{1|0}\right) \\
&\equiv D_{HM} \qquad (35)
\end{aligned}
$$

where the inequality follows from $G_{i_0}\left(\mathbf{P}^{1|0},\mathbf{Q}^{1|0}\right) \geq 0$. Thus the objective function $L\left(\mathbf{P}^0,\mathbf{Q}^0\right)$ for conventional density modelling is bounded above by the objective function D_{HM} for optimising a 2-layer HM with one hidden layer. The $-\sum_{i_0=1}^{M_0} P_{i_0}^0 H_{i_0}\left(\mathbf{P}^{1|0}\right)$ term is minus the entropy of layer 1 given that layer 0 is known (averaged over layer 0); this is the "bits-back" term of D_{HM}. The $L\left(\mathbf{P},\mathbf{Q}\right)$ term is the standard Markov source objective function (see equation 17), which may be be rearranged in order to make contact with the FMC approach as in section 3.2. Thus

$$
\begin{aligned}
D_{HM} &\leq L\left(\mathbf{P},\mathbf{Q}\right) \\
&= \frac{1}{4\sigma^2} D_{FMC} - \log\frac{V}{\sqrt{2\pi}\sigma} + L\left(\mathbf{P}^1,\mathbf{Q}^1\right) \qquad (36)
\end{aligned}
$$

where the inequality follows from $H_{i_0}\left(\mathbf{P}^{1|0}\right) \geq 0$.

Two inequalities $L\left(\mathbf{P}^0,\mathbf{Q}^0\right) \leq D_{HM} \leq L\left(\mathbf{P},\mathbf{Q}\right)$ are used in the above derivation. $L\left(\mathbf{P}^0,\mathbf{Q}^0\right) \leq D_{HM}$ arises because $\sum_{i_0=1}^{M_0} P_{i_0}^0 G_{i_0}\left(\mathbf{P}^{1|0},\mathbf{Q}^{1|0}\right) \geq 0$ (i.e. the model $\mathbf{Q}^{1|0}$ is imperfect, so that $\mathbf{Q}^{1|0} \neq \mathbf{P}^{1|0}$), and $D_{HM} \leq L\left(\mathbf{P},\mathbf{Q}\right)$ arises because $\sum_{i_0=1}^{M_0} P_{i_0}^0 H_{i_0}\left(\mathbf{P}^{1|0}\right) \geq 0$ (i.e. the source $\mathbf{P}^{1|0}$ is stochastic). If the model is perfect ($\mathbf{Q}^{1|0} = \mathbf{P}^{1|0}$) and the source is deterministic ($\mathbf{P}^{1|0}$ is such that the state of layer 1 is known once the state of layer 0 is given), then the pair of inequalities reduces to $L\left(\mathbf{P}^0,\mathbf{Q}^0\right) = L\left(\mathbf{P},\mathbf{Q}\right)$. In this special case the objective function for optimising a density model in layer 0 is the same as for optimising the corresponding joint density model in layers 0 and 1.

Now rewrite the expression for D_{HM} in order to see how it acts to disrupt the "sparse coding" property of D_{FMC}.

$$
D_{HM} = \sum_{i_0=1}^{M_0} P_{i_0}^0 K_{i_0}\left(\mathbf{P}^{1|0},\mathbf{Q}^{0|1}\right) + \sum_{i_0=1}^{M_0} P_{i_0}^0 G_{i_0}\left(\mathbf{P}^{1|0},\mathbf{Q}^1\right) \qquad (37)
$$

The $\sum_{i_0=1}^{M_0} P_{i_0}^0 K\left(\mathbf{P}^{1|0},\mathbf{Q}^{0|1}\right)$ part (i.e. the pure 2-layer FMC, which is equal to $\frac{1}{4\sigma^2} D_{FMC} - \log\frac{V}{\sqrt{2\pi}\sigma}$ if $\mathbf{Q}^{0|1}$ is Gaussian) and the $\sum_{i_0=1}^{M_0} P_{i_0}^0 G\left(\mathbf{P}^{1|0},\mathbf{Q}^1\right)$ part compete with each other when the 2-layer HM objective function is minimised. Assuming that $P_{i_0}^0 > 0$, the $\sum_{i_0=1}^{M_0} P_{i_0}^0 G\left(\mathbf{P}^{1|0},\mathbf{Q}^1\right)$ part likes to

make $\mathbf{Q}^1$ approximate $\mathbf{P}^{1|0}$, because this reduces $G\left(\mathbf{P}^{1|0}, \mathbf{Q}^1\right)$. On the other hand, assuming that $P_{i_1}^1 > 0$, the $\sum_{i_0=1}^{M_0} P_{i_0}^0 K\left(\mathbf{P}^{1|0}, \mathbf{Q}^{0|1}\right)$ part likes to make $\mathbf{Q}^{0|1}$ approximate $\mathbf{P}^{0|1}$, because this reduces $\sum_{i_0=1}^{M_0} P_{i_0}^0 K\left(\mathbf{P}^{1|0}, \mathbf{Q}^{0|1}\right)$. These two conditions cannot be met simultaneously, because $\sum_{i_0=1}^{M_0} P_{i_0}^0 G\left(\mathbf{P}^{1|0}, \mathbf{Q}^1\right)$ acts to make $\mathbf{P}^{1|0}$ "spread out" to become like $\mathbf{Q}^1$, whereas $\sum_{i_0=1}^{M_0} P_{i_0}^0 K\left(\mathbf{P}^{1|0}, \mathbf{Q}^{0|1}\right)$ acts to make $\mathbf{P}^{1|0}$ a "sparse coder".

The above derivation can be generalised to an $(L+1)$-layer network. Firstly, the 2-layer result can be written as

$$\begin{array}{lll} L\left(\mathbf{P}^0, \mathbf{Q}^0\right) & \leq L\left(\left(\mathbf{P}^0, \mathbf{P}^{1|0}\right), \left(\mathbf{Q}^0, \mathbf{Q}^{1|0}\right)\right) - \sum_{i_0=1}^{M_0} P_{i_0}^0 H_{i_0}\left(\mathbf{P}^{1|0}\right) & \text{2-layer HM} \\ & = L\left(\mathbf{P}, \mathbf{Q}\right) - \sum_{i_0=1}^{M_0} P_{i_0}^0 H_{i_0}\left(\mathbf{P}^{1|0}\right) & \\ & \leq \sum_{i_0=1}^{M_0} P_{i_0}^0 K_{i_0}\left(\mathbf{P}^{1|0}, \mathbf{Q}^{0|1}\right) & \text{2-layer FMC} \\ & + L\left(\mathbf{P}^1, \mathbf{Q}^1\right) & + L\left(\mathbf{P}^1, \mathbf{Q}^1\right) \end{array} \tag{38}$$

which may be generalised to

$$\begin{array}{lll} L\left(\mathbf{P}^0, \mathbf{Q}^0\right) & \leq L\left(\left(\mathbf{P}^0, \mathbf{P}^{1|0}, \cdots, \mathbf{P}^{L|L-1}\right), \left(\mathbf{Q}^0, \mathbf{Q}^{1|0}, \cdots, \mathbf{Q}^{L|L-1}\right)\right) & \\ & - \sum_{l=0}^{L-1} \sum_{i_l=1}^{M_l} P_{i_l}^l H_{i_l}\left(\mathbf{P}^{l+1|l}\right) & \text{multilayer HM} \\ & = L\left(\mathbf{P}, \mathbf{Q}\right) - \sum_{l=0}^{L-1} \sum_{i_l=1}^{M_l} P_{i_l}^l H_{i_l}\left(\mathbf{P}^{l+1|l}\right) & \\ & \leq \sum_{l=0}^{L-1} \sum_{i_l=1}^{M_l} P_{i_l}^l K_{i_l}\left(\mathbf{P}^{l+1|l}, \mathbf{Q}^{l|l+1}\right) & \text{coupled} \\ & + L\left(\mathbf{P}^L, \mathbf{Q}^L\right) & \text{2-layer FMCs} \\ & & + L\left(\mathbf{P}^L, \mathbf{Q}^L\right) \end{array} \tag{3}$$

If $\mathbf{P}^{l+1|l}$ is deterministic (i.e. the state of layer $l+1$ is derived deterministically from the state of layer l), then $\sum_{l=0}^{L-1} \sum_{i_l=1}^{M_l} P_{i_l}^l H\left(\mathbf{P}^{l+1|l}\right) = 0$, in which case D_{HM} simplifies to

$$D_{HM} = \sum_{l=0}^{L-1} \left(\frac{1}{4\left(\sigma^l\right)^2} D_{FMC}^l - \log \frac{V^l}{\sqrt{2\pi}\sigma^l} \right) + L\left(\mathbf{P}^L, \mathbf{Q}^L\right) \tag{40}$$

This highlights an important difference between the FMC and HM approaches: the FMC approach encourages the formation of deterministic $\mathbf{P}^{l+1|l}$ (because this type of $\mathbf{P}^{l+1|l}$ a zero entropy $H\left(\mathbf{P}^{l+1|l}\right)$), whereas the $-\sum_{l=0}^{L-1} \sum_{i_l=1}^{M_l} P_{i_l}^l H\left(\mathbf{P}^{l+1|l}\right)$ term in the HM objective function tries to encourage stochastic $\mathbf{P}^{l+1|l}$ (because this type of $\mathbf{P}^{l+1|l}$ has a large entropy $H\left(\mathbf{P}^{l+1|l}\right)$). Thus the multilayer network produced by optimising a set of

coupled FMCs tends to have a minimal amount of stochastic behaviour. This is another way of saying that the FMC approach naturally leads to sparse codes.

As in the case of a 2-layer network, if the source is deterministic and the model is perfect, then $L\left(\mathbf{P}^0, \mathbf{Q}^0\right) = L\left(\mathbf{P}, \mathbf{Q}\right)$, so that input density optimisation is equivalent to joint density optimisation.

4.2 Alternative Viewpoints

The relationship between conventional density models and FMCs can be stated from the point of view of a conventional density modeller. The goal is to build a density model $\mathbf{Q}^0$ of the source $\mathbf{P}^0$, such that the number of bits per symbol $L\left(\mathbf{P}^0, \mathbf{Q}^0\right)$ required to encode $\mathbf{P}^0$ is minimised. However, if the source $\mathbf{P}^0$ is transformed through L layers of a network to produce a transformed source $\mathbf{P}^L$, then $L\left(\mathbf{P}^0, \mathbf{Q}^0\right)$ is bounded above by the expression $\sum_{l=0}^{L-1} \sum_{i_{l+1}=1}^{M_{l+1}} P_{i_{l+1}}^{l+1} L_{i_{l+1}} \left(\mathbf{P}^{l|l+1}, \mathbf{Q}^{l|l+1}\right) + L\left(\mathbf{P}^L, \mathbf{Q}^L\right)$, which is the sum of the number of bits per symbol $L\left(\mathbf{P}^L, \mathbf{Q}^L\right)$ required to encode $\mathbf{P}^L$, plus (for $l = 0, 1, \cdots, L-1$) the number of bits per symbol $\sum_{i_{l+1}=1}^{M_{l+1}} P_{i_{l+1}}^{l+1} L_{i_{l+1}} \left(\mathbf{P}^{l|l+1}, \mathbf{Q}^{l|l+1}\right)$ required to encode $\mathbf{P}^{l|l+1}$ (this is a set of coupled FMCs). The relationship $\sum_{i_l=1}^{M_l} P_{i_l}^{l} K_{i_l} \left(\mathbf{P}^{l+1|l}, \mathbf{Q}^{l|l+1}\right) = \sum_{i_{l+1}=1}^{M_{l+1}} P_{i_{l+1}}^{l+1} L_{i_{l+1}} \left(\mathbf{P}^{l|l+1}, \mathbf{Q}^{l|l+1}\right)$ was used to obtain this interpretation. Thus the problem of encoding the source $\mathbf{P}^0$ can be split into three steps: transform the source from $\mathbf{P}^0$ to $\mathbf{P}^L$, encode the transformed source $\mathbf{P}^L$, and encode all of the transformations $\mathbf{P}^{l|l+1}$ (for $l = 0, 1, \cdots, L-1$) to allow the original source to be reconstructed from the transformed source. The total number of bits required to encode $\mathbf{P}^L$ and $\mathbf{P}^{l|l+1}$ (for $l = 0, 1, \cdots, L-1$) is then an upper bound on the total number of bits required to encode $\mathbf{P}^0$. In this picture, coupled FMCs are used to connect the original source $\mathbf{P}^0$ to the transformed source $\mathbf{P}^L$, so they connect one conventional density modelling problem (i.e. optimising $\mathbf{Q}^0$) to another (i.e. optimising $\mathbf{Q}^L$).

The above description of the relationship between conventional density models and FMCs was presented from the point of view of a conventional density modeller, who asserts that the goal is to build an optimum (i.e. minimum number of bits per symbol) density model $\mathbf{Q}^0$ of the source $\mathbf{P}^0$. From this point of view, the coupled FMCs are merely a means of transforming the problem from modelling the source $\mathbf{P}^0$ to modelling the transformed source $\mathbf{P}^L$. That this process is imperfect is reflected in the fact that more bits per

symbol are required to encode the transformed source $\mathbf{P}^L$ (plus the coupled FMCs leading to it) than the original source $\mathbf{P}^0$. A conventional density modeller might reasonably ask what is the point of using FMCs, if they give only an upper bound on the number of bits per symbol for encoding the original source $\mathbf{P}^0$. However, it is not at all clear that the conventional density modeller is using the correct objective function in the first place. Why should the number of bits per symbol for encoding the original source $\mathbf{P}^0$ be especially important? It is as if the world has been separated into an external world (i.e. $\mathbf{P}^0$) and an internal world (i.e. the $\mathbf{P}^{l+1|l}$ for $l = 0, 1, \cdots, L-1$), and a special status is accorded to the external world, which deems that it is important to model its density $\mathbf{P}^0$ accurately, at the expense of modelling the $\mathbf{P}^{l+1|l}$ accurately. In the FMC approach, this artificial boundary between external and internal worlds is removed, because the coupled FMCs model the joint density $\left(\mathbf{P}^0, \mathbf{P}^{1|0}, \cdots, \mathbf{P}^{L-1|L-2}, \mathbf{P}^{L|L-1}\right)$, where $\mathbf{P}^0$ and the $\mathbf{P}^{l+1|l}$ are all accorded equal status. This even-handed approach is much more natural than one in which a particular part of the source (i.e. the external source) is accorded a special status.

In the language of multilayer neural networks, the $\left(\mathbf{P}^0, \mathbf{P}^{0|1}, \cdots, \mathbf{P}^{L-1|L-2}, \mathbf{P}^{L|L-1}\right)$ is the source which comprises the bottom-up transformations (or recognition models) which generate the states of the internal layers of the network, and the $\left(\mathbf{Q}^{0|1}, \mathbf{Q}^{1|2}, \cdots, \mathbf{Q}^{L-1|L}, \mathbf{Q}^L\right)$ is the model of the source which comprises the top-down transformations (or generative models). Thus the network is self-referential, because it forms a model of a source that includes its own internal states, because part of the source (i.e. the $\mathbf{P}^{l+1|l}$ for $l = 0, 1, \cdots, L-1$) is the state of the network layers. This self-referential behaviour is present in both the conventional density modelling approach and in the FMC approach, but whereas in the former case it is not optimised in an even-handed fashion, in the latter case it is optimised in an even-handed fashion. One could adopt an extreme viewpoint, in which no distinction is made at all between the part of the world that is external to the neural network, and the part that is internal to the neural network. This is a natural approach, because the neural network is itself part of the world, so it should therefore be treated in exactly the same way as the part of the world that is external to the neural network. In effect, the neural network is inevitable; it is the world's way of modelling itself.

5 Kohonen Self-Organising Network

In this section the Kohonen topographic mapping neural network [7] will be derived from the 2-layer FMC objective function including probability leakage. The result is only approximate, because the training algorithm proposed by Kohonen does not correspond to the minimisation of any objective function. However, the objective function approach is useful nevertheless, because it produces very similar results to Kohonen's original proposal, whilst also allowing such neural networks to be treated in a unified way.

5.1 2-Layer Folded Markov Chain Network

In section 3.4 the concept of probability leakage was introduced as an empirical way of modelling the damage that FMCs $k+1, k+2, \cdots, L-1$ do to the output of FMC k in the FMC-ladder. Probability leakage can readily be implemented by the replacement for $\Pr(y|\mathbf{x})$ given in equation 32. In the case of the 2-layer FMC discussed in section 3.2, and in particular the objective function given in equation 21, probability leakage leads to the modified objective function

$$D_{FMC} = 2 \int d\mathbf{x} \Pr(\mathbf{x}) \sum_{y=1}^{M} \sum_{y'=1}^{M} \Pr(y|y') \Pr(y'|\mathbf{x}) \ \left\|\mathbf{x} - \mathbf{x}'(y)\right\|^2 \qquad (41)$$

which corresponds to the objective function for a soft VQ with code noise modelled by $\Pr(y|y')$; this is related to earlier results on vector quantisation for transmission along a noisy communication channel [8, 4]. The corresponding hard VQ (i.e. $\Pr(y|\mathbf{x}) = \delta_{y,y(\mathbf{x})}$) objective function becomes

$$D_{FMC} = 2 \int d\mathbf{x} \Pr(\mathbf{x}) \sum_{y=1}^{M} \Pr(y|y(\mathbf{x})) \ \left\|\mathbf{x} - \mathbf{x}'(y)\right\|^2 \qquad (42)$$

This is a self-organising map (SOM) objective function, because the optimal code vectors $\mathbf{x}'(y)$ must arrange themselves so that not only can they reconstruct the input with small average L^2 distortion, but they must also make the information in the output layer robust with respect to the damage that can be caused by probability leakage. $\Pr(y|y')$ can be identified with the topographic neighbourhood function used by Kohonen in his SOM algorithm. However, minimisation of the objective function in equation 42 does not quite lead to the same training algorithm as specified by Kohonen; the encoding process is no longer a nearest neighbour prescription, but a

minimum distortion prescription (i.e. pick the winning code index as the one that on average will lead to minimum L^2 reconstruction distortion, when the effects of probability leakage have been taken into account. Thus the encoding process anticipates the average effect of probability leakage (which in turn models the damage caused by the higher layers of the FMC-ladder, as discussed in section 3.4).

If one does not mind the slight difference between the training algorithm derived from equation 42 and Kohonen's original algorithm, then this approach supplies a nice interpretation of what the Kohonen algorithm is actually doing. It can be understood only by referring to the type of damage that higher layers of the network are going to do to the output of layer 1.

This approach may be generalised to a multilayer network. If leakage is introduced in each layer then a multilayer Kohonen network is obtained. See [10] for a simple application of multilayer Kohonen networks, and see [11, 12] for an early paper written from the FMC point of view.

6 Partitioned Mixture Distributions (PMD)

This section introduces a useful parameterisation of the conditional probability $\mathbf{P}^{l+1|l}$ for building the Markov source. Because the multilayer network produced by optimising a set of coupled FMCs tends to have a minimal amount of stochastic behaviour (see the end of section 4.1), a way of encouraging $\mathbf{P}^{l+1|l}$ to form distributed codes must be found, so that more that one state in layer $l+1$ can have high probability, given that the state in layer l is known. Different parts of layer $l+1$ could then be used to encode different parts of layer l, which would then allow factorial codes to develop. It turns out that there is a simple way of allowing such codes to develop, called the partitioned mixture distribution (PMD) [18, 20, 22]. A PMD is essentially a large number of mixture distribution models run "in reverse" (i.e. used to compute posterior probabilities over class labels, rather than to compute probability densities in the mixture distribution input space). When these mixture distributions are used to connect together layer l and layer $l+1$, where each mixture distribution is connected to only part of layer l and layer $l+1$, then the resulting network (i.e. a PMD) contains all of the ingredients that are necessary for factorial codes to develop.

6.1 Multiple Recognition Models

In the expression for the objective function

$$L\left(\mathbf{P},\mathbf{Q}\right)=\sum_{l=0}^{L-1}\sum_{i_l=1}^{M_l}P_{i_l}^{l}K_{i_l}\left(\mathbf{P}^{l+1|l},\mathbf{Q}^{l|l+1}\right)+L\left(\mathbf{P}^{L},\mathbf{Q}^{L}\right) \tag{43}$$

the $\sum_{l=0}^{L-1}\sum_{i_l=1}^{M_l}P_{i_l}^{l}K_{i_i}\left(\mathbf{P}^{l+1|l},\mathbf{Q}^{l|l+1}\right)$ term corresponds to a set of coupled FMCs (i.e. an FMC-ladder), in which the $\mathbf{P}^{l+1|l}$ and the $\mathbf{Q}^{l|l+1}$ (for $l=0,1,\cdots,L-1$) need to be constructed. Currently, in an FMC-ladder the generative models $\mathbf{Q}^{l|l+1}$ are parameterised as Gaussian probability densities, whereas the recognition models $\mathbf{P}^{l+1|l}$ are parameterised in a more general way.

The simplest parameterisation of the recognition model $P_{i_{l+1},i_l}^{l+1|l}$ is

$$P_{i_{l+1},i_l}^{l+1|l}=\frac{P_{i_l,i_{l+1}}^{l|l+1}P_{i_{l+1}}^{l+1}}{\sum_{i'_{l+1}=1}^{M_{l+1}}P_{i_l,i'_{l+1}}^{l|l+1}P_{i'_{l+1}}^{l+1}} \tag{44}$$

which guarantees the normalisation condition $\sum_{i_{l+1}=1}^{M_{l+1}}P_{i_{l+1},i_l}^{l+1|l}=1$; this is the posterior probability (of class i_{l+1} given data i_l) derived from a mixture distribution $\sum_{i'_{l+1}=1}^{M_{l+1}}P_{i_l,i'_{l+1}}^{l|l+1}P_{i'_{l+1}}^{l+1}$. A limitation of this type of recognition model is that it allows only a *single* explanation i_{l+1} of the data i_l (in the case of a hard $P_{i_{l+1},i_l}^{l+1|l}$) or a probability distribution over *single* explanations (in the case of a soft $P_{i_{l+1},i_l}^{l+1|l}$), so it cannot lead to a factorial encoding of the data.

The simplest way of allowing a factorial encoding to develop is to make simultaneous use more than one recognition model. Each recognition model uses its own $\mathbf{P}^{l+1}$ vector and $\mathbf{P}^{l|l+1}$ matrix to compute a posterior probability of the type shown in equation 44, so that if each recognition model is sensitised to a different piece of the data, then a factorial code can develop. This approach can be formalised by making the replacement $i_{l+1}\rightarrow\mathbf{i}_{l+1}$ in equation 44 (i.e. replace the scalar code index by a vector code index, where the number of vector components is equal to the number of recognition models). If the components of $\mathbf{i}_{l+1}$ are determined independently of each other, then their joint posterior probability $P_{\mathbf{i}_{l+1},i_l}^{l+1|l}$ is a product of independent posterior probabilities, where each posterior probability corresponds to one of the recognition models, and thus has its own $\mathbf{P}^{l+1}$ vector and $\mathbf{P}^{l|l+1}$ matrix.

If this type of posterior probability, which is a product of n independent factors if there are n independent recognition models, is then inserted into equation 21 (i.e. a 2-layer FMC, or equivalently a soft VQ) it yields

$$\begin{aligned} D_{FMC} &= 2 \int d\mathbf{x} \Pr(\mathbf{x}) \sum_{y_1=1}^{M_1} \sum_{y_2=1}^{M_2} \cdots \sum_{y_n=1}^{M_n} \Pr(y_1|\mathbf{x}, 1) \Pr(y_2|\mathbf{x}, 2) \cdots \Pr(y_n|\mathbf{x}, n) \\ &\quad \times \left\| \mathbf{x} - \mathbf{x}'(y_1, y_2, \cdots, y_n) \right\|^2 \end{aligned} \tag{45}$$

where $\Pr(y_k|\mathbf{x}, k)$ denotes the posterior probability that (given input $\mathbf{x}$) code index y_k occurs in recognition model k, and $\mathbf{x}'(y_1, y_2, \cdots, y_n)$ takes the value that minimises D_{FMC} (i.e. $\mathbf{x}'(y_1, y_2, \cdots, y_n) = \int d\mathbf{x} \Pr(y_1|\mathbf{x}) \Pr(y_2|\mathbf{x}) \cdots \Pr(y_n|\mathbf{x})\ \mathbf{x}$). If the L^2 norm is then expanded thus

$$\left\| \mathbf{x} - \mathbf{x}'(y_1, y_2, \cdots, y_n) \right\|^2 \equiv \left\| \begin{array}{c} \left(\mathbf{x} - \frac{1}{n} \sum_{k=1}^{n} \mathbf{x}'_k(y_k) \right) \\ + \left(\frac{1}{n} \sum_{k=1}^{n} \mathbf{x}'_k(y_k) - \mathbf{x}'(y_1, y_2, \cdots, y_n) \right) \end{array} \right\|^2 \tag{46}$$

then D_{FMC} can be bounded above as follows

$$\begin{aligned} D_{FMC} &\leq 2 \int d\mathbf{x} \Pr(\mathbf{x}) \sum_{y_1=1}^{M_1} \sum_{y_2=1}^{M_2} \cdots \sum_{y_n=1}^{M_n} \Pr(y_1|\mathbf{x}, 1) \Pr(y_2|\mathbf{x}, 2) \cdots \Pr(y_n|\mathbf{x}, n) \\ &\quad \times \left\| \mathbf{x} - \frac{1}{n} \sum_{k=1}^{n} \mathbf{x}'_k(y_k) \right\|^2 \end{aligned} \tag{47}$$

First of all the $\Pr(y_k|\mathbf{x}, k)$ are used to produce soft encodings in each of the recognition models ($k = 1, 2, \cdots, n$), then a sum $\frac{1}{n} \sum_{k=1}^{n} \mathbf{x}'_k(y_k)$ of the vectors $\mathbf{x}'_k(y_k)$ is used as the reconstruction of the input $\mathbf{x}$. In the special case where hard encodings are used, so that $\Pr(y_k|\mathbf{x}, k) = \delta_{y_k, y_k(\mathbf{x})}$, then the upper bound on D_{FMC} reduces to

$$D_{FMC} \leq 2 \int d\mathbf{x} \Pr(\mathbf{x}) \left\| \mathbf{x} - \frac{1}{n} \sum_{k=1}^{n} \mathbf{x}'_k(y_k(\mathbf{x})) \right\|^2 \tag{48}$$

This is related to the objective function for a cooperative vector quantiser (CVQ), where n VQs are used *independently* to encode the input, and then a reconstruction is formed from a sum of vectors. Note that the code vectors used for the encoding operation $y_k(\mathbf{x})$ are not necessarily the same as the $\mathbf{x}'_k(y_k)$, except in the special case $n = 1$.

Suppose that a single recognition model is independently used n times, rather than n independent recognition models each independently being used

once. This corresponds to constraining the $\mathbf{P}^{l+1}$ vectors and $\mathbf{P}^{l|l+1}$ matrices to be the same for each of the n recognition models. The upper bound on D_{FMC} can be manipulated into the form

$$\begin{aligned} D_{FMC} \quad \leq \quad & \frac{2}{n} \int d\mathbf{x} \Pr(\mathbf{x}) \sum_{y=1}^{M} \Pr(y|\mathbf{x}) \left\| \mathbf{x} - \mathbf{x}'(y) \right\|^2 \\ & + \frac{2(n-1)}{n} \int d\mathbf{x} \Pr(\mathbf{x}) \left\| \mathbf{x} - \sum_{y=1}^{M} \Pr(y|\mathbf{x}) \mathbf{x}'(y) \right\|^2 \end{aligned} \tag{49}$$

where the k index is no longer needed. In the case $n = 1$ this correctly reduces to equation 21 (the inequality reduces to an equality in this case). When $n > 1$ the second term offers the possibility of factorial encoding, because it contains a weighted linear combination $\sum_{y=1}^{M} \Pr(y|\mathbf{x}) \mathbf{x}'(y)$ of vectors.

6.2 Average Over Recognition Models

Now combine the above two approaches to factorial encoding, so that a single recognition model is used (as in equation 49), which is parameterised in such a way that it can emulate multiple recognition models (as in equation 47). The simplest possibility is to make the replacement $P_{i_{l+1}}^{l+1} \rightarrow A_{k,i_{l+1}}^{l+1} P_{i_{l+1}}^{l+1}$ (where $A_{k,i_{l+1}}^{l+1} \geq 0$) in equation 44, where k is a recognition model index which ranges over $k = 1, 2, \cdots, K$ (note that K is not constrained to be the same as n), and to average over k, to produce

$$P_{i_{l+1},i_l}^{l+1|l} \rightarrow \frac{1}{K} \sum_{k=1}^{K} \frac{P_{i_l,i_{l+1}}^{l|l+1} A_{k,i_{l+1}}^{l+1} P_{i_{l+1}}^{l+1}}{\sum_{i'_{l+1}=1}^{M_{l+1}} P_{i_l,i'_{l+1}}^{l|l+1} A_{k,i'_{l+1}}^{l+1} P_{i'_{l+1}}^{l+1}} \tag{50}$$

In effect, K recognition models are embedded between layer l and layer $l+1$, and the $\mathbf{A}^{l+1}$ matrix specifies which indices i_{l+1} in layer $l+1$ are associated with recognition model k.

A partitioned mixture distribution (PMD) is precisely this type of multiple embedded recognition model. In the simplest type of PMD the $\mathbf{A}^{l+1}$ matrix is chosen to contain only 0's and 1's, which are arranged so that the K recognition models partition layer $l+1$ into K overlapping patches. Other PMDs are possible. For instance, the $\mathbf{A}^{l+1}$ matrix could specify K recognition models which partition layer $l+1$ into K *non*-overlapping patches. In this case, because the n code indices are generated independently, they

are not guaranteed to occupy different partitions (i.e. different recognition models); some partitions might have no code indices, some might have only one, and others might have more than one, subject only to the constraint that the total number was n. This is not a fundamental problem, because provided that $n \geq K$ there is a finite probability (which increases monotonically towards 1 as $n \to \infty$) that at least one code index occupies each of the K partitions. This is how a single recognition model (when parameterised as a PMD given in equation 50) can emulate multiple recognition models.

As in the Kohonen network (see section 5) probability leakage (see section 3.4) can be used to anticipate the damage that higher layers of a multilayer network cause, by encouraging the network properties to become topographically ordered.

6.3 Full Bayesian Average Over Recognition Models

One possible criticism of the recognition model given in equation 50 is that it is a mixture of K recognition models, where each contributing model is assigned the *same* weight $\frac{1}{K}$. Normally, a posterior probability $\Pr(y|\mathbf{x})$ is decomposed as a sum over contributing model posterior probabilities $\Pr(y|\mathbf{x}, k)$ as follows

$$\Pr(y|\mathbf{x}) = \sum_{k=1}^{K} \Pr(y|\mathbf{x}, k) \Pr(k|\mathbf{x}) \tag{51}$$

where each of the K recognition models is assigned a *different* data-dependent weight $\Pr(k|\mathbf{x})$. The conditional probabilities $\Pr(k|\mathbf{x})$ and $\Pr(y|\mathbf{x})$ can be evaluated to yield

$$\begin{aligned} \Pr(k|\mathbf{x}) &= \frac{\sum_{y=1}^{M} \Pr(\mathbf{x}|y, k) \Pr(y|k) \Pr(k)}{\sum_{k'=1}^{K} \sum_{y'=1}^{M} \Pr(\mathbf{x}|y', k') \Pr(y'|k') \Pr(k')} \\ \Pr(y|\mathbf{x}, k) &= \frac{\Pr(\mathbf{x}|y, k) \Pr(y|k) \Pr(k)}{\sum_{y'=1}^{M} \Pr(\mathbf{x}|y', k) \Pr(y'|k) \Pr(k)} \end{aligned} \tag{52}$$

so that

$$\Pr(y|\mathbf{x}) = \sum_{k=1}^{K} \frac{\Pr(\mathbf{x}|y, k) \Pr(y|k) \Pr(k)}{\sum_{k'=1}^{K} \sum_{y'=1}^{M} \Pr(\mathbf{x}|y', k') \Pr(y'|k) \Pr(k')} \tag{53}$$

If the replacements $\Pr(k) \rightarrow 1$ and $\Pr(y|k) \rightarrow A^{l+1}_{k,i_{l+1}} P^{l+1}_{i_{l+1}}$ (both of these modulo a constant factor), and $\Pr(\mathbf{x}|y,k) \rightarrow P^{l|l+1}_{i_l,i_{l+1}}$, then

$$\Pr(y|\mathbf{x}) \rightarrow \sum_{k=1}^{K} \frac{P^{l|l+1}_{i_l,i_{l+1}} A^{l+1}_{k,i_{l+1}} P^{l+1}_{i_{l+1}}}{\sum_{k'=1}^{K} \sum_{i'_{l+1}=1}^{M_{l+1}} P^{l|l+1}_{i_l,i'_{l+1}} A^{l+1}_{k',i'_{l+1}} P^{l+1}_{i'_{l+1}}} \tag{54}$$

which is *not* the same as the PMD recognition model in equation 50, which would have been obtained if k were independent of $\mathbf{x}$ (i.e. $\Pr(k|\mathbf{x}) = \Pr(k)$).

However, the PMD recognition model has a strong advantage over the full recognition model in equation 54, because it uses only local connectivity in layer $l+1$, which determines the contributions to the sums over i_{l+1} and k. In equation 54 the normalisation term in the denominator has a double summation $\sum_{k=1}^{K} \sum_{i_{l+1}=1}^{M_{l+1}} P^{l|l+1}_{i_l,i_{l+1}} A^{l+1}_{k,i_{l+1}} P^{l+1}_{i_{l+1}}$, which involves all pairs of indices k and i_{l+1} which have $A^{l+1}_{k,i_{l+1}} > 0$, which thus corresponds to long-range lateral interactions in layer $l+1$. On the other hand, the PMD recognition model in equation 50 has a normalisation term in the denominator which involves only a single summation $\sum_{i_{l+1}=1}^{M_{l+1}} P^{l|l+1}_{i_l,i_{l+1}} A^{l+1}_{k,i_{l+1}} P^{l+1}_{i'_{l+1}}$, so the lateral interactions in layer $l+1$ are determined by the structure of the matrix $A^{l+1}_{k,i_{l+1}}$, which defines only short-range lateral connections (i.e. for a given recognition model k, only a limited number of index values i_{l+1} satisfy $A^{l+1}_{k,i_{l+1}} > 0$.

As discussed in section 2.4, a PMD can be endowed with a memory of its previous state, just as a standard mixture distribution can, to obtain a dynamical PMD [22].

7 Adaptive Cluster Expansion (ACE)

This section discusses the adaptive cluster expansion (ACE) [14, 19, 21], which is a tree-structured density network.

7.1 ACE: Tree-Structured Density Network

Consider the objective function D for an $L+1$ layer FMC-ladder

$$D \equiv \sum_{l=0}^{L-1} \sum_{i_l=1}^{M_l} P^l_{i_l} K_{i_l}\left(\mathbf{P}^{l+1|l}, \mathbf{Q}^{l|l+1}\right)$$

$$= -\sum_{l=0}^{L-1}\sum_{i_l=1}^{M_l} P_{i_l}^{l}\sum_{i_{l+1}=1}^{M_{l+1}} P_{i_{l+1},i_l}^{l+1|l}\,\log Q_{i_l,i_{l+1}}^{l|l+1} \tag{55}$$

Now assume that the $Q_{i_l,i_{l+1}}^{l|l+1}$ part of the model (i.e. the Markovian part) is perfect so that $Q_{i_l,i_{l+1}}^{l|l+1} = P_{i_l,i_{l+1}}^{l|l+1}$ (for $l = 0, 1, \cdots, L-1$), and that the $P_{i_{l+1},i_l}^{l+1|l}$ part of the source (i.e. the Markovian part) is deterministic so that $P_{i_{l+1},i_l}^{l+1|l} = \delta_{i_{l+1},i_{l+1}(i_l)}$ (for $l = 0, 1, \cdots, L-1$), in which case D simplifies as follows

$$\begin{aligned}
D &= -\sum_{l=0}^{L-1}\sum_{i_l=1}^{M_l} P_{i_l}^{l}\sum_{i_{l+1}=1}^{M_{l+1}} P_{i_{l+1},i_l}^{l+1|l}\,\log P_{i_l,i_{l+1}}^{l|l+1} \\
&= -\sum_{l=0}^{L-1}\sum_{i_l=1}^{M_l} P_{i_l}^{l}\sum_{i_{l+1}=1}^{M_{l+1}} \delta_{i_{l+1},i_{l+1}(i_l)}\,\log\frac{\delta_{i_{l+1},i_{l+1}(i_l)}P_{i_l}^{l}}{P_{i_{l+1}}^{l+1}} \\
&= -\sum_{l=0}^{L-1}\sum_{i_l=1}^{M_l} P_{i_l}^{l}\,\log P_{i_l}^{l} + \sum_{l=0}^{L-1}\sum_{i_l=1}^{M_l} P_{i_l}^{l}\sum_{i_{l+1}=1}^{M_{l+1}} \delta_{i_{l+1},i_{l+1}(i_l)}\log P_{i_{l+1}}^{l+1} \\
&= -\sum_{l=0}^{L-1}\sum_{i_l=1}^{M_l} P_{i_l}^{l}\,\log P_{i_l}^{l} + \sum_{l=0}^{L-1}\sum_{i_{l+1}=1}^{M_{l+1}} P_{i_{l+1}}^{l+1}\log P_{i_{l+1}}^{l+1} \\
&= -\sum_{i_0=1}^{M_0} P_{i_0}^{0}\,\log P_{i_0}^{0} + \sum_{i_L=1}^{M_L} P_{i_L}^{L}\log P_{i_L}^{L} \\
&= H\left(\mathbf{P}^0\right) - H\left(\mathbf{P}^L\right)
\end{aligned} \tag{56}$$

This is the number of bits per symbol that is required to convert a $\mathbf{P}^L$-message into a $\mathbf{P}^0$-message, assuming that the Markovian part of the source is deterministic, and that the model is perfect. This result is not very interesting in itself.

However, if the Markovian part of the source is not only deterministic, but is also tree-structured, and the model is similarly tree-structured, then the notation must be modified thus

$$\begin{aligned}
i_l &\rightarrow \mathbf{i}_l = \left(\mathbf{i}_l^1, \mathbf{i}_l^2, \cdots\right) \\
i_{l+1} &\rightarrow \mathbf{i}_{l+1} = \left(i_{l+1}^1, i_{l+1}^2, \cdots\right) \\
P_{i_{l+1},i_l}^{l+1|l} &\rightarrow P_{\mathbf{i}_{l+1},\mathbf{i}_l}^{l+1|l} = P_{i_{l+1}^1,\mathbf{i}_l^1}^{l+1|l} P_{i_{l+1}^2,\mathbf{i}_l^2}^{l+1|l}\cdots = \delta_{i_{l+1}^1,i_{l+1}^1\left(\mathbf{i}_l^1\right)}\delta_{i_{l+1}^2,i_{l+1}^2\left(\mathbf{i}_l^2\right)}\cdots
\end{aligned}$$

$$Q^{l|l+1}_{i_l,i_{l+1}} \quad \rightarrow \quad Q^{l|l+1}_{\mathbf{i}_l,\mathbf{i}_{l+1}} = P^{l|l+1}_{\mathbf{i}^1_l,i^1_{l+1}} P^{l|l+1}_{\mathbf{i}^2_l,i^2_{l+1}} \cdots \tag{57}$$

where the components of the vector $\mathbf{i}_l$ have been partitioned as $(\mathbf{i}^1_l, \mathbf{i}^2_l, \cdots)$, where each $\mathbf{i}^c_l$ contains a complete set of siblings whose parent is in layer $l+1$, and the components of the vector $\mathbf{i}_{l+1}$ have been partitioned as $\left(i^1_{l+1}, i^2_{l+1}, \cdots\right)$, where i^c_{l+1} is the parent of $\mathbf{i}^c_l$. This notation may be used to rearrange D as follows

$$\begin{aligned}
D &= -\sum_{l=0}^{L-1} \sum_{\mathbf{i}_l} P^l_{\mathbf{i}_l} \sum_{\mathbf{i}_{l+1}} P^{l+1|l}_{\mathbf{i}_{l+1},\mathbf{i}_l} \log \left(P^{l|l+1}_{\mathbf{i}^1_l,i^1_{l+1}} P^{l|l+1}_{\mathbf{i}^2_l,i^2_{l+1}} \cdots \right) \\
&= -\sum_{l=0}^{L-1} \sum_{\mathbf{i}_l} P^l_{\mathbf{i}_l} \sum_{\mathbf{i}_{l+1}} P^{l+1|l}_{\mathbf{i}_{l+1},\mathbf{i}_l} \sum_c \log P^{l|l+1}_{\mathbf{i}^c_l,i^c_{l+1}} \\
&= -\sum_{l=0}^{L-1} \sum_{\mathbf{i}_l} P^l_{\mathbf{i}_l} \sum_{\mathbf{i}_{l+1}} P^{l+1|l}_{\mathbf{i}_{l+1},\mathbf{i}_l} \sum_c \log \left(\frac{P^{l+1|l}_{i^c_{l+1},\mathbf{i}^c_l} P^l_{\mathbf{i}^c_l}}{P^{l+1}_{i^c_{l+1}}} \right) \\
&= -\sum_{l=0}^{L-1} \sum_{\mathbf{i}_l} P^l_{\mathbf{i}_l} \sum_c \log P^l_{\mathbf{i}^c_l} \\
&\quad -\sum_{l=0}^{L-1} \sum_{\mathbf{i}_l} P^l_{\mathbf{i}_l} \sum_{\mathbf{i}_{l+1}} P^{l+1|l}_{\mathbf{i}_{l+1},\mathbf{i}_l} \sum_c \log P^{l+1|l}_{i^c_{l+1},\mathbf{i}^c_l} \\
&\quad +\sum_{l=0}^{L-1} \sum_{\mathbf{i}_l} P^l_{\mathbf{i}_l} \sum_{\mathbf{i}_{l+1}} P^{l+1|l}_{\mathbf{i}_{l+1},\mathbf{i}_l} \sum_c \log P^{l+1}_{i^c_{l+1}} \\
&= -\sum_{l=0}^{L-1} \sum_{\mathbf{i}_l} P^l_{\mathbf{i}_l} \sum_c \log P^l_{\mathbf{i}^c_l} + \sum_{l=0}^{L-1} \sum_{\mathbf{i}_{l+1}} P^{l+1}_{\mathbf{i}_{l+1}} \sum_c \log P^{l+1}_{i^c_{l+1}} \\
&= \sum_{l=0}^{L-1} \sum_{\text{cluster } c} H\left(\mathbf{P}^l_c\right) - \sum_{l=1}^{L} \sum_{\text{component } c} H\left(\mathbf{P}^l_c\right)
\end{aligned} \tag{58}$$

where the fact that $-\sum_{l=0}^{L-1} \sum_{\mathbf{i}_l} P^l_{\mathbf{i}_l} \sum_{\mathbf{i}_{l+1}} P^{l+1|l}_{\mathbf{i}_{l+1},\mathbf{i}_l} \sum_c \log P^{l+1|l}_{i^c_{l+1},\mathbf{i}^c_l} = 0$ has been used.

This expression for D can be rewritten in terms of the mutual information $I\left(\mathbf{P}^l_c\right)$ between the components of cluster $\mathbf{i}^c_{l+1}$ by making use of the following result

$$\sum_{\text{cluster } c} I\left(\mathbf{P}^l_c\right) \quad \equiv \quad \sum_{\text{component } c} H\left(\mathbf{P}^l_c\right) - \sum_{\text{cluster } c} H\left(\mathbf{P}^l_c\right)$$

$$= -\sum_{\text{component } c} P^l_{i^c_l} \log P^l_{i^c_l} + \sum_{\text{cluster } c} \sum_{\mathbf{i}^c_l} P^l_{\mathbf{i}^c_l} \log P^l_{\mathbf{i}^c_l} \quad (59)$$

to yield

$$\begin{aligned} D &= \sum_{\text{cluster } c} H\left(\mathbf{P}^0_c\right) - \sum_{\text{cluster } c} H\left(\mathbf{P}^L_c\right) \\ &\quad + \sum_{l=1}^{L} \sum_{\text{cluster } c} H\left(\mathbf{P}^l_c\right) - \sum_{l=1}^{L} \sum_{\text{component } c} H\left(\mathbf{P}^l_c\right) \\ &= -\sum_{l=1}^{L} \sum_{\text{cluster } c} I\left(\mathbf{P}^l_c\right) + \sum_{\text{cluster } c} H\left(\mathbf{P}^0_c\right) - \sum_{\text{cluster } c} H\left(\mathbf{P}^L_c\right) \quad (60) \end{aligned}$$

Now add $L\left(\mathbf{P}^L, \mathbf{Q}^L\right)$ (the contribution from the output layer) to D (the objective function for an FMC-ladder) to obtain $L\left(\mathbf{P}, \mathbf{Q}\right)$, and assume that the model is perfect in the output layer so that $\mathbf{Q}^L$ is given by $Q^L_{\mathbf{i}_L} = P^L_{\mathbf{i}^1_l} P^L_{\mathbf{i}^2_l} \cdots$. This allows $L\left(\mathbf{P}^L, \mathbf{Q}^L\right)$ to be simplified to $L\left(\mathbf{P}^L, \mathbf{Q}^L\right) = \sum_{\text{cluster } c} H\left(\mathbf{P}^L_c\right)$, so that $L\left(\mathbf{P}, \mathbf{Q}\right)$ simplifies as [14]

$$\begin{aligned} L\left(\mathbf{P}, \mathbf{Q}\right) &= D + L\left(\mathbf{P}^L, \mathbf{Q}^L\right) \\ &= -\sum_{l=1}^{L} \sum_{\text{cluster } c} I\left(\mathbf{P}^l_c\right) + \sum_{\text{cluster } c} H\left(\mathbf{P}^0_c\right) \quad (61) \end{aligned}$$

The $-\sum_{l=1}^{L} \sum_{\text{cluster } c} I\left(\mathbf{P}^l_c\right)$ term is (minus) the sum of the mutual informations within all of the clusters in the $L+1$ layer network, and the $\sum_{\text{cluster } c} H\left(\mathbf{P}^0_c\right)$ term is constant for a given external source $\mathbf{P}^0$. This means that minimising $L\left(\mathbf{P}, \mathbf{Q}\right)$ is equivalent to maximising $\sum_{l=1}^{L} \sum_{\text{cluster } c} I\left(\mathbf{P}^l_c\right)$. This is the maximum mutual information result for ACE networks. The mutual information maximisation principle in [1] is a special case of the above result.

As was noted in section 4.1, if the source is deterministic and the model is perfect (as they are here), then $L\left(\mathbf{P}^0, \mathbf{Q}^0\right) = L\left(\mathbf{P}, \mathbf{Q}\right)$, which implies that input density optimisation is equivalent to joint density optimisation. This equivalence was used in [14], where the sum-of-mutual-informations objective function was derived by minimising $L\left(\mathbf{P}^0, \mathbf{Q}^0\right)$.

7.2 ACE: Hierarchical Vector Quantiser

If the above ACE network is modified slightly, so that the model $\mathbf{Q}$ has exactly the same structure as before, but is Gaussian rather than perfect, then $Q^{l|l+1}_{i_l,i_{l+1}}$ becomes

$$Q^{l|l+1}_{i_l,i_{l+1}} \rightarrow Q^{l|l+1}_{\mathbf{i}_l,\mathbf{i}_{l+1}} = Q^{l|l+1}_{\mathbf{i}^1_l,i^1_{l+1}} Q^{l|l+1}_{\mathbf{i}^2_l,i^2_{l+1}} \cdots \tag{62}$$

where the individual $Q^{l|l+1}_{\mathbf{i}^c_l,i^c_{l+1}}$ are Gaussian. The expression for D (i.e. an FMC-ladder without the output term) then becomes (compare equation 23)

$$\begin{aligned}
D &= -\sum_{l=0}^{L-1} \sum_{\mathbf{i}_l} P^l_{\mathbf{i}_l} \sum_{\mathbf{i}_{l+1}} P^{l+1|l}_{\mathbf{i}_{l+1},\mathbf{i}_l} \log \left(Q^{l|l+1}_{\mathbf{i}^1_l,i^1_{l+1}} Q^{l|l+1}_{\mathbf{i}^2_l,i^2_{l+1}} \cdots \right) \\
&= -\sum_{l=0}^{L-1} \sum_{\mathbf{i}_l} P^l_{\mathbf{i}_l} \sum_{i^1_{l+1},i^2_{l+1},\cdots} \left(P^{l+1|l}_{i^1_{l+1},\mathbf{i}^1_l} P^{l+1|l}_{i^2_{l+1},\mathbf{i}^2_l} \cdots \right) \log \left(Q^{l|l+1}_{\mathbf{i}^1_l,i^1_{l+1}} Q^{l|l+1}_{\mathbf{i}^2_l,i^2_{l+1}} \cdots \right) \\
&= -\sum_{l=0}^{L-1} \sum_{\mathbf{i}_l} P^l_{\mathbf{i}_l} \sum_c \sum_{i^c_{l+1}} P^{l+1|l}_{i^c_{l+1},\mathbf{i}^c_l} \log Q^{l|l+1}_{\mathbf{i}^c_l,i^c_{l+1}} \\
&= \sum_{l=0}^{L-1} \sum_c \left(\frac{D^{l,c}_{FMC}}{4\left(\sigma^{l,c}\right)^2} - \log \frac{V^{l,c}}{\sqrt{2\pi}\sigma^{l,c}} \right)
\end{aligned} \tag{63}$$

which is the objective function for a tree of coupled soft VQs. Thus the ACE network, with a Gaussian model $\mathbf{Q}$, is a hierarchical vector quantiser, in which each layer encodes the clusters in the previous layer [11, 12].

8 Conclusions

The objective function for optimising the density model of a Markov source may be applied to the problem of optimising the joint density of all the layers of a neural network. This is possible because the joint state of all of the network layers may be viewed as a Markov chain of states (each layer is connected only to adjacent layers). The objective function may readily be shown to be equivalent to a sum of folded Markov chain objective functions, each of which connects a pair of adjacent layers, plus a term which specifies the cost of building a density model in the output layer. This representation makes contact with the results reported in [17], which allows many results to be unified into a single approach (i.e. a single objective function).

The most significant aspect of this unification is the fact that all layers of a neural network are treated on an equal footing, unlike in the conventional approach to density modelling where the input layer is accorded a special status. For instance, this leads to a modular approach to building neural networks, where all of the modules have the same structure.

9 Acknowledgements

I thank Chris Webber for many useful conversations that we had during the course of this research. I also thank Peter Dayan and Geoffrey Hinton for conversations that we had about the relationship between folded Markov chains and Helmholtz machines during the "Neural Networks and Machine Learning" programme at the Newton Institute in Cambridge.

References

[1] Becker S and Hinton G E, 1992, *Nature*, **355**, 161-163, Self-organising neural network that discovers surfaces in random-dot stereograms.

[2] Dayan P, Hinton G E, Neal R M and Zemel R S, 1995, *Neural Computation*, **7**, 889-904, The Helmholtz machine.

[3] Dayan P and Hinton G E, 1996, *Neural Networks*, **9**(8), 1385-1403, Varieties of Helmholtz machine.

[4] Farvardin N, 1990, *IEEE Trans. IT*, **36**(4), 799-809, A study of vector quantisation for noisy channels.

[5] Hinton G E and Zemel R S, 1994, in Cowan J D, Tesauro G, and Alspector F (eds), *Advances in Neural Information Processing Systems*, **6**, San Francisco: Morgan Kaufmann, Autoencoders, minimum description length, and Helmholtz free energy.

[6] Hinton G E, Dayan P, Frey B J and Neal R M, 1995, *Science*, **268**, 1158-1161, The "wake-sleep" algorithm for unsupervised neural networks.

[7] Kohonen T, 1989, Springer-Verlag, Self-organisation and associative memory.

[8] Kumazawa H, Kasahara M and Namekawa T, 1984, *Electronic. Eng. Japan*, **67B**(4), 39-47, A construction of vector quantizers for noisy channels.

[9] Linde Y, Buzo A and Gray R M, 1980, *IEEE Trans. COM*, **28**, 84-95, An algorithm for vector quantiser design.

[10] Luttrell S P,1988, *Proc. 2nd IEEE Int. Conf. on Neural Networks*, San Diego, **1**, 93-100, Self-organising multilayer topographic mappings.

[11] Luttrell S P, 1989, *Proc. 1st IEE Conf. on Artificial Neural Networks*, 2-6, Hierarchical self-organising networks.

[12] Luttrell S P, 1989, *Proc. IEE Part I*, **136**(6), 405-413, Hierarchical vector quantisation.

[13] Luttrell S P, 1990, *IEEE Transactions on Neural Networks*, **1**, 229-232, Derivation of a class of training algorithms.

[14] Luttrell S P, 1991, *Proc. SPIE Conf. on Adaptive Signal Processing*, **1565**, 518-528, A hierarchical network for clutter and texture modelling.

[15] Luttrell S P, 1991, *Proc. 2nd IEE Conf. on Artificial Neural Networks*, Bournemouth, 5-9, Self-supervised training of hierarchical vector quantisers.

[16] Luttrell S P, 1992, Proc. IEE Part F, **139**(6), 371-377, Self-supervised adaptive networks.

[17] Luttrell S P, 1994, *Neural Computation*, **6**, 767-794, A Bayesian analysis of self-organising maps.

[18] Luttrell S P, 1994, *Proc. IEE Vision, Image and Signal Processing*, **141**, 251-260, The partitioned mixture distribution: an adaptive Bayesian network for low-level image processing.

[19] Luttrell S P, 1994, *Proc. 14th Int. MAXENT Workshop*, 269-278, Kluwer, The cluster expansion: a hierarchical density model.

[20] Luttrell S P, 1994, *Proc. 14th Int. MAXENT Workshop*, 279-286, Kluwer, The partitioned mixture distribution: multiple overlapping density models.

[21] Luttrell S P, 1996, *Network*, **7**, 285-290, A discrete firing event analysis of the adaptive cluster expansion network.

[22] Luttrell, 1997, *Proc. 5th IEE Conf. on Artificial Neural Networks*, 59-63, Partitioned mixture distributions: the dynamical case.

[23] Rissanen J, 1978, *Automatica*, **14**, 465-471, Modelling by shortest data description.

[24] Rissanan J, 1989, World Scientific, Stochastic complexity in statistical enquiry.

[25] Shannon C E, 1948, *Bell Syst. Tech. J.*, The mathematical theory of communication, **27**, 379-423 and 623-656.