To appear in: *Neural Computation*

Localist Attractor Networks

Richard S. Zemel
Department of Computer Science
University of Toronto
Toronto, ON M5S 1A4
zemel@cs.toronto.edu

Michael C. Mozer
Department of Computer Science
University of Colorado
Boulder, CO 80309-0430
mozer@cs.colorado.edu

Abstract

Attractor networks, which map an input space to a discrete output space, are useful for *pattern completion*—cleaning up noisy or missing input features. However, designing a net to have a given set of attractors is notoriously tricky; training procedures are CPU intensive and often produce spurious attractors and ill-conditioned attractor basins. These difficulties occur because each connection in the network participates in the encoding of multiple attractors. We describe an alternative formulation of attractor networks in which the encoding of knowledge is local, not distributed. Although localist attractor networks have similar dynamics to their distributed counterparts, they are much easier to work with and interpret. We propose a statistical formulation of localist attractor net dynamics, which yields a convergence proof and a mathematical interpretation of model parameters. We present simulation experiments that explore the behavior of localist attractor networks, showing that they yield few spurious attractors, and they readily exhibit two desirable properties of psychological and neurobiological models, *priming*—faster convergence to an attractor if the attractor has been recently visited—and *gang effects*—in which the presence of an attractor enhances the attractor basins of neighboring attractors.

1 Introduction

Attractor networks map an input space, usually continuous, to a sparse output space. For an interesting and important class of attractor nets, the output space is composed of a discrete set of alternatives. Attractor networks have a long history in neural network research, from relaxation models in vision (Marr and Poggio, 1976; Hummel and Zucker, 1983), to the early work of Hopfield (1982, 1984), to the stochastic Boltzmann machine (Ackley, Hinton, and Sejnowski, 1985; Movellan, 1990), to symmetric recurrent back-propagation networks (Almeida, 1987; Pineda, 1987).

Acknowledgements: We thank Joe Holmgren for help with running simulation studies. We also thank the reviewers for helpful comments, and the Vision and Modeling Group at the MIT Media Lab for making the face image database publicly available. Supported by ONR award N00014-98-1-0509 and McDonnell-Pew award 95-1 to RZ, and NSF award IBN-9873492 and McDonnell-Pew award 97-18 to MM.

Attractor networks are often used for *pattern completion*, which involves filling in missing, noisy, or incorrect features in an input pattern. The initial state of the attractor net is determined by the input pattern. Over time, the state is drawn to one of a predefined set of states—the *attractors*. An attractor net is generally implemented by a set of visible units whose activity represents the instantaneous state, and optionally, a set of hidden units that assist in the computation. Attractor dynamics arise from interactions among the units, and can be described by a state trajectory (Figure 1a)

Pattern completion can be accomplished using algorithms other than attractor nets. For example, a nearest-neighbor classifier will compare the input to each of a set of predefined prototypes, and will select as the completion the prototype with the smallest Mahalanobis distance to the input (see e.g., Duda and Hart, 1973). Attractor networks have several benefits over this scheme. First, if the attractors can be characterized by compositional structure, it is inefficient to enumerate them as required by a nearest-neighbor classifier; the compositional structure can be encoded implicitly in the attractor network weights. Second, attractor networks have some degree of biological plausibility (Amit and Brunel, 1997; Kay, Lancaster, and Freeman, 1996). Third, in most formulations of attractor nets (e.g., Golden, 1988; Hopfield, 1982), the dynamics can be characterized by gradient descent in an energy landscape, allowing one to partition the output space into *attractor basins* (Figure 1a). In many domains, the energy landscape and the corresponding structure of the attractor basins are key to obtaining desirable behavior from an attractor net. Consider a domain where attractor nets have proven to be extremely valuable—psychological models of human cognition. Instead of homogeneous attractor basins, modelers often sculpt basins that depend on the recent history of the network (*priming*) and the arrangement of attractors in the space (*the gang effect*).

Priming is the situation where a network is faster to land at an attractor if it has recently visited the same attractor. Priming is achieved by broadening and deepening attractor basins as they are visited (Becker, Moscovitch, Behrmann, and Joordens, 1997; McClelland and Rumelhart, 1985; Mozer, Sitton, and Farah, 1998; Plaut and Shallice, 1994). This mechanism allows modelers to account for a ubiquitous property of human behavior: People are faster to perceive a stimulus if they have recently experienced the same or a closely related stimulus, or made the same or a closely related response.

In the gang effect, the strength or pull of an attractor is influenced by other attractors in its neighborhood. Figure 1b illustrates the gang effect: the proximity of the two rightmost attractors creates a deeper attractor basin, so that if the input starts at the origin it will get pulled to the right. This effect is an emergent property of the distribution of attractors. The gang effect results in the mutually reinforcing or inhibitory influence of similar items and allows for the explanation of behavioral data in a range of domains including reading (McClelland and Rumelhart, 1981; Plaut, McClelland, Seidenberg, and Patterson, 1996), syntax (Tabor, Juliano, and Tanenhaus, 1997), semantics (McRae, de Sa, and Seidenberg, 1997), memory (Redish and Touretzky, 1998; Samsonovich and McNaughton, 1997), olfaction (Kay et al., 1996), and schizophrenia (Horn and Ruppin, 1995).

Training an attractor net is notoriously tricky. Training procedures are CPU intensive and often produce spurious attractors and ill-conditioned attractor basins (e.g., Mathis, 1997; Rodrigues and Fontanari, 1997). Indeed, we are aware of no existing procedure that can

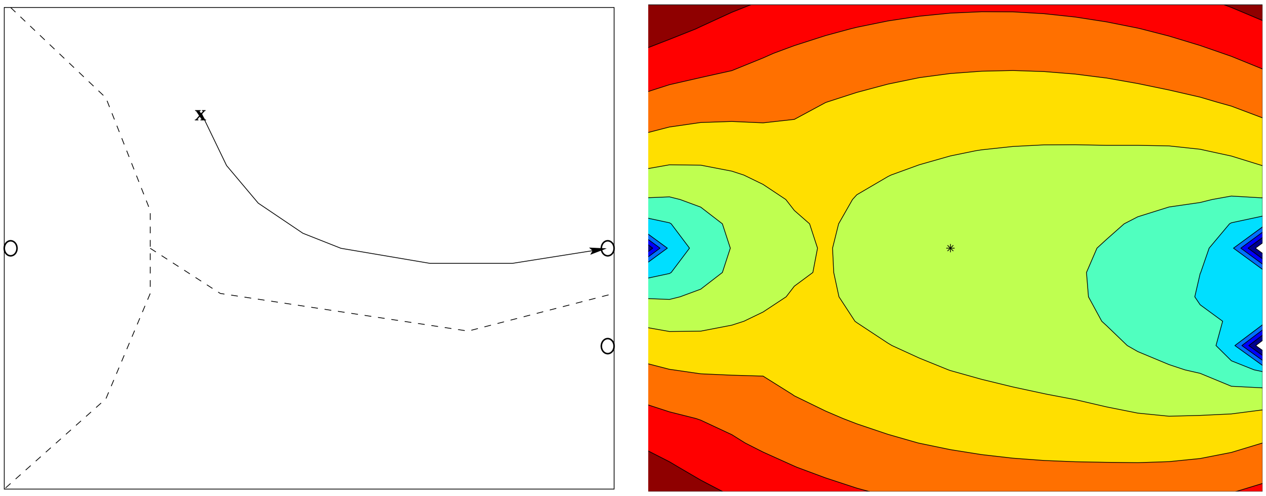

Figure 1: (a) A two-dimensional space can be carved into three regions (dashed lines) by an attractor net. The dynamics of the net cause an input pattern (the X) to be mapped to one of the attractors (the O's). The solid line shows the temporal trajectory of the network state. (b) the actual energy landscape for a localist attractor net as a function of $\hat{\mathbf{y}}$, when the input is fixed at the origin and there are three attractors, with a uniform prior. The shapes of attractor basins are influenced by the proximity of attractors to one another (the *gang effect*). The origin of the space (depicted by a point) is equidistant from the attractor on the left and the attractor on the upper right, yet the origin clearly lies in the basin of the right attractors.

robustly translate an arbitrary specification of an attractor landscape into a set of weights. These difficulties are due to the fact that each connection participates in the specification of multiple attractors; thus, knowledge in the net is *distributed* over connections.

We describe an alternative attractor network model in which knowledge is *localized*, hence the name *localist attractor network*. The model has many virtues, including:

- it provides a trivial procedure for wiring up the architecture given an attractor landscape,
- spurious attractors are eliminated,
- an attractor can be primed by adjusting a single parameter of the model,
- the model achieves gang effects,
- the model parameters have a clear mathematical interpretation, which clarifies how the parameters control the qualitative behavior of the model (e.g., the magnitude of gang effects), and
- we offer proofs of convergence and stability.

A localist attractor net consists of a set of n *state units* and m *attractor units*. Parameters associated with an attractor unit i encode the center in state space of its attractor basin, denoted $\mathbf{w}_i$, and its "pull" or strength, denoted π_i. For simplicity, we assume that the attractor basins have a common width σ. The activity of an attractor at time t, $q_i(t)$, reflects the normalized distance from its center to the current state, $\mathbf{y}(t)$, weighted by its strength:

$$q_i(t) = \frac{\pi_i g(\mathbf{y}(t), \mathbf{w}_i, \sigma(t))}{\sum_j \pi_j g(\mathbf{y}(t), \mathbf{w}_j, \sigma(t))} \tag{1}$$

$$g(\mathbf{y}, \mathbf{w}, \sigma) = \exp(-|\mathbf{y} - \mathbf{w}|^2/2\sigma^2) \tag{2}$$

Thus, the attractors form a layer of normalized radial-basis-function units.

In most attractor networks, the input to the net, $\mathcal{E}$, serves as the initial value of the state, and thereafter the state is pulled toward attractors in proportion to their activity. A straightforward expression of this behavior is:

$$\mathbf{y}(t+1) = \alpha(t)\mathcal{E} + (1 - \alpha(t)) \sum_i q_i(t)\mathbf{w}_i. \tag{3}$$

where α trades off the external input and attractor influences. We will refer to the external input, $\mathcal{E}$, as the *observation*. A simple method of moving the state from the observation towards the attractors involves setting $\alpha = 1$ on the first update and $\alpha = 0$ thereafter. More generally, however, one might want to gradually reduce α over time, allowing for a persistent effect of the observation on the asymptotic state.

In our formulation of a localist attractor network, the attractor width σ is not a fixed, free parameter, but instead is dynamic and model-determined:

$$\sigma_y^2(t) = \frac{1}{n} \sum_i q_i |\mathbf{y}(t) - \mathbf{w}_i|^2 \tag{4}$$

In addition, we include a fixed parameter σ_z that describes the unreliability of the observation, i.e., if $\sigma_z = 0$, then the observation is wholly reliable, so $\mathbf{y} = \mathcal{E}$. Finally, $\alpha(t)$ is then defined as a function of these parameters:

$$\alpha(t) = \sigma_y^2(t)/(\sigma_y^2(t) + \sigma_z^2) \tag{5}$$

Below we present a formalism from which these particular update equations can be derived.

The localist attractor net is motivated by a generative model of the observed input based on the underlying attractor distribution, and the network dynamics corresponds to a search for a maximum likelihood interpretation of the observation. In the following section, we derive this result, and then present simulation studies of the architecture.

2 A Maximum Likelihood Formulation

The starting point for the statistical formulation of a localist attractor network is a mixture-of-Gaussians model. A standard mixture-of-Gaussians consists of m Gaussian density functions in n dimensions. Each Gaussian is parameterized by a mean, a covariance matrix, and a mixture coefficient. The mixture model is *generative*, i.e., it is considered to have produced a set of observations. Each observation is generated by selecting a Gaussian with probability proportional to the mixture coefficients and then stochastically selecting a point from the corresponding density function. The model parameters (means, covariances, and mixture coefficients) are adjusted to maximize the likelihood of a set of observations. The Expectation-Maximization (EM) algorithm provides an efficient procedure for estimating the parameters (Dempster, Laird, and Rubin, 1977). The Expectation step calculates the posterior probability q_i of each Gaussian for each observation, and the Maximization step calculates the new parameters based on the previous values and the set of q_i. The operation of this standard mixture-of-Gaussians model is cartooned in Figure 2a.

The mixture-of-Gaussians model can provide an interpretation for a localist attractor network, in an unorthodox way. Each Gaussian corresponds to an attractor, and an observation corresponds to the state. Now, however, instead of fixing the observation and adjusting the Gaussians, we fix the Gaussians and adjust the observation. If there is a single observation, $\alpha = 0$, and all Gaussians have uniform spread σ, then Equation 1 corresponds to the Expectation step, and Equation 3 to the Maximization step in this unusual mixture model. The operation of this unorthodox mixture-of-Gaussians model is cartooned in Figure 2b.

Unfortunately, this simple characterization of the localist attractor network does not produce the desired behavior. Many situations produce partial solutions, or *blend* states, in which the observation does not end up at an attractor. For example, if two Gaussians overlap significantly, the most likely value for the observation is midway between them rather than at the center of either Gaussian.

We therefore extend this mixture-of-Gaussians formulation to better characterize the attractor network. We introduce an additional level at the bottom containing the observation. This observation is considered fixed, as in a standard generative model. The state, contained in the middle level, is an internal model of the observation that is generated from the attractors. The attractor dynamics involve an iterative search for the internal model that

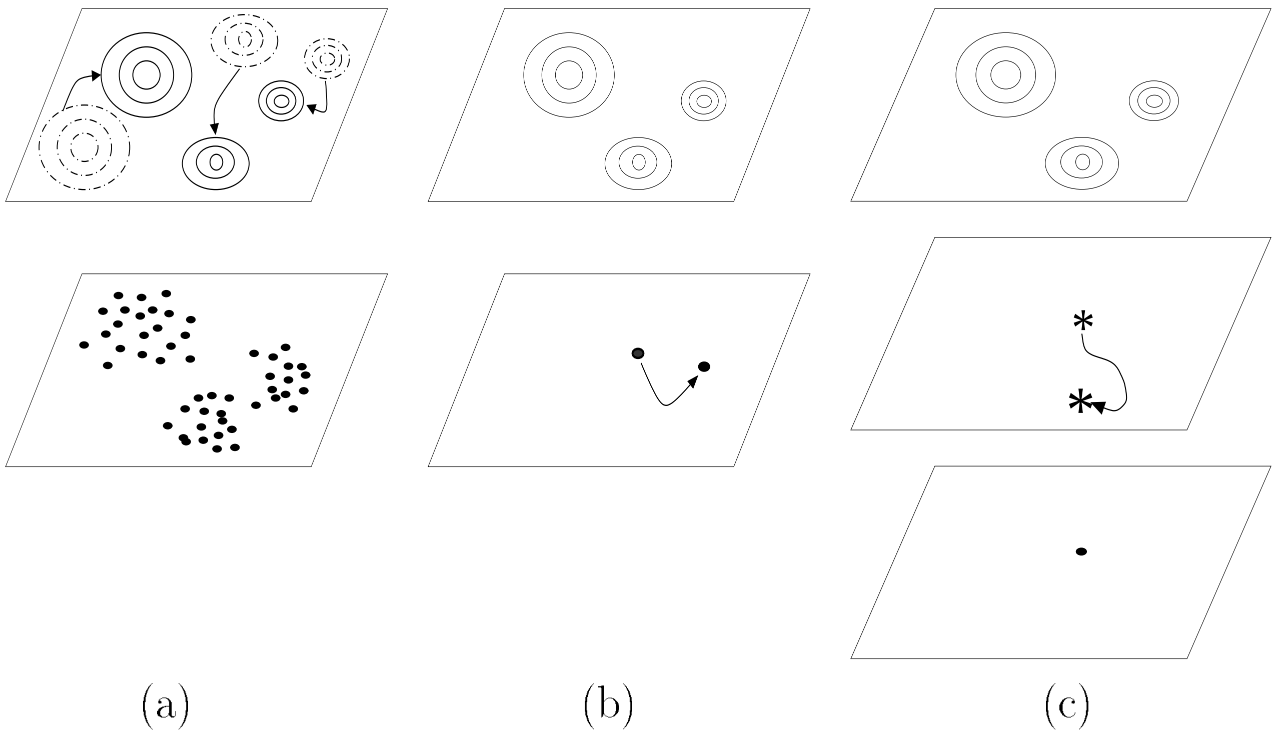

Figure 2: A standard mixture-of-Gaussians model adjusts the parameters of the generative Gaussians (elliptical contours in upper rectangle, which represents an n-dimensional space) to fit a set of observations (dots in lower rectangle, which also represnts an n-dimensional space). (b). An unorthodox mixture-of-Gaussians model initializes the state to correspond to a single observation and then adjusts the state to fit the fixed Gaussians. (c) A localist attractor network adds an extra level to this hierarchy, a bottom level containing the observation, which is fixed. The middle level contains the state. The attractor dynamics can then be viewed as an iterative search to find a single state that could have been generated from one of the attractors while still matching the observation.

fits the observation and attractor structure. These features make the formulation more like a standard generative model, in that the internal aspects of the model are manipulated to account for a stable observation. The operation of the localist attractor model is cartooned in Figure 2c.

Formulating the localist attractor network in a generative framework permits the derivation of update equations from a statistical basis. Note that the generative framework is not actually used to generate observations, or to update the state. Instead it provides the setting in which to optimize and analyze computation.

In this new formulation, each observation is considered to have been generated by moving down this hierarchy:

1. select an attractor $\mathbf{x} = i$ from the set of attractors
2. select a state (i.e., a pattern of activity across the state units) based on the preferred location of that attractor: $\mathbf{y} = \mathbf{w}_i + \mathcal{N}_y$
3. select an observation $\mathbf{z} = \mathbf{y}\mathbf{G} + \mathcal{N}_z$

The observation $\mathbf{z}$ produced by a particular state $\mathbf{y}$ depends on the generative weight matrix $\mathbf{G}$. In the networks we consider here, the observation and state spaces are identical, so $\mathbf{G}$ is the identity matrix, but the formulation allows for the observation to lie in some other space. $\mathcal{N}_y$ and $\mathcal{N}_z$ describe the zero-mean, spherical Gaussian noise introduced at the two levels, with deviations σ_y and σ_z, respectively.

Under this model, one can fit an observation $\mathcal{E}$ by finding the posterior distribution over the hidden states (X and $\mathbf{Y}$) given the observation:

$$p(X = i, \mathbf{Y} = \mathbf{y} | \mathbf{Z} = \mathcal{E}) = \frac{p(\mathcal{E}|\mathbf{y}, i)p(\mathbf{y}, i)}{p(\mathcal{E})} = \frac{p(\mathcal{E}|\mathbf{y})\pi_i p(\mathbf{y}|i)}{\int_{\mathbf{y}} p(\mathcal{E}|\mathbf{y}) \sum_i \pi_i p(\mathbf{y}|i) d\mathbf{y}} \tag{6}$$

where the conditional distributions are Gaussian: $p(\mathbf{Y} = \mathbf{y}|X = i) = \mathcal{G}(\mathbf{y}|\mathbf{w}_i, \sigma_y)$ and $p(\mathcal{E}|\mathbf{Y} = \mathbf{y}) = \mathcal{G}(\mathcal{E}|\mathbf{y}, \sigma_z)$. Evaluating the distribution in Equation 6 is tractable, because the partition function (the denominator) is a sum of a set of Gaussian integrals. This posterior distribution for the attractors corresponds to a distribution in state space that is a weighted sum of Gaussians.

During inference using this model, this intermediate level can be considered as encoding the distribution over states implied by the attractor posterior probabilities. However, we impose a key restriction: at any one time, the intermediate level can only represent a single position in state space, rather than the entire distribution over states. This restriction is motivated by the fact that the goal of the computation in the network is to settle on the single best state, and furthermore, at every step of the computation, the network should represent the single most likely state that fits the attractors and observation. Hence the attractor dynamics corresponds to an iterative search through state space to find the most likely single state that: (a) was generated by the mixture-of-Gaussian attractors, and (b) in turn generated the observation.

To accomplish these objectives, we adopt a *variational* approach, approximating the posterior by another distribution $\mathcal{Q}(X, \mathbf{Y}|\mathcal{E})$. Based on this approximation, the network dynamics can be seen as minimizing an objective function that describes an upper bound on the negative log probability of the observation given the model and its parameters. In a variational approach, one is free to choose any form of $\mathcal{Q}$ to estimate the posterior distribution, but a better estimate will allow the network to approach a maximum likelihood solution (Saul, Jaakkola, and Jordan, 1996). Here we select a very simple posterior: $\mathcal{Q}(X, \mathbf{Y}) = q_i \delta(\mathbf{Y} = \hat{\mathbf{y}})$, where $q_i = \mathcal{Q}(X = i)$ is the responsibility assigned to attractor i, and $\hat{\mathbf{y}}$ is the estimate of the state that accounts for the observation. The delta function over $\mathbf{Y}$ implements the restriction that the explanation of an input consists of a single state.

Given this posterior distribution, the objective for the network is to minimize the free energy F, described here for a particular observation $\mathcal{E}$:

$$\begin{aligned} F(\mathbf{q}, \hat{\mathbf{y}}|\mathcal{E}) &= \sum_i \int \mathcal{Q}(X = i, \mathbf{Y} = \mathbf{y}) \ln \frac{\mathcal{Q}(X = i, \mathbf{Y} = \mathbf{y})}{\mathcal{P}(\mathcal{E}, X = i, \mathbf{Y} = \mathbf{y})} d\mathbf{y} \\ &= \sum_i q_i \ln \frac{q_i}{\pi_i} - \ln p(\mathcal{E}|\hat{\mathbf{y}}) - \sum_i q_i \ln p(\hat{\mathbf{y}}|i) \end{aligned}$$

where π_i is the prior probability (mixture coefficient) associated with attractor i. These priors are parameters of the generative model, as are σ_y, σ_z, and $\mathbf{w}$. Thus, the free energy is:

$$F(\mathbf{q}, \hat{\mathbf{y}}|\mathcal{E}) = \sum_i q_i \ln \frac{q_i}{\pi_i} + \frac{1}{2\sigma_z^2}|\mathcal{E} - \hat{\mathbf{y}}|^2 + \frac{1}{2\sigma_y^2} \sum_i q_i |\hat{\mathbf{y}} - \mathbf{w}_i|^2 + n \ln(\sigma_y \sigma_z) \tag{7}$$

Given an observation, a good set of parameters for the estimated posterior $\mathcal{Q}$ can be determined by alternating between updating the generative parameters and these posterior parameters. The update procedure is guaranteed to converge to a minimum of F, as long as the updates are done asynchronously and each update step minimizes F with respect to that parameter (Neal and Hinton, 1998).

The update equations for the various parameters can be derived by minimizing F with respect to each of them. In our simulations, we hold most of the parameters of the generative model constant, such as the priors π, the weights $\mathbf{w}$, and the generative noise in the observation, σ_z. Minimizing F with respect to the other parameters—$q_i, \mathbf{y}, \sigma_y, \alpha$—produces the update equations (Equations 1, 3, 4, and 5, respectively) given above.

3 Running the Model

The updates of the parameters using Equations 1-5 can be performed sequentially in any order. In our simulations, we initialize the state $\hat{\mathbf{y}}$ to $\mathcal{E}$ at time 0, and then cycle through updates of $\{q_i\}$, σ_y, and then $\hat{\mathbf{y}}$.

Note that for a given set of attractor locations and strengths, and a fixed σ_z, the operation of the model is solely a function of the observation $\mathcal{E}$. Even though the model of the generative process has stochasticity, the process of inferring its parameters from an observation is deterministic. That is, there is no randomness in the operation of the model.

The energy landscape changes as a function of the attractor dynamics. The initial shape of the attractor basins are determined by the set of $\mathbf{w}_i$, π_i, and σ_y. Figure 3 illustrates how the free energy landscape changes over time.

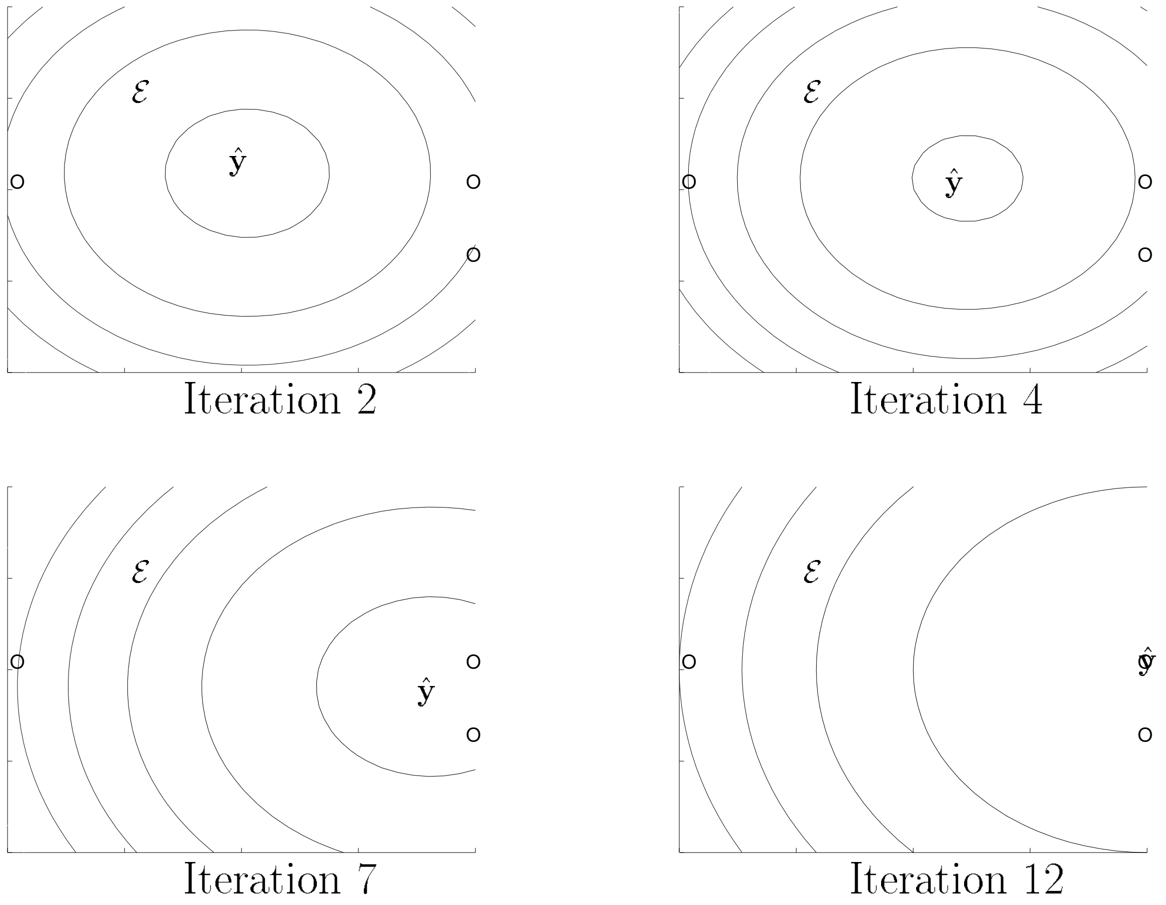

Figure 3: Each figure in this sequence shows the free energy landscape for the attractor net, where the three attractors (each marked by an "O") are in the same locations as in Figure 1, and the observation is in the location marked by an $\mathcal{E}$. The state (marked by $\hat{\mathbf{y}}$) begins at $\mathcal{E}$. Each contour plot shows the free energy for points throughout state space, given the current values of σ_y and $\{q_i\}$, and the constant values of $\mathcal{E}$, σ_z, $\{\mathbf{w}_i\}$, and $\{\pi_i\}$. In this simulation, $\sigma_z = 1.0$ and the priors on the attractors are uniform. Note how the gang effect pulls the state $\hat{\mathbf{y}}$ away from the nearest attractor (the leftmost one).

This generative model avoids the problem of spurious attractors described above for the standard Gaussian mixture model. Intuition into how the model avoids spurious attractors can be gained by inspecting the update equations. These equations have the effect of tying together two processes: moving $\hat{\mathbf{y}}$ closer to some $\mathbf{w}_i$ than the others, and increasing the corresponding responsibility q_i. As these two processes evolve together, they act to decrease the noise σ_y, which accentuates the pull of the attractor. Thus, stable points that do not correspond to the attractors are rare.

This characterization of the attractor dynamics indicates an interesting relationship to standard techniques in other forms of attractor networks. Many attractor networks (e.g., Geva and Sitte, 1991) avoid settling into spurious attractors by using simulated annealing (Kirkpatrick, Gellat, and Vecchi, 1983), in which a temperature parameter that controls the gain of the unit activation functions is slowly decreased. Another common attractor-network technique is "soft clamping", where some units are initially influenced predominantly by external input, but the degree of influence from other units increases during the settling procedure. In the localist attractor network, σ_y can be seen as a temperature parameter, controlling the gain of the state activation function. Also, as this parameter decreases, the influence of the observation on the state decreases relative to that of the other units. The key point is that as opposed to the standard techniques, these effects naturally fall out of the localist attractor network formulation. This obviates the need for additional parameters and update schedules in a simulation.

Note that we introduced the localist attractor net by extending the generative framework of a Gaussian mixture model. The Gaussian mixture model can also serve as the basis for an alternative formulation of a localist attractor net. The idea is straightforward: Consider the Gaussian mixture density model as a negative energy landscape, and consider the search for an attractor to take place via gradient ascent in this landscape. As long as the Gaussians are not too close together, the local maxima of the landscape will be the centers of the Gaussians, which we designated as the attractors. Gang effects will be obtained because the energy landscape at a point will depend on all the neighboring Gaussians. However, this formulation has a serious drawback: if attractors are nearby one another, spurious attractors will be introduced (because the point of highest probability density in a Gaussian mixture will lie between two Gaussian centers, if they are close to one another); and if attractors are moved far away, gang effects will dissipate. Thus, the likelihood of spurious attractors increases with the strength of gang effects. The second major drawback of this formulation is that it is slower to converge than the model we have proposed, as is typical of using gradient ascent versus EM for search. A third aspect of this formulation, which could also be seen as a drawback, is that it involves additional parameters in specifying the landscape. The depth and width of an attractor can be controlled independently of one another (via the mixture and variance coefficients, respectively), whereas in the model we presented, the variance coefficient is not a free parameter. Given the drawbacks of the gradient-ascent approach, we see the maximum-likelihood formulation as being more promising, and hence have focused on it in our presentation.

4 Simulation Studies

To create an attractor net, learning procedures could be used to train the parameters associated with the attractors $(\pi_i, \mathbf{w}_i)$ based on a specification of the desired behavior of the net, or the parameters could be hand-wired based on the desired structure of the energy landscape. The only remaining free parameter, σ_z, plays an important role in determining how responsive the system is to the observation.

We have conducted three sets of simulation studies to explore properties of localist attractor networks.

4.1 Random State Space

We have simulated the behavior of a network with a 200-dimensional state space and 200 attractors, which have been randomly placed at corners of the 200-D hypercube. For each condition we studied, we ran 100 trials, each time selecting one *source* attractor at random, corrupting it to produce an input pattern, and running the attractor net to determine its asymptotic state. The conditions involve systematically varying the value of σ_z and the percentage of missing features in the input, ν. Because the features have (1,–1) binary values, a missing feature is indicated with the value zero. In most trials, the final value of $\hat{\mathbf{y}}$ corresponds to the source attractor. Two other types of responses are observed: *spurious* responses are those in which the final state corresponds to no attractor; *adulterous* responses are those in which the state ends up not at the source attractor but instead at a neighboring one. Adulterous responses are due to gang effects: pressure from gangs has shrunk the source attractor basin, causing the network to settle to an attractor other than the one with smallest Euclidean distance from the input.

Figure 4a shows that the input must be severely corrupted (more than 85% of an input's features are distorted) before the net makes spurious responses. Further, the likelihood of spurious responses increases as σ_z increases, because the pull exerted by the input, $\mathcal{E}$, is too strong to allow the state to move into an attractor basin. Figure 4b also shows that the input must be severely corrupted before adulterous responses occur. However, as σ_z increases, the rate of adulterous responses decreases.

Gang effects are mild in these simulations because the attractors populated the space uniformly. If instead there was structure among the attractors, gang effects are more apparent. We simulated a structured space by creating a gang of attractors in the 200-D space—20 attractors that are near to one another, and another 20 attractors that are far from the gang and from each other. Across a range of values of σ_z and ν, the gang dominates, in that typically over 80% of the adulterous responses result in selection of a gang member.

4.2 Faces

We conducted studies of localist attractor networks in the domain of faces. In these simulations, different views of the faces of 16 different subjects comprise the attractors, and the associated gray-level images are points in the 120x128 state space. The database contains 27 versions of each face, obtained by varying the head orientation (left tilt, upright, or right

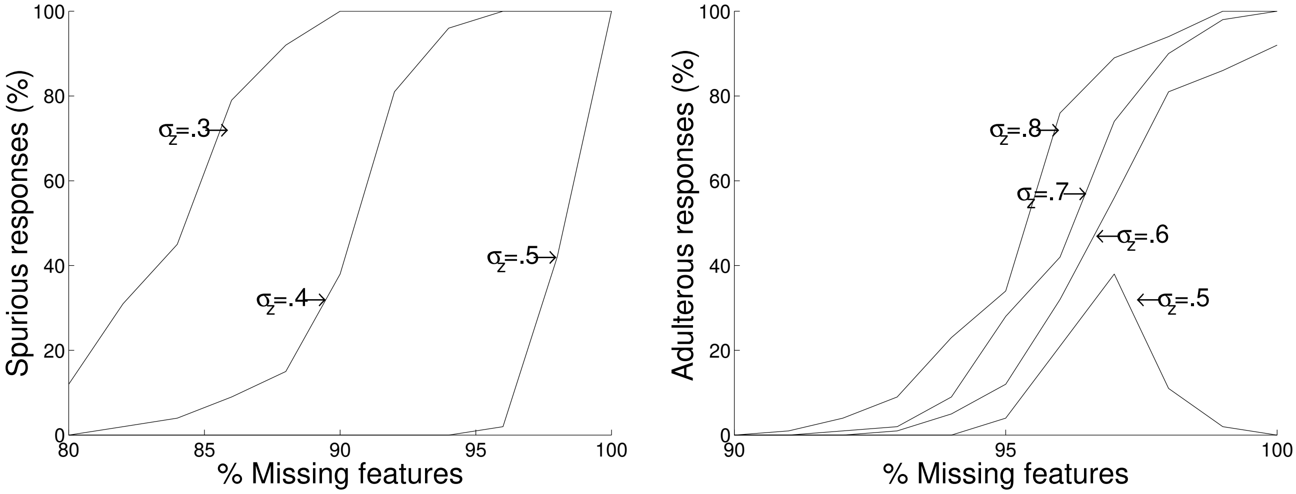

Figure 4: (a) The number of spurious responses increases as σ_z shrinks. (b) The final state ends up at a neighbor of the generating attractor more frequently when the system is less responsive to the external state (i.e., high σ_z). Note that at $\sigma_z = .5$, these responses fall when spurious responses increase.

Figure 5: Experiment with the face localist attractor network. Right-tilted images of the 16 different faces are the attractors. Here the initial state is a left-tilted image of one of these faces, and the final state corresponds to the appropriate attractor. The state is shown after every 3 iterations.

tilt), the lighting, and the scale. For our studies, we chose to vary tilt, and picked a single value for the other variables.

One aim of the studies was to validate basic properties of the attractor net. One simulation used as attractors the 16 right-tilted heads, and tested the ability of the network to associate face images at other orientations with the appropriate attractor. The asymptotic state was correct on all 32 input cases (left-tilted and upright) (see Figure 5). Note that this behavior can largely be accounted for by the greater consistency of the backgrounds in the images of the same face. Nonetheless, this simple simulation validates the basic ability of the attractor network to associate similar inputs.

We also investigated gang effects in face space. The 16 upright faces were used as attractors. In addition, the other 2 faces for one of the 16 subjects were also part of the attractor set, forming the "gang". The initial observation was a morphed face, obtained by a linear combination of the upright face of the gang-member and one of the other 15 subjects. In each case, the asymptotic state corresponded to the gang-member even when the initial weighting

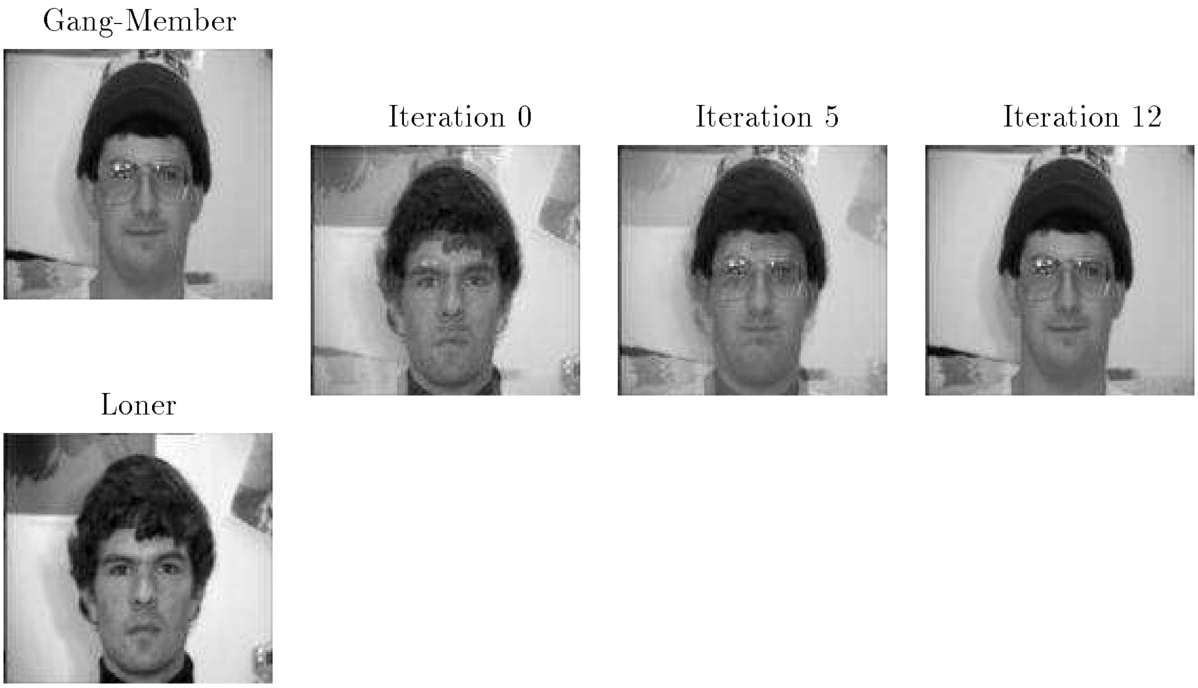

Figure 6: Another simulation study with the face localist attractor network. Upright images of each of the 16 faces, and both tilted version of one of the faces (the "Gang-Member" face shown above) are the attractors. The initial state is a morphed image constructed as a linear combination of the upright image of the Gang-Member face and the upright image of a second face, the "Loner" shown above, with the Loner face weighted by .65 and the Gang-Member by .35. Even though the initial state is much closer to the Loner, the asymptotic state corresponds to the Gang-Member.

assigned to this face was less than 40% (see Figure 6). This study shows that the dynamics of the localist attractor networks makes its behavior different than a nearest-neighbor lookup scheme.

4.3 Words

To test the architecture on a larger, structured problem, we modeled the domain of three-letter English words. The idea is to use the attractor network as a content addressable memory which might, for example, be queried to retrieve a word with P in the third position and any letter but A in the second position, a word such as HIP. The attractors consist of the 423 three-letter English words, from ACE to ZOO. The state space of the attractor network has one dimension for each of the 26 letters of the English alphabet in each of the 3 positions, for a total of 78 dimensions. We can refer to a given dimension by the letter and position it encodes, e.g., P_3 denotes the dimension corresponding to the letter P in the third position of the word. The attractors are at the corners of a $[-1, +1]^{78}$ hypercube. The attractor

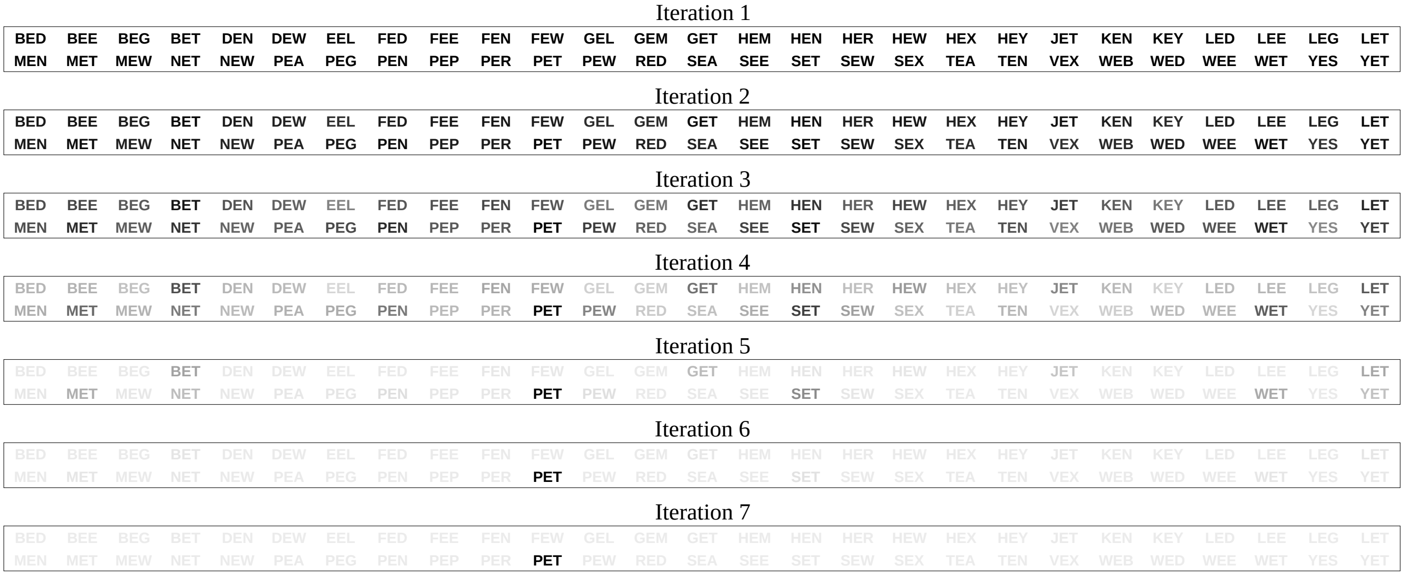

Figure 7: Simulation of the 3-letter word attractor network, queried with E_2. Each frame shows the relative activity of attractor units at a given processing iteration. Activity in each frame is normalized such that the most active unit is printed in black ink; the lighter the ink color, the less active the unit. All 54 attractor units for words containing E_2 are shown here; the other 369 attractor units do not become significantly active, because they are inconsistent with the input constraint. Initially the state of the attractor net is equidistant from all E_2 words, but by iteration 3, the subset containing N_3 or T_3 begin to dominate. The gang effect is at work here: 7 words contain N_3 and 10 contain T_3; these are the two most common endings. The selected word—PET—contains the most common first and last letters in the set of 3-letter E_2 words.

for a word such as HIP is located at the state having value –1 on all dimensions except for H_1, I_2, and P_3 which have value +1. The observation $\mathcal{E}$ specifies a state that constrains the solution. For example, one might specify "P in the third position" by setting $\mathcal{E}$ to +1 on dimension P_3 and to –1 on dimensions α_3, for all letters α other than P. One might specify "any letter but A in the second position" by setting $\mathcal{E}$ input to –1 on dimension A_2 and to 0 on all other dimensions α_2, for all letters α other than A. Finally, one might specify the absence of a constraint in a particular letter position, ρ, by setting $\mathcal{E}$ to 0 on dimensions α_ρ, for all letters α.

The task of the attractor net is to settle on a state corresponding to one of the three-letter words, given soft constraints on the letters of the word. McClelland and Rumelhart's (1981) interactive-activation model of word perception performs a similar computation, and our implementation exhibits the key qualitative properties of their model.

If the observation specifies a word, of course the attractor net will select that word. The interesting queries are those in which the observation underconstrains or overconstrains the solution. We illustrate with two examples of the network's behavior.

Suppose we provide an observation that specifies E_2 but no constraint on the first or third letter of the word. Fifty-four three-letter words contain E_2 but the attractor net lands

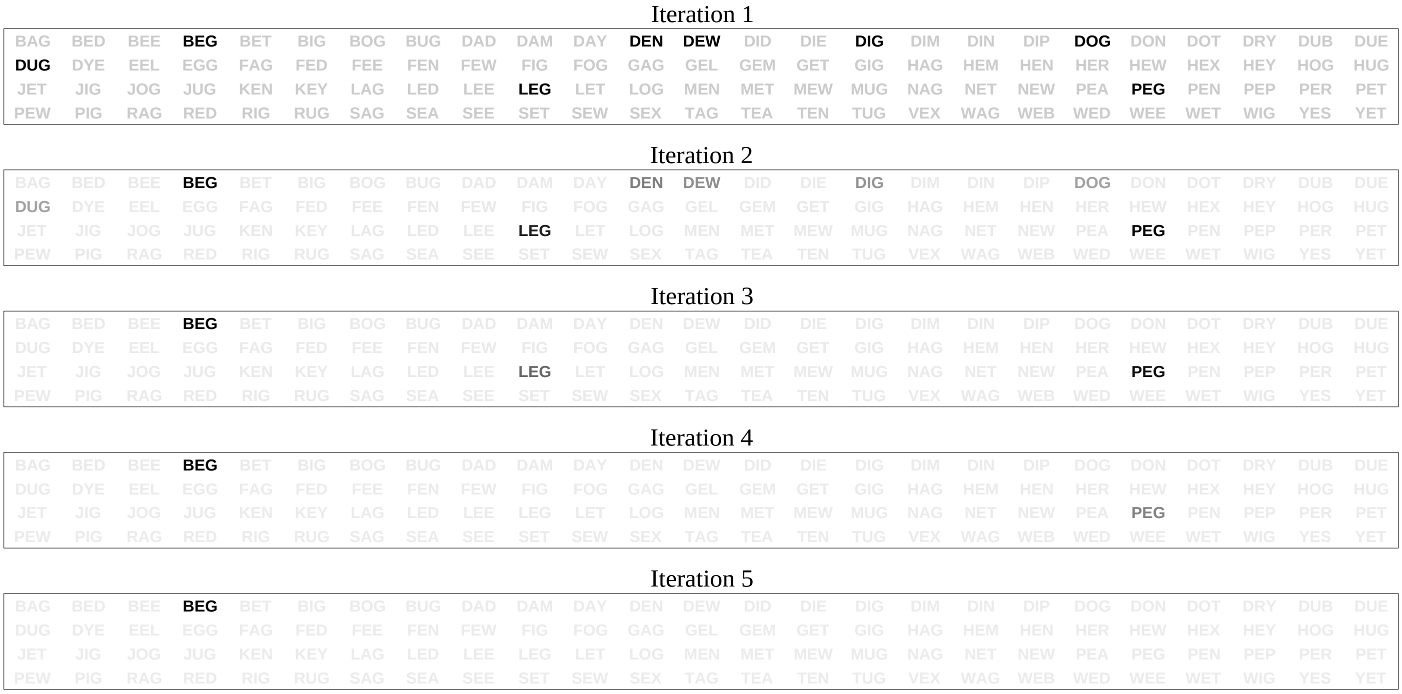

Figure 8: Simulation of the 3-letter word attractor network, queried with DEG. Only attractor units sharing at least one letter with DEG are shown. The selection, BEG, is a product of a gang effect. The gangs in this example are formed by words sharing two letters. The most common word beginnings are PE– (7 instances) and DI– (6); the most common word endings are –AG (10) and –ET (10); the most common first-last pairings are B–G (5) and D–G (3). One of these gangs supports B_1, two support E_2, and three support G_3, hence BEG is selected.

in the state corresponding to one specific word. Figure 7 shows the state of the attractor net as PET is selected. Each frame of the figure shows the relative activity at a given processing iteration of all 54 attractor units for words containing E_2. The other 369 attractor units do not become significantly active, because they are inconsistent with the input constraint. Activity in each frame is normalized such that the most active unit is printed in black ink; the lighter the ink color, the less active the unit. As the figure shows, initially the state of the attractor net is equally distant from all E_2 words, but by iteration 3, the subset containing N_3 or T_3 begin to dominate. The gang effect is at work here: 7 words contain N_3 and 10 contain T_3; these are the two most common endings. The selected word—PET—contains the most common first and last letters in the set of 3-letter E_2 words. The gang effect is sensitive to the set of candidate alternatives, not the entire set of attractors: A, B, and S are the most common first letters in the corpus, yet words beginning with these letters do not dominate over others.

Consider a second example in which we provide an observation that specifies D_1, E_2, and G_3. Because DEG is a nonword, no attractor exists for that state. The closest attractor shares two letters with DEG. Many such attractors exist, e.g., PEG, BEG, DEN, and DOG. Figure 8 shows the relative activity of attractor units for the DEG query. Only attractor

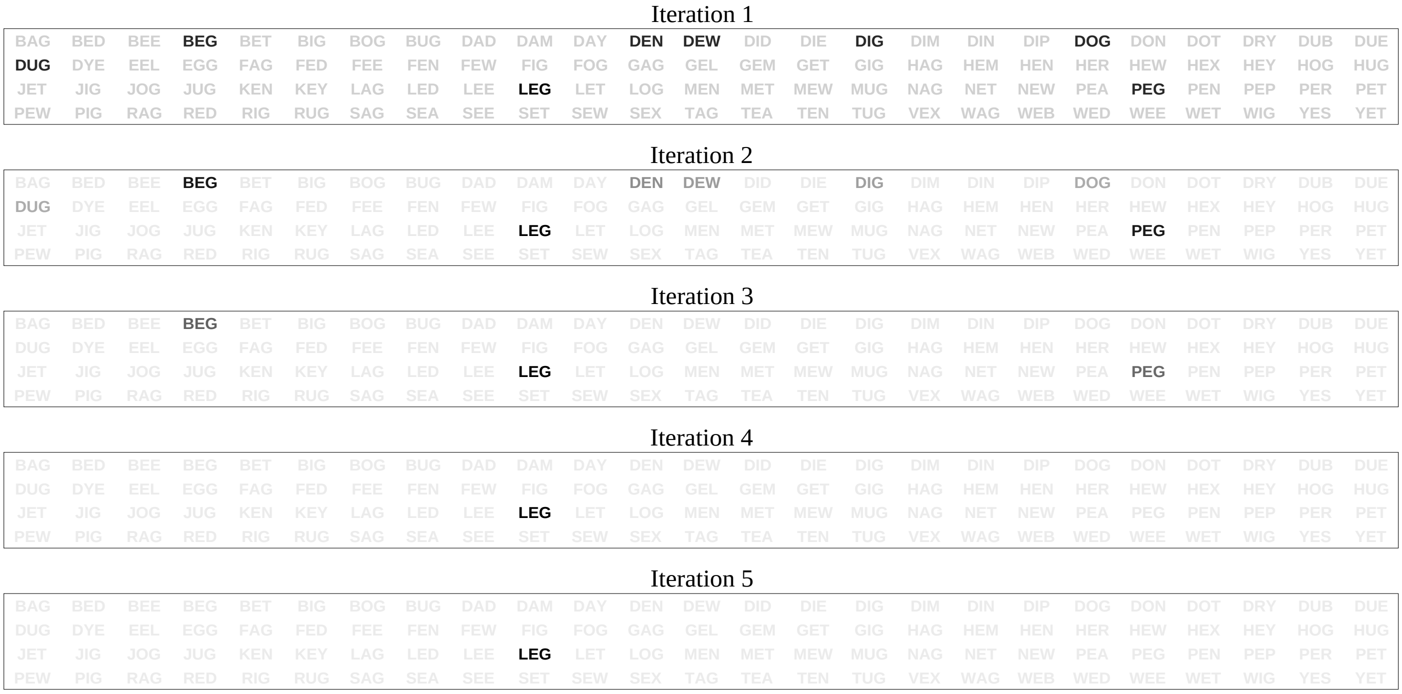

Figure 9: Simulation of the 3-letter word attractor network, queried with DEG, when the attractor LEG is primed by setting its prior to 1.2 times that of the other attractors. Priming causes the network to select LEG instead of BEG.

units sharing at least one letter with DEG are shown. The selection —BEG—is again a product of a gang effect. The gangs in this example are formed by words sharing two letters. The most common word beginnings are PE– (7 instances) and DI– (6); the most common word endings are –AG (10) and –ET (10); the most common first-last pairings are B–G (5) and D–G (3). One of these gangs supports B_1, two support E_2, and three support G_3 – the letters of the selected word.

For an ambiguous input like DEG, the competition is fairly balanced. BEG happens to win out due to its neighborhood support. However, the balance of the competition can be altered by priming another word that shares two letters with the input DEG. In the simulations reported above, the priors, π_i, were uniform. We can prime a particular attractor by slightly raising its prior relative to the priors of the other attractors. For example, in Figure 9, DEG is presented to the network after setting the relative prior of LEG to 1.2. LEG has enough of an advantage to overcome gang effects and suppress DEG. However, with a relative prior of 1.1, LEG is not strong enough to win out. The amount of priming required to ensure a word will win the competition depends on the size of its gang. For example, DOG, which belongs to fewer and smaller gangs, requires a relative prior of 2.3 to beat out DEG. By this simple demonstration, we have shown that incorporating mechanisms of priming into the localist attractor network model is simple. In contrast, implementing priming in distributed attractor networks is tricky, because strengthening one attractor may have broad consequences throughout the attractor space.

As a final test in the three-letter word domain, we presented random values of $\mathcal{E}$ to determine how often the net reached an attractor. The random input vectors had 80% of their elements set to zero, 10% to +1, and 10% to –1. In presentation of 1000 such inputs, only 1 did not reach an attractor—the criterion for reaching an attractor being that all state vector elements were within .1 units from the chosen attractor—clearly a demonstration of robust convergence.

5 Conclusion

Localist attractor networks offer an attractive alternative to standard attractor networks, in that their dynamics are easy to specify and adapt. Localist attractor network use *local* representations for the m attractors, each of which is a *distributed*, n-dimensional state representation. We described a statistical formulation of a type of localist attractor net, and showed that it provides a Lyapunov function for the system as well as a mathematical interpretation for the network parameters.

A related formulation of localist attractor networks was introduced by Mathis and Mozer (1996) and Mozer, Sitton, and Farah (1998). These papers described the application of this type of network to model psychological and neural data. The network and parameters in these earlier systems were motivated primarily on intuitive grounds.

In this paper, we propose a version of the localist attractor net whose dynamics are derived not from intuitive arguments but from a formal mathematical model. Our principled derivation provides a number of benefits, including more robust dynamics and reliable convergence than the model based on intuition, and a clear interpretation to the model parameters. In simulation studies, we found that the architecture achieves gang effects and accounts for priming effects, and also seldom ends up at a spurious attractor.

This primary drawback of the localist attractor approach its inefficiency if the attractors have componential structure. This can be traced to the lack of spurious attractors, which is generally a desirable property. However, in domains which have componential structure, spurious attractors may be a feature, not a drawback. That is, one may want to train an attractor net on a subset of the potential patterns in a componential domain and expect it will generalize to the remainder, e.g., train on attractors AX, BX, CX, AY, and BY, and generalize to the attractor CY. In this situation, "generalization" comes about by the creation of spurious attractors. Yet it turns out that traditional training paradigms for distributed attractor networks fare poorly on discovering componential structure (Noelle and Zimdars, 1999).

In conclusion, a localist attractor network shares the qualitative dynamics of a distributed attractor network, yet the model is easy to build, the parameters are explicit, and the dynamics are easy to analyze. The model is one step further from neural plausibility, but is useful for cognitive modeling or engineering. It is applicable when the number of items being stored is relatively small, which is characteristic of pattern recognition or associative memory applications. Finally, the approach is especially useful in cases where attractor locations are known, and the key focus of the network is the mutual influence of the attractors, as in many cognitive modelling studies.

References

Ackley, D.H., G.E. Hinton, and T.J. Sejnowski (1985). A learning algorithm for Boltzmann Machines. *Cognitive Science, 9.*

Almeida, L. B. (1987). A learning rule for asynchronous perceptrons with feedback in a combinatorial environment. In *Proceedings, 1st First International Conference on Neural Networks* , 609-618.

Amit, D.J. and Brunel, N. (1997). Model of global spontaneous activity and local structured activity during delay periods in the cerebral cortex. *Cerebral Cortex, 7(3)*, 237-252.

Becker, S., Moscovitch, M., Behrmann, M., and Joordens, S. (1997). Long-term semantic priming: A computational account and empirical evidence. *Journal of Experimental Psychology: Learning, Memory, and Cognition, 23(5)*, 1059-1082.

Dempster, A. P., Laird, N. M., and Rubin, D. B. (1997). Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society, B, 39(1)*, 1-38.

Duda, R. O. and Hart, P. E. (1973). *Pattern Classification and Scene Analysis*. Wiley.

Geva, S. and Sitte, J. (1991). An exponential response neural net. *Neural Computation, 3(4)*, 623-632.

Golden, R. (1988). Probabilistic characterization of neural model computations. In D. Z. Anderson (Ed.), *Neural Information Processing Systems* (pp. 310-316). American Institute of Physics.

Hendin, O., Horn, D., and Tsodyks, M. V. (1998). Associative memory and segmentation in an oscillatory neural model of the olfactory bulb. *Journal of Computational Neuroscience, 5(2)*, 157-169.

Hopfield, J. J. (1982). Neural networks and physical systems with emergent collective computational abilities. *Proceedings of the National Academy of Sciences*, *79*, 2554-2558.

Hopfield, J. J. (1984). Neurons with graded response have collective computational properties like those of two-state neurons *Proceedings of the National Academy of Sciences*, *81*, 3088-3092.

Horn, D. and Ruppin E. (1995). Compensatory mechanisms in an attractor neural network model of schizophrenia. *Neural Computation, 7(1)*, 182-205.

Hummel, R. A. and Zucker, S. W. (1983). On the foundations of relaxation labeling processes. *IEEE PAMI*, *5*, 267-287.

Marr, D. and Poggio, T. (1976). Cooperative computation of stereo disparity. *Science*, *194*, 283-287.

Mathis, D. (1997). *A computational theory of consciousness in cognition.* Unpublished Doctoral Dissertation. Boulder, CO: Department of Computer Science, University of Colorado.

Mathis, D., and Mozer, M. C. (1996). Conscious and unconscious perception: A computational theory. In G. Cottrell (Ed.), *Proceedings of the Eighteenth Annual Conference of the Cognitive Science Society* (pp. 324-328). Erlbaum.

McClelland, J. L. and Rumelhart, D. E. (1981). An interactive activation model of context effects in letter perception: Part I. An account of basic findings. *Psychological Review, 88*, 375-407.

McClelland, J. L. and Rumelhart, D. E. (1985). Distributed memory and the representation of general and specific information. *Journal of Experimental Psychology: General, 114(2)*, 159-188.

McRae, K., de Sa, V. R., and Seidenberg, M. S. (1997) On the nature and scope of featural representations of word meaning. *Journal of Experimental Psychology: General, 126(2)*, 99-130.

Movellan, J. (1990). Contrastive hebbian learning in interactive networks. *Neural Computation.*

Mozer, M. C., Sitton, M., and Farah, M. (1998). A superadditive-impairment theory of optic aphasia. In M. I. Jordan, M. Kearns, and S. Solla (Eds.), *Advances in Neural Information Processing Systems 10.* MIT Press.

Neal, R. M. and Hinton, G. E. (1998). A view of the EM algorithm that justifies incremental, sparse, and other variants. In M. I. Jordan (Ed.), *Learning in Graphical Models.* Kluwer Academic Press.

Noelle, D. C. and Zimdars, A. L. (1999). Methods for learning articulated attractors over internal representations. In M. Hahn and S. C. Stoness (Eds.), *Proceedings of the 21st Annual Conference of the Cognitive Science Society.* Hillsdale, NJ: LEA.

Pineda, F. J. (1987). Generalization of back propagation to recurrent and higher order neural networks. *Proceedings of IEEE Conference on Neural Information Processing Systems.*

Plaut, D. C., McClelland, J. L., Seidenberg, M. S., and Patterson, K. (1996). Understanding normal and impaired word reading: Computational principles in quasi-regular domains. *Psychological Review, 103(1)*, 56-115.

Plaut, D. C. and Shallice, T. (1994). *Connectionist modelling in cognitive neuropsychology: A case study.* Hove, England UK: Erlbaum.

Kay, L.M., Lancaster, L.R., and Freeman W.J. (1996). Reafference and attractors in the olfactory system during odor recognition. *Int J Neural Systems, 7(4)*, 489-95.

Redish, A. D. and Touretzky, D. S. (1998). The role of the hippocampus in solving the Morris water maze. *Neural Computation, 10(1)*, 73-111.

Rodrigues, N. C., and Fontanari, J. F. (1997). Multivalley structure of attractor neural networks. *Journal of Physics A (Mathematical and General), 30*, 7945-7951.

Samsonovich, A. and McNaughton, B. L. (1997) Path integration and cognitive mapping in a continuous attractor neural network model. *Journal of Neuroscience, 17(15)*, 5900-5920.

Saul, L.K., Jaakkola, T., and Jordan, M.I. (1996). Mean field theory for sigmoid belief networks. *Journal of AI Research, 4*, 61–76.

Tabor, W., Juliano, C., Tanenhaus, M. K. (1997). Parsing in a dynamical system: An attractor-based account of the interaction of lexical and structural constraints in sentence processing. *Language and Cognitive Processes, 12*, 211-271.