Bayesian Inference, Generative Models, and Probabilistic Computations in Interactive Neural Networks*

James L. McClelland
Stanford University

January 6, 2013

This tutorial provides an introduction to several key concepts relatated to the computations performed by interactive neural networks. Many of these concepts are relied on in the development of an new version of the interactive activation (IA) model of McClelland & Rumelhart (1981) called the multinomial interactive activation (MIA) model (Mirman, Bolger, Kaitan & McClelland, 2013). The IA model might be viewed as a simplified model of interactive processing within perceptual and memory networks, and indeed models based on the IA model and related interactive activation and competition networks (McClelland, 1981) are widespread in psychological research. The development here is intended to connect the intuitive principles of the original IA model with Bayesian ideas, showing how a variant of the original model provides a system for principled probabilistic inference similar to that envisioned in a precursor to the interactive activation model by Rumelhart (1977) and systematized by Pearl (1982). The ideas are also closely related to several of the ideas underlying deep belief networks (Hinton & Salakhutdinov, 2006).

Taken together, these models provide a bridge between neurophysiological ideas and cognitive theories, and between probabilistic models of cognition and process-oriented connectionist or parallel-distributed processing models. Thus, this tutorial may prove useful as an introduction for those interested in understanding more about the relationship between a simple form of Bayesian computation and both real and artificial neural networks.

We begin by presenting Bayes' formula as a tool for inferring the posterior probability that some hypothesis is true, given prior knowledge of

*Draft: send comments to mcclelland@stanford.edu.

certain probabilistic quantities and some evidence[1] This part of the presentation starts with the case of two mutually exclusive and exhaustive hypotheses and a single source of evidence, and shows how Bayes' formula follows from the definition of conditional probability. We then extend the formula to cover cases involving an arbitrary number of mutually exclusive and exhaustive hypotheses and to cases involving more than one element of evidence, introducing the concept of conditional independence. We then develop the idea of a generative model within which the quantities needed to infer posterior probabilities can be seen as representing parameters of a causal process that generates the inputs to a perceptual system.

We next consider how Bayesian inference can be carried out by a neural network. In particular, we observe how the softmax and logistic activation functions often used in neural networks can produce outputs corresponding to posterior probabilities, provided that the biases and connection weights used in producing these outputs represent the logarithms of appropriate probabilistic quantities.

With the above background, we then describe how bottom-up and top-down information can be combined in computing posterior probabilities of letters presented in context, in accordance with Bayes' formula and the generative model assumed to underlie the perceptual inputs to the multinomial interactive activation model. We describe three procedures by which such posteriors (or samples from them) can be computed - one that is completely non-interactive (appearing to accord with the proposals of Massaro (1989 and elsewhere) and of Norris and McQueen (2008), and two that involve bi-directional propagation of information, as in the original interactive activation model (McClelland & Rumelhart, 1981). One of these procedures computes these posteriors exactly, and relates to proposals in Rumelhart (1977) and Pearl (1982). The other samples from the posterior, using Gibbs Sampling as in the Boltzmann machine (Hinton & Sejnowski, 1983); this is the approach taken in the Multinomial Interactive Activation Model. The approach is also related to the modeling framework underlying Deep Belief Networks, which we consider briefly at the end of the tutorial.

As can be seen from the citations above, the key ideas reviewed here have been in circulation for about 30 years. These ideas establish an intimate connection between the computations performed by neural networks and computations necessary to carry out correct probabilistic inference. Unfortunately, to my knowledge there has not been extensive recognition of these

[1]Often the word *data* is used instead of *evidence*. Some writers use evidence to refer to quite a different concept.

connections, at last among many researchers working in the psychological and cognitive science disciplines. The presentation draws on an earlier paper with similar goals (McClelland, 1998) and is intended to help provide an intuitive understanding of some of the relevant concepts involved, and of the reasons why certain things are true, without relying on formal proofs.

Using Bayes' Formula to Infer Posterior Probabilities

We begin by reviewing the canonical version of Bayes' formula, expressing the posterior probability that one of two mutually exclusive and exhaustive hypotheses is true given some evidence e in terms of other quantities which we will shortly define:

$$p(h_i|e) = \frac{p(h_i)p(e|h_i)}{p(h_1)p(e|h_1) + p(h_2)p(e|h_2)} \tag{1}$$

In this expression, $p(h_i)$ corresponds to the prior probability that hypothesis i is true, where h_i could be hypothesis 1 or hypothesis 2. $p(e|h_i)$ corresponds to the probability of the evidence given that hypothesis i is true, and $p(h_i|e)$ corresponds to the posterior probability of hypothesis i given the evidence. The expression is often called "Bayes' law", or "Bayes' rule", although some use "Bayes' rule" for a formulation that expresses the ratio of the posterior probability of h_1 to h_2. Bayes' rule in that form is easily derived from Bayes' formula and *vice versa*. The formula is also sometimes described as "Bayes' Theorem", but we will use that phrase to refer to the proof of the validity of the formula, rather than the formula itself.

As an example (from the Wikipedia entry on *Bayes' theorem*), suppose a friend of yours meets a person with long hair. What is the probability that this person is a woman? Our two possible hypotheses here are that the person is a woman or that the person is a man. We treat them as mutually exclusive and exhaustive — that is, a person must be either a man or a woman; there are no other possibilities, and the person cannot be both a man and a woman at the same time. The evidence e is that the person has long hair.

Bayes' formula allows us to calculate the answer to this question, as long as some additional relevant facts are known. First, we need to know the overall probability that a person your friend might meet is a woman. We could call $p(h_1)$, but to aid maintaining contact with the example, we'll call it $p(W)$. Since we have assumed that the only other possibility is that the person is a man, the probability that the person is not a woman $p(\bar{W})$ is

equal to the probability that the person is a man, $p(M)$. From this it follows that $p(W) + p(M) = 1$, and that $p(M) = p(\bar{W}) = 1 - p(W)$.

The quantity $p(W)$ corresponds to a given or assumed quantity corresponding to the overall probability that a person your friend might meet is a woman. This quantity is often called the *prior*, a usage that makes sense if our goal is to use evidence to update our beliefs about the probability that a person your friend might meet is a woman once we observe the particular person's gender. Here, we are just using this quantity as a premise in an inference process. Nevertheless, writers often use the term prior when describing such terms, and we will often do so here. Another phrase that is sometimes used is *base rate*. Humans often neglect base rates in carrying out probabilistic inference when given probabilistic information in explicit form. When the base rate is low, this can lead to an over-estimate of the posterior probability. It might be noted that there could be uncertainty about the prior or base rate. This is certainly true, and indeed, the question that the Reverend Bayes was primarily interested in was how to use evidence to update ones beliefs about such probabilities. This is a rich and important topic, but it is not the one we are examining here. Instead we are considering the simpler problem of using a set of known probabilistic quantities to infer another probabilistic quantity, the probability that the hypothesis would be true in a particular instance, given some evidence.

In addition to knowledge of the prior probability of the hypotheses, $p(h_1)$ and $p(h_2)$, we also must know the probability of observing the evidence when each hypothesis it true. In our example, we need to know the probability of long hair when the person is a woman (for our example, $p(L|W)$ or more generally $p(e|h_1)$), and also the probability of long hair when the person is a man ($p(L|M)$ or more generally, $p(e|h_2)$). Here, too, there could be considerable uncertainty. However, as with the prior, we will treat these as quantities that are known, and proceed from there.

Using these quantities, we can plug them into Eq. 1 to calculate $p(W|L)$, the probability that the person your friend met is a woman given that the person had long hair. The expression below replaces the abstract variables h_1 and h_2 from Eq. 1 with W and M, and replaces the abstract variable e with the L for long hair, to connect the various quantities in the expression to the relevant conceptual quantities in the example:

$$p(W|L) = \frac{p(W)p(L|W)}{p(W)p(L|W) + p(M)p(L|M)}$$

Let's plug in some actual numbers. If the overall probability of your friend meeting a woman, $p(W)$, is .5; the probability of a woman having long hair

$p(L|W)$ is .8; and the probability of a man having long hair is $p(L|M)$ is .3, then (relying on $p(M) = 1 - p(W) = .5$), we obtain:

$$p(W|L) = \frac{.5 * .8}{.5 * .8 + .5 * .3} = \frac{.8}{.8 + .3} = \frac{.8}{1.1} = .727$$

As an exercise, the reader can explore what happens to the result when one of the relevant quantities changes. What if $p(L|M)$ goes down to .01? In a world where few men have long hair we get a much stronger conclusion. On the other hand, what if $p(L|M) = .8$? You should see that in this case we learn nothing about the person's gender from knowing the person has long hair. Now, what about the prior or base rate, $P(W)$? We have assumed that a person your friend might meet is equally likely to be a woman or a man, but what if instead $p(W)$ is only .1 — this might happen, for example, if the people your friend meets are all computer science majors. Using our initial values for the likelihoods $p(L|W) = .8$ and $p(L|M) = .3$, you should find that the posterior probability that the person is a woman is less than .3. If you neglected the base rate, you might overestimate this probability.

As a second exercise, the reader should be able to calculate $p(W|S)$, the probability that a person your friend met is a woman given that the person had *short* hair, given specific values for $p(L|W)$, $p(L|M)$ and $p(W)$. Use .8, .3, and .5 for these quantities. What gender should we guess to maximize the probability of being correct if we were told that a person your friend met had short hair? Assume for this example that each person either has short hair or long hair — that is, that short and long are mutually exclusive and exhaustive alternatives. As before, also assume that male and female are mutually exclusive and exhaustive alternatives.

Bayes' formula can easily be applied to cases in which the two hypotheses under consideration are the hypothesis that some proposition is true and the hypothesis that the proposition is false. For example, we might want to dermine whether a person is French or not. In this case, our hypotheses could be 'Person X is French' and 'Person X is not French', where no specific alternative hypothesis is specified. Here it is natural to use h for the positive case and $\bar{h}$ for the negative case, and to rewrite the formula as:

$$p(h|e) = \frac{p(h)p(e|h)}{p(h)p(e|h) + p(\bar{h})p(e|\bar{h})}$$

Given that h and $\bar{h}$ as mutually exclusive an exhaustive, $p(\bar{h}) = 1 - p(h)$, so we can also write our formula as:

$$p(h|e) = \frac{p(h)p(e|h)}{p(h)p(e|h) + (1 - p(h))p(e|\bar{h})} \tag{2}$$

It is also worth noting that the posterior probabilities sum to one: $p(h|e) + p(\bar{h}|e) = 1$, so $p(\bar{h}|e) = 1 - p(h|e)$. Thus, the evidence simultaneously informs us about the posterior probability that h is true, and that h is false.

Remark. Clearly, Bayes' formula only gives valid results if the quantities that go into the calculation are accurate. It could be wrong to assume that human perception always relies on the correct values of these quantities. One could propose that human perceivers rely on estimates of such quantities, and that these may differ from their actual values. A further point is that an experimenter might generate inputs according to a protocol that is not fully consistent with the knowledge perceivers rely on to make perceptual inferences. In that case, if the estimates perceivers rely on are not altered to match the protocol used in the experiment, the inferences could be invalid, and therefore not optimal under the conditions of the experiment. For example, a perceiver in a word identification experiment might rely on estimates of each word's probability of occurrence based on its frequency of occurrence in past experience. However, an experimenter might choose words from a word list without regard to their frequency. Under these conditions, use of a words' frequency to represent its probability of occurrance would be invalid. Many perceptual 'biases' or 'illusions' can be explained as resulting from the use of estimates of probabilistic quantities that may be valid (or approximately valid) in the real world, but are not valid within the context of a psychology experiment. If such knowledge were wired into the connections amoung neurons in a perceiver's perceptual system, as it is assumed to be in the interactive activation model, it might not be easily discarded and replaced with other values.

Decision Policies

So far, we have shown how to calculate a posterior probability, but we have not discussed what one might actually do with it. In many situations, we may simply want to take note of the posterior probability — in the case of our first example above, we might not wish to reach a definite conclusion, since the evidence is far from conclusive. However, often a choice between the alternatives is required. There are two possibilities that are often considered: One policy tries to pick the best response, that is, the one that maximizes the probability of being correct, while the other generates responses probabilistically, according to the posterior probability.

The first policy is called maximizing, and corresponds to choosing the

alternative with the largest posterior probability. Formally, we could write:

$$Choice = \text{argmax}(p(h_1|e), p(h_2|e))$$

where the **argmax** function returns the index of the hypothesis with the largest posterior probability. In our example, with the priors $p(W) = .5$, $p(L|W) = .8$ and $p(L|M) = .3$, we calculated that $p(W|M) = .727$. Following this policy, we would conclude that the person is a woman given that the person has long hair.

The second policy is called probability matching. Under this policy, decision makers' choices would vary from trial to trial with the same evidence, but would occur with a probability that matches the posterior probability. Formally, we would write this as:

$$p(Choice = i) = p(h_i|e)$$

One of these two policies is better than the other, in the sense that one maximizes the probability of choosing the correct answer. If you would win a doller for guessing right and loose a dollor for guessing wrong, which of these policies should you chose? Surprisingly, in many cases, the behavior of humans and other animals appears closer to matching rather than maximizing, but there are situations in which people clearly do maximize (Green et al., 2010). There are worse policies, including one that children around 5 sometimes use in explicit outcome guessing tasks, namely alternating choices, regardless of the probability of each of the two outcomes (Derks and Paclisanu, 1967).

Bayes' Theorem: Bayes' Formula Follows from the Definition of Conditional Probability

So far, we have used Bayes' formula without considering why it is true. Here, we will show that the validity of the formula follows from the definition of conditional probability. We have already used the concept of conditional probability. Here we will review its definition, and then use it to derive Bayes' formula.

The conditional probability of some event a given some other event b, written $p(a|b)$, is defined as the ratio of the probability of both a and b, $p(a\&b)$ to the probability of b, $p(b)$:

$$p(a|b) = \frac{p(a\&b)}{p(b)}$$

The definition can be read as defining conditional probability $p(a|b)$ as the proportion of the times when b occurs that a also occurs. Let's relate this to our case, letting e correspond to a and h correspond to b:

$$p(e|h) = \frac{p(e\&h)}{p(h)} \tag{3}$$

In our case, if 50% of the people your friend might meet are women, and 40% of the people your friend might meet are women with long hair, then the probability of long hair given that the person is a woman – or equivalently, the proportion of women who have long hair – would be $.4/.5 = .8$, the value we already used in our example.

Now we can also use the definition of conditional probability to express $p(h|e)$, letting e correspond to b and h correspond to a:

$$p(h|e) = \frac{p(e\&h)}{p(e)} \tag{4}$$

Bayes' formula can now be derived from the fact that $p(e\&h)$ occurs in the definition of both $p(e|h)$ and $p(h|e)$. To derive it, we multiply both sides of Eq. 3 by $p(h)$ to obtain:

$$p(e\&h) = p(h)p(e|h)$$

. For our example, this corresponds to the fact that the proportion of people who have long hair and are women is equal to the proportion of all people who are women, times the proportion of women who have long hair.

We can now replace $p(e\&h)$ in Eq. 4 with $p(h)p(e|h)$ to obtain:

$$p(h|e) = \frac{p(h)p(e|h)}{p(e)}$$

This can be stated: The probability of some hypothesis h being true given some evidence e is equal to the prior probability of the hypothesis, $p(h)$, times the probability of the evidence, given the hypothesis, divided by the overall probability of the evidence $p(e)$.

It remains only to note that the denominator, $p(e)$ is equal to $p(e\&h) + p(e\&\bar{h})$. That is, the total probability of situations in which e is true is the sum of the probabilities of two situations, one in which e is true and the hypothesis h is also true, and another in which e is true and the hypothesis is false. This exhausts the cases in which e is present, given that all persons with long hair must either be women or not. Using the fact that $p(a\&b) =$

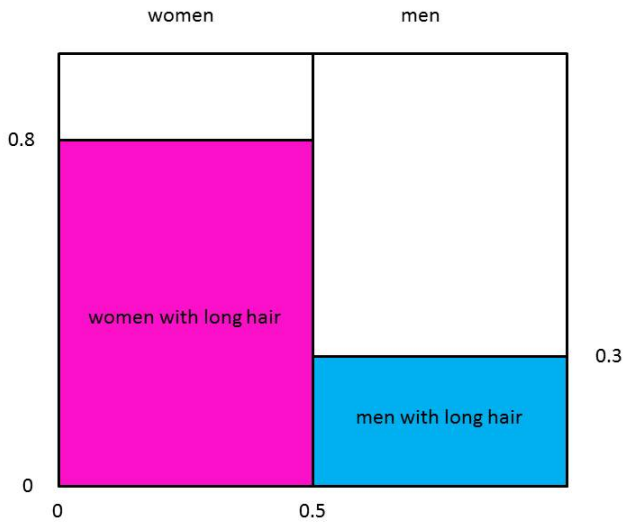

Figure 1: Graphical depiction of posterior probability based on relative area.

$p(b)p(a|b)$ twice more, applying it to both $p(e\&h)$ and to $p(e\&\bar{h})$ we finally obtain:

$$p(h|e) = \frac{p(h)p(e|h)}{p(h)p(e|h) + p(\bar{h})p(e|\bar{h})}$$

and from $p(\bar{h}) = 1 - p(h)$, we can then obtain Eq. 2. Of course the same all works out for cases in which we have two mutually exclusive and exhaustive hypotheses called h_1 and h_2 as in the version shown in Eq. 1, as well.

Figure 1 gives a graphical representation of the posterior probability of a hypothesis constructed by partitioning a square with sides of length 1. We use the horizontal dimension to partition the square into two parts, one corresponding to the overall probability that a person your friend might meet would be a woman, and the other corresponding to the remaining area of the square, corresponding to the probability that the person your friend might meet would be a man. Restating, the areas of these two parts correspond to $p(W)$ and $p(M)$ respectively. Then, we partition the region corresponding to women into two parts along the vertical axis at the point $y = p(L|W)$. This divides the total probability that the person is a woman into two parts, one whose area corresponds to the probability that the person is a woman

and has long hair, and one corresponding to the probability that the person is a woman and does not have long hair. Likewise we partition the region corresponding to men into two parts along the vertical axis at the point $y = p(L|M)$. This gives us two more rectangles, one whose area corresponds to the probability that the person is a man and has long hair, and the other corresponding to the probability that the person is a man and does not have long hair. The area of each resulting rectangle is a joint probability as well as the product of a prior and a conditional probability. The posterior probability $p(L|W)$ is the ratio of the area of the rectangle corresponding to women with long hair to the area corresponding to all persons with long hair, which in turn corresponds to the sum of the areas of the two shaded rectangles.

To fix your understanding of these ideas, you could draw an approximate version of this figure for the case in which (i) the overall probability that a person your friend might meet is a woman is .25; (ii) the probability of a woman having long hair is .75; and (iii) and the probability of a man having long hair is .25. Inspecting the relevant subrectangles withing the unit rectangle, you should be able to estimate the probability that the a person your friend meets is a woman, given that she has long hair. You do this by noting the area corresponding to the probability of being a woman and having long hair, and comparing that to the area corresponding to the probability of being a man and having long hair. Given that these areas are about equal, what is the probability that a person with long hair is a woman?

Multiple Alternative Hypotheses

Note that we have thus far considered a case in which there are only two possible hypotheses: Either the person my friend met was a woman or the person was a man. Now let's suppose we have many alternative hypotheses $\{h_i\}$, and we are trying to determine the posterior probability of each given some evidence e. One example arises if we are trying to determine the identity of a letter given one of its features. For example, in the font used by Rumelhart & Siple (1974), and in the Multinominal Interactive Activation Model, one of the features (which we'll call F_{ht} is a horizontal line segment at the top of a letter-feature block (See Figure 2). Some letters have this feature. For example, the letter T has it and the letter U does not. Treating these statements as absolutes, we could state $p(F_{ht}|T) = 1$ and $p(F_{ht}|U) = 0$. However, let us allow for the possibility of error, so that with a small probability, say .05, feature values will be registered incorrectly.

Figure 2: The line segments used by Rumelhart & Siple (1974) and the letters composed from these segments.

Then $p(F_{ht}|T) = .95$ and $p(F_{ht}|U) = .05$. Now, suppose we want to calculate $p(T|F_{ht})$. For each letter, l_i we would need to know $p(F_{ht}|l_i)$ and we would also need to know the prior probability of occurrence of each letter as well. Given this information, the overall formula for the posterior probability now becomes:

$$p(T|F_{ht}) = \frac{p(T)p(F_{ht}|T)}{\sum_{i'} p(l_i)p(F_{ht}|l_i)}$$

Note that the summation in the denominator runs over all possible letters, including T. In general, the probability that a particular hypothesis h_i is correct given a specific element of evidence e can be written:

$$p(h_i|e) = \frac{p(h_i)p(e|h_i)}{\sum_{i'} p(h_{i'})p(e|h_{i'})}$$

The indexing scheme is potentially confusing: Here and elsewhere, we use a bare single letter such as i to index a specific item or hypothesis of interest and a primed version of the same letter such as i' to index all of the items or hypotheses, including i.

Several points are worth making here. First, under the assumption that the display always represents one of the letters, the sum of the quantities $p(l_i)$ is equal to 1. If there are 26 letters, there are only 25 independent letter probabilities, since the probability of the last one must be equal to 1 minus the sum of the probabilities of all of the others. In general, if there are N mutually exclusive items, there are only $N - 1$ degrees of freedom in the values of their prior probabilities. The last degree of freedom is associated with the fact that the sum of the prior probabilities is equal to 1.[2]

[2]It is worth noting that, if one tabulated counts of occurrences of letters in a randomly

Another point is that when there is only one hypothesis of interest - say, is the item a T or not - all of the other possibilities can be lumped together and we have:

$$p(T|F_{ht}) = \frac{p(F_{ht}|T)p(T)}{p(F_{ht}|T)p(T) + \sum_{i' \neq T} p(F_{ht}|l_{i'})p(l_{i'})}$$

where the summation in the denominator runs over all possibilities other that T of terms corresponding to the product of the prior and the likelihood. This is a generalization of Eq. 2, previously given, with $\sum_{i' \neq T} p(F_{ht}|l_{i'})p(l_{i'})$ playing the role of $p(F_{ht}|\bar{T})p(\bar{T})$.[3]

A third point is that we may want to include the possibility that the observed feature arose from some underlying cause other than the intent to produce one of the known letters. We can include this as an additional hypothesis, if we also provide the probability that the feature value arises from this other cause. In this case the sum of the probabilities of the enumerated causes is less than one, with the other causes consuming the remainder of the total probability. Then we can write Bayes Formula as:

$$p(h_i|e) = \frac{p(h_i)p(e|h_i)}{\sum_{i'} p(h_{i'})p(e|h_{i'}) + p(o)p(e|o)}$$

In psychological models, e.g., the Generalized Context Model of Categorization, Nosofsky, (1994) and the logogen model of word recognition, Morton, (1969), the elements of the expression $p(o)p(e|o)$ are not separately estimated, and are lumped together using a constant, sometimes represented by the letter C.

Multiple Elements of Evidence and Conditional Independence

In general, when we are attempting to recognize letters or other things, there may be more than one element of evidence (e.g. more than one feature) at a time. How can we deal with such situations? A first step is to generalize Bayes' formula by using a likelihood term that encompasses all of the evidence. For example, we might have evidence that there's a horizontal

chosen book, the counts could be considered to be independent quantities. In this case, the total count of letters in the book would be the sum of the counts for all of the letters. If we were to be given this total count, we could infer the count for any given letter from the counts for the others letters and the total count. Either way, there are N+1 quantities (each count and the sum) but only N independent quantities.

[3]Note that in general, $\sum_{i' \neq T} p(F_{ht}|l_{i'})p(l_{i'})$ may not be equivalent to $\sum_{i' \neq T} p(F_{ht}|l_{i'}) \sum_{i' \neq T} p(l_{i'})$.

feature across the top of a feature array and a vertical segment down the middle. We could then make use of expressions such as $p(F_{ht}\&F_{vm}|T)$ to represent the probability of observing both of these features, given that the letter in question is T.

A problem that arises here is that the number of possible combinations of elements of evidence can grow large very quickly, and so it becomes intractable to imagine that a perceiver knows and represents all of these probabilities. Luckily, there is a condition under which the computation of the values of such expressions becomes very simple. This condition is known as 'conditional independence', which can be defined for two or more events with respect to some other, conditioning event. For two events, conditional independence is defined as follows:

Definition of Conditional Independence. Elements of evidence e_1 and e_2 are conditionally independent given condition c if the probability of both pieces of evidence given c, $p(e_1\&e_2|c)$, is equal to the product of the separate conditional probabilities $p(e_1|c)$ and $p(e_2|c)$ for each element of the evidence separately.

We can generalize this to an ensemble of any number elements of evidence e_i and express the relationship succinctly: Conditional independence of an ensemble of n elements of evidence e_i given some condition c holds when

$$p(e_1\&e_2\&...\&e_n|c) = \prod_j p(e_j|c)$$

.

Considering our example, we can consider the presence of a horizontal across the top, F_{ht}, and the presence of a vertical down the middle, F_{vm}. These would be conditionally independent given that the underlying letter was in fact intended to be a T if it were true of the world that error entered into the registration of each of these two features of the letter T independently.

We can now write a version of our formula for inferring posterior probabilities under the assumption that conditional independence holds for all elements of evidence e_j conditioned on all of the hypotheses h_i:

$$p(h_i|e) = \frac{p(h_i)p(e_j|h_i)}{\sum_{i'} p(h_{i'}) \prod_j p(e_j|h_{i'})}$$

We are still relying on many probabilistic quantities, but not as many as we would have to rely on if we separately represented the probability of each feature combination conditional on each hypothesis.

Remark. Clearly, the assumption of conditional independence is unlikely to be exactly correct. However, it is hard to imagine proceeding without it. One way of alleviating the concern that relying on this assumption will lead us astray is to note that in cases where the occurrence of elements of evidence is highly correlated (even after conditioning on hypotheses), we might treat these elements as a single element, instead of as separate elements. Maybe that is what features are: clusters of elements that have a strong tendency to co-occur with each other. Another response to this situation would be to note that any explicit probability model involving sets of explicit hypotheses and elements of evidence is unlikely to be exactly correct; an implicit probability model of the kind embodied in a Deep Belief Network (Hinton & Salakhutdinov, 2006) is likely to be a better model for naturalistic stimuli. Words spelled using letters and their features as in the Rumelhart font are not completely natural, since they actually do consist of discrete units (letters) and these in turn consist of independent sub-units (letter features). This allows for the possibility of validly characterizing displays of such features in terms of a process in which conditional independence of features actually exactly.

A Generative Model of Feature Arrays

Consider the following description of how displays of letter features registered by a perceiver might be generated. An experimenter selects a letter to display from the alphabet with probability $p(l_i)$, which for now we will take to be simply $1/26$ for each letter, and then generates a feature array as follows. Each letter has a set of correct feature values. For example, for T, the feature F_{ht} is present, the feature F_{vm} is present, and the feature F_{hb}, a horizontal line across the bottom, is absent (for simplicity, we will just consider these three features for now). However, when the actual feature array is generated, there is some small probability that each feature will not be generated correctly. The correctness of each feature is separately determined by an independent random process, such as the role of a 20-sided die with a spot on just one side. If the spot comes up, the incorrect value of the feature is displayed. If it does not, the feature is generated correctly. A different instance of such a die is rolled for each feature.

The above is a simple example of a generative process. If features were generated according to this process, then the probabilities of features are conditionally independent, given the letter identities. Note that if the generative process usually works perfectly correctly generating all the correct features, but occasionally hiccups and gets all the features wrong at the same

time, the elements of the evidence would not be conditionally independent. Note also that conditional independence can hold if the probability of feature perturbation is different for different features; this is likely if we think of the perturbation as occurring within the visual system, so that some features are more likely to be mis-registered than others, due to differences in their size, retinal position, or other factors.

Now, the true process generating feature arrays may not be exactly as described, just as the quantities used in Bayesian inference may not be exactly accurate. However, it is possible that a perceiver might rely upon the assumption that the features are generated according to such a process, or his perceptual system might be constructed to contained units and connections what caused it to treat the features as though they had been generated according to such a process.

The adoption of a generative model in which feature values are perturbed independently can be construed as constituting an assumption about the actual generative process, or alternatively as constituting an assumption about the model of the generative process that is utilized by the perceiver in a letter perception experiment. Such a model could be false, or only approximately true, and still be used by a perceiver. A further possibility is that the true model used by the perceiver is more complex, but that the assumption that the perceiver uses such a model provides a good approximation to the true model being used by the perceiver.

Adopting this generative model, it will be helpful in our later development to write an expression we will call the Support (S_i) for a given alternative hypothesis h_i, given a set of elements of evidence $e = e_1, e_2, \ldots$ as follows:

$$S_i = p(h_i) \prod_j p(e_j|h_i)$$

For our example, the h_i correspond to the different possible letter hypotheses and the e_j correspond to the elements of the evidence. We will describe this overall support as consisting of the product of two terms, the prior $p(h_i)$ and the likelihood $p(e|h_i)$, which under the generative model is equal to the product of terms that might be called the element-wise likelihoods of each element of the evidence.

With this expression for the support of hypothesis i in the presence of evidence e, we can write Bayes' formula as:

$$p(h_i|e) = \frac{S_i}{\sum_{i'} S_{i'}}$$

As before, i' is an index running over all of the alternative hypotheses, including hypothesis i. Readers familiar with the Luce (1959) choice rule will notice that this expression corresponds to Luce's rule, with the S_i corresponding to the response strengths associated with the different choice alternatives.

As an exercise, consider a letter microworld with just the three features we have considered so far and just the letters T, U and I. Assume that according to the generative model, each letter is equally likely $p(T) = p(U) = p(I) = 1/3$; assume that the probability of each feature occurring for each letter is as given in the table below, where h stands for a high probability (let's say .95) and l for a low probability (.05). We assume that features cannot be both present and absent, so $l = 1 - h$. Assuming actual features are generated in a conditionally independent manner, what is the probability that the underlying letter was a T given that the following features were detected: Horizontal at top *present*, Vertical at middle *absent*, Horizontal at bottom *absent*. Although these features don't match the full set of features of letter T, the underlying letter is more likely to be a T than a U or an I. Why?

Letter	Feature					
	Horiz at Top		Vert thru Middle		Horiz at Bottom	
	present	absent	present	absent	present	absent
T	h	l	h	l	l	h
U	l	h	l	h	h	l
I	h	l	h	l	h	l

One minor additional element may now be considered. We may ask, what happens if we are not told about one of the elements of the evidence? For example, we are told that the horizontal bar across the top is present and the vertical bar down the center is present but we simply are not told about the horizontal bar across the bottom (perhaps something is blocking our view of that feature in a perceptual display, for example). We would simply use those elements of evidence that we do have, an exclude the elements that are unspecified. Our existing expression already captures this policy implicitly, since when an element of evidence is missing it simply does not show up in the ensemble of elements e_j. However, it will prove useful to capture this case by elaborating the expression for S above by including in this expression additional information specifying whether particular items of evidence are or are not present. A nice way to do this is to have a binary vector indicating whether the element of evidence is present or not. We have

six possible elements of evidence in our example, as enumerated above. If we are given Horizontal at Top present, Vertical thru Middle absent, this vector would become: $v = 1\ 0\ 0\ 1\ 0\ 0$. Then we would obtain the same results as before by writing S_i as follows:

$$S_i = p(h_i) \prod_j p(e_j|h_i)^{v_j}$$

Where $\prod_j$ represents the product over all possible elements, and v_j is equal to 1 for elements of evidence that are present, or 0 otherwise. Note that elements that are absent have no effect since for any non-zero x, $x^0 = 1$, and for all p, $p \cdot 1 = p$.[4] Note that the model we use here distinguishes between evidence of absence ('No horizontal bar is present at the bottom of the feature array') and the absence of evidence ('We do not know whether or not a bar is present at the bottom of the feature array'). In many cases, it is useful to distinguish between these two situations.

Remark: Using $p(e|h)$ to infer $p(h|e)$. It is worth noting that we use knowledge of the probability of evidence given a hypothesis to infer the probability of a hypothesis given evidence. At first, this may seem counter-intuitive. Why don't we just store the value of $p(h|e)$, rather than always having to compute it. A similar counter-intuition arises in thinking about the 'bottom-up' support for letter hypotheses by feature evidence. One might think that the effect of a feature's presence on the probability of a letter should depend on the probability of the letter given the feature, and not the other way around. The resolution of this counter-intuition depends on noticing that the posterior probabilities are not directly defined in the generative model, while the prior and the $p(e|h)$ terms are. Indeed, the posterior probability that a hypothesis is true depends on the entire ensemble of quantities in the generative model and the particular ensemble of elements of evidence that may be present, while the $p(h)$ and $p(e|h)$ values can be stable and independent. To contemplate this in a specific context, let us return to the question of the probability that a person is a woman, given that she has long hair. This quantity depends on three other quantities: The overall probability that a person is a woman; the probability that a woman has long hair; and the probability that a man has long hair. Each of

[4]Strictly speaking, this formula is valid if the causes that could lead to missing evidence - e.g. no information about whether a particular feature is present or absent - are independent of the process that generates the feature values. This would be true if, for example, the probability that an occluder would partially restrict the visibility of an object were independent of the identity of an object.

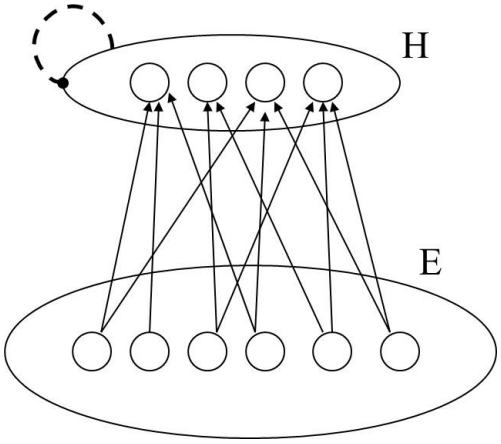

Figure 3: Sketch of a pool of units that can calculate posterior probabilities of patterns represented on its inputs using the **softmax** function. Dashed line signifies lateral inhibition to normalize the activations of units in the pool.

these quantities can be changed independently, without affecting the others, while the probability that a person with long hair is a woman depends on all three. In short, in many contexts at least, it makes sense that we use $p(h)$ and $p(e|h)$ to compute $p(h|e)$.

To summarize the above development, the generalized Bayes formula for the posterior probability of hypothesis h_i, for $i = 1, \ldots, n$ mutually exclusive hypotheses and $j = 1, \ldots, m$ possible conditionally independent elements of evidence:

$$p(h_i|e) = \frac{S_i}{\sum_{i'} S_{i'}} \tag{5}$$

Here S_i stands for the total support for hypothesis h_i, defined as:

$$S_i = p(h_i) \prod_j p(e_j|h_i)^{v_j} \tag{6}$$

Calculating Posterior Probabilities with Connectionist Units Using the Softmax and Logistic Functions

We now develop the idea that the posterior probability calculation just presented can be computed by a group of connectionist processing units, using a function called the **softmax** function. The neural network is illustrated in Figure 3. In this network, each unit corresponds to an hypothesis h_i, and

has a bias term b_i, as well as incoming connections from units outside the ensemble. Each of these outside units indexed by j stands for a possible element of evidence. When the element of evidence is present, the unit will have an activation value a_j equal to 1; when it is absent, its activation will be 0. Each connection to a hypothesis unit from an evidence unit will have a strength or weight represented by the variable w_{ij}.

Concretely, pursuing our example, the units in the pool could correspond to possible letters, each unit's bias term could reflect a perceiver's bias to think the input contains the given letter, and the connection weights could reflect the perceiver's tendency to think the hypothesis is more (or less) likely, when the corresponding element of evidence is present. The pool described corresponds to one of the pools of letter level units in the multinomial interactive activation model, although here we are considering just one such pool in isolation for now, without additional input from the word level.

In our network, as in most neural networks, each unit computes a summed or net input that reflects both its bias and the weighted sum of activations of other units:

$$net_i = bias_i + \sum_j w_{ij} a_j$$

We will now see that if we set the weights and biases to appropriate values, then apply the **softmax** function defined below, the output of the function, represented here as ρ_i, will be equal to the posterior probability of the letter the unit stands for, as expressed by the generalized Bayes formula.

The **softmax** function is:

$$\rho_i = \frac{e^{net_i}}{\sum_{i'} e^{net_{i'}}}$$

The reader should already be able to see that the softmax has some relationship to the generalized Bayes formula. Indeed, as we shall discuss, the expressions e^{net_i} and $e^{net_{i'}}$ correspond to the expressions for S_i and $S_{i'}$ in that equation.

The essential idea is that the bias term and the weights will be chosen to correspond to the logarithms of the quantities that are multiplied together to deterimine the S_i terms. Using the logs of these quantities, we add rather than multiply to combine the influences of the prior and the evidence. The resulting net input term corresponds to the log of the S_i terms defined above. We then reverse the logarithmic transformation at the end of the calculation, using the exponential function.

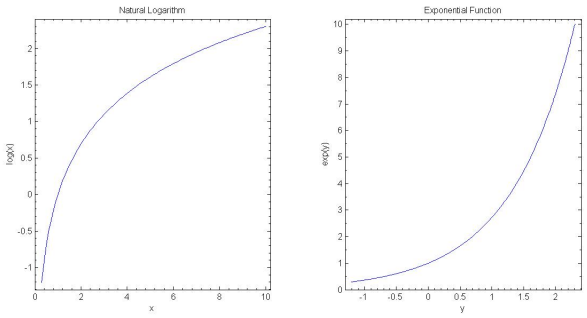

Figure 4: The log and exponential functions.

The analysis relies on several facts about the log and exponential functions that we now review. First, the function $y = \log(x)$ is defined as the function that produces, when applied to its argument x, a number y such that $e^y = x$. Note that log is used here to correspond to the natural logarithm, sometimes written $\log_e$ or ln. The exponential function of y, e^y corresponds to the number e taken to the power y, and is sometimes written $\exp(y)$. Given these definitions, it follows that $\log(e^y) = y$ and $e^{\log(x)} = x$. The graphs of the log and exp functions are shown in Figure 4.

The second important fact is that the log of the product of any number of quantities is the sum of the logs of the quantities:

$$log(a \cdot b \cdot c \cdot \dots) = \log(a) + \log(b) + \log(c) + \dots$$

Similarly, the log of the ratio of two quantities is equal to the difference between the logs of the quantities:

$$\log(a/b) = \log(a) - \log(b)$$

Finally, the log of a quantity to a power is that power times the log of the quantity:

$$\log(a^b) = b \log(a)$$

. There are also useful related facts about exponentials, namely $e^{(a+b+c+\dots)} = e^a \cdot e^b \cdot e^c \cdot \dots$; $e^{(a-b)} = \frac{e^a}{e^b}$; and $e^{(a \cdot b)} = (e^a)^b$.[5]

[5]Those wanting to gain familiarity with these functions can obtain values by reading off of these functions, and check that the above relationships all hold up. For example $\log(2)+\log(3) = \log(2 \cdot 3) = \log(6)$ and $\log(8)-\log(4) = \log(8/4) = \log(2)$, and not $\log(4)$.

With this information in hand, we consider the expression we previously presented for S_i, the support for hypothesis i.

$$S_i = p(h_i) \prod_j p(e_j|h_i)^{v_j}$$

Taking logs, we see that:

$$\log(S_i) = \log(p(h_i)) + \sum_j v_j \log(p(e_j|h_i)) \ .$$

It should now be apparent that the net input as described above would correspond to the log of the support for the hypothesis represented by the unit if: (a) the value of the bias term were set to correspond to the log of the prior probability of the hypothesis; (b) each incoming weight were set to correspond to the log of the probability of the corresponding element of the evidence given the hypothesis; and (c) the activation of the external unit sending activation through the weight were to be equal to 1 when the evidence is present, and 0 otherwise. Stated succinctly in terms of defined quantities: $net_i = \log(S_i)$ if $bias_i = \log(p(h_i))$, $w_{ij} = \log(p(e_j|h_i))$, and $a_j = v_j$.

Now it should be clear that applying the softmax function

$$\rho_i = \frac{e^{net_i}}{\sum_{i'} e^{net_{i'}}}$$

should set the value of the variable ρ_i to be equal to the posterior probability of hypothesis i given the set of elements of the evidence e_j as long as net_i corresponds to $\log(S_i)$ for all i, since $e^{\log(x)} = x$, as noted above. Substituting $\log(S_i)$ and $\log(S_{i'})$ into the softmax function where we find net_i and $net_{i'}$ we will clearly obtain the generalized Bayes formula.

Thus, the neural network in Figure 3, employing the softmax function, calculates posterior probabilities by relying on a non-linear but monotonic function (the exponential fuction) of the sum of a set of terms, one for the prior probability of the hypothesis and one for each of the elements of the evidence.

Why sums rather than products? One might be inclined to ask at this point, why should neural network modelers even bother computing net inputs as additive quantities? Why not compute the posterior probabilities more directly, without ever taking logs? The answer may in part be historical: the original McCulloch-Pitts neuron summed weighted inputs, and if

they exceeded a threshold the neuron's output was set to 1; otherwise the output was 0. This was intended to mimic both real neurons (which fire action potentials if their state of depolarization reaches a critical level) and logic gates (devices then send out a 1 or a 0 based on some logical function of their inputs). The logistic function was adopted largely because it produced a graded rather than a discrete response, and could be differentiated. Only later did the connection to probability become apparent (first reflected, to my knowledge, in Hinton & Sejnowski, 1983). But what about the brain itself? It is common to treat synaptic currents as being summed to determine the neuron's potential, which in turn determines its firing rate. It is possible that addition may be more robust and easier to implement in neurons than multiplication, especially when small probabilities are involved, since noise affecting such quantities can drastically distort the results of multiplying products, and in any case the computations are just as valid when conducted using addition of logarithms rather than multiplication, as long as we have a non-linear activation function like softmax to convert the influences back. Some further relevant observations are provided in the section below.

Maximizing and matching using the neural network

We can imagine a number of policies we might employ in using the ρ_i values as a basis for overt responding. One policy would be to choose the alternative with the largest value of ρ_i; this corresponds to maximizing. Matching would occur if we were to choose alternatives with probability equal to the value of ρ_i. A gradient of possibilities between these extremes can be obtained by introducing a parameter usually called temperature, following the analogy to statistical physics introduced into neural networks research by Hinton & Sejnowski (1983). This usage corresponds to the analogy from physics, in which the temperature determines the degree of randomness in the behavior of elements of the system. In this version of the formula, our expression now becomes:

$$\rho_i(T) = \frac{e^{net_i/T}}{\sum_{i'} e^{net_{i'}/T}}$$

Our previous case corresponds to the situation in which $T = 1$. We can now imagine a policy in which we choose each alternative with probability $\rho_i(T)$, for different values of the T parameter. As T becomes small, the largest net input term strongly dominates, and in the limit as $T \to 0$ our policy converges on maximizing, since $\rho_i(T)$ will approach 1 for the unit with the largest net input and will approach 0 for all other units. As T

becomes large, the $\rho_i(T)$ will all approach $1/N$ where N is the number of alternatives, corresponding to random guessing.

Example. The softmax function can be used to to model response choice probabilities in many situations, under a matching assumption, where the ρ_i correspond to choice probabilities. One case where the model provided an excellent fit arose in an experiment by Saltzman and Newsome (1994). Here a monkey received a visual motion stimulus, corresponding to evidence favoring a particular alternative direction out of eight alternative motion directions. On some trials, the monkey also received direct electrical stimulation of neurons representing motion in a particular direction (treated in the model as another source of conditionally independent evidence). The monkey's choice behavior when both sources of evidence were presented together corresponded well to the predictions of the model. The experimenters estimated quantities corresponding to the bias terms and weights used in the softmax formulation. Although they did not mention Bayesian ideas, these terms could be treated as corresponding to logarithms of the corresponding Bayesian quantities.

Lateral inhibition and effects of noise in the net input. The denominator of the softmax function can be seen expressing a particular form of lateral inhibition, in that strong support for one alternative will reduce the value of ρ_i for another. Some readers may notice that the inhibitory influence a unit exerts on others depends on its net input term (specifically, $e^{net_i/T}$), whereas it is natural to think of the ρ_i as corresponding to the activation of the unit, which is the quantity units are usually thought to transmit to each other, both to exert excitatory between-layer influences and for mutual inhibition. Do units use one variable for mutual inhibition and another to influence outside units? It is certainly a possibility. A computation of this kind could certainly be carried out, say, if the units in our networks corresponded to columns of neurons, in which some engaged in lateral inhibitory interactions while others sent excitatory signals to neurons in other pools. Also, it may be worth noticing that in practice, an iterative computational procedure in which the net input terms build up gradually and the denominator relies on the ρ_i instead of the e^{net_i} terms should converge to the same result, as in the REMERGE model (Kumeran & McClelland, 2012).

It is also possible to view the softmax function as simply describing the outcome of a simple winner-take all process. Suppose we simply allow each unit to compute its net input, subject to Gaussian noise, and adopt the policy of choosing as our response the unit with the largest net input. If the

noise is very small, and the weights and biases correspond to the probabilistic quantities above, then by choosing the unit with the largest net input we will always be maximizing the posterior probability. On the other hand if the noise is sufficiently large, the net input will be effectively swamped by the noise, and choosing the unit with the largest net input will correspond to random responding. With an intermediate amount of noise, the process just described closely approximates choosing alternatives with probability $\rho_i(T)$ as calculated by the softmax function, for some value of the parameter T that depands on the standard deviation of the noise, σ. The relationship between the standard deviation of the noise, σ, and the value of T that approximates the same behavior is continuous and monotonic, and there is a value of σ that corresponds to $T = 1$. Thus, the simple winner take all procedure can carry out probability matching, if the weights and biases have the right values and the noise has the right standard deviation, and can also approximate all policies between maximizing and pure guessing depending on the level of the noise.[6]

The Logistic Function

It is helpful to consider a variant of the scenario described above, for cases in which we have just two mutually exclusive hypotheses. In this case it is possible to use bias terms and weights that allow us to calculate the posterior probability of one of the two hypotheses more directly, using a function called the logistic function — a function very frequently used in setting the activations of units in neural network models. The approach is very natural when h_1 corresponds to the hypothesis that some proposition is true, and h_2 corresponds to the proposition that it is false, but can be applied to any situation in which there are two mutually exclusive and exhaustive alternatives. We consider the calculation of the posterior probability of h_1, noting that the posterior probability of h_2 must be 1 minus this quantity. Specializing the softmax function of this case, we can write

$$\rho_1 = \frac{e^{net_1}}{e^{net_1} + e^{net_2}}$$

[6]It may be interesting to note that what determines how likely it is that the unit with the strongest net input is the most active unit is not the absolute value of the noise but the ratio of the strength of the noise to the signal coming from other units. Given this, the noise might remain constant, while the strength of the net input might build up gradually over time. In this way, as time progressed, we could go from chance performance to matching, and, if signals continue to grow stronger, to maximizing as a function of time spent processing.

where net_1 and net_2 are based on the values of the biases and weights as described above. Dividing the numerator by e^{net_2} and recalling that $e^a/e^b = e^{a-b}$ we obtain:

$$\rho_1 = \frac{e^{net_1 - net_2}}{e^{net_1 - net_2} + 1}$$

Rather than compute each net input term separately and then subtract them, we can instead compute a single net input using biases and weights corresponding to the difference between the corresponding terms in each of these two expressions. That is, we define the combined net input as:

$$net = bias + \sum_j a_j w_j$$

where $bias = bias_1 - bias_2$ and $w_j = w_{1j} - w_{2j}$. Recalling that $\log(a) - \log(b) = \log(a/b)$, we see that if the old biases and weights corresponded to the appropriate Bayesian quantities, the new combined bias term with be equal to $\log(p(h_1)/p(h_2))$ and each new combined weight w_j will be equal to $\log(p(e_j|h_1)/p(e_j|h_2))$.

In terms of a single hypothesis h that is either true or false, the bias term becomes $\log(p(h)/p(\bar{h}))$ or $\log(p(h)/(1-p(h))$ and the w_j becomes $\log(p(e_j|h)/p(e_j|\bar{h}))$. These are quantities often used in discussions of probabilities. The first is called the *log-odds*. The second is the log of the likelihood ratio, although in this case it is the element-specific likelihood ratio, specifying the log of the ratio of the likelihood of a specific element of the evidence when h is true to the likelihood of that same element of the evidence when h is false. The overall log likelihood ratio given n conditionally independent elements of evidence is the sum of these quantities over all of the conditionally independent elements of the evidence.

From this we now can see that the posterior probability that some hypothesis h is true can be expressed as:

$$\rho = \frac{e^{net}}{e^{net} + 1}$$

where the net input is the sum of a bias term equal to the log of the prior odds and each weight on the contribution from each element of the evidence is equal to the element-specific log likelihood ratio. This expression does not look exactly like the logistic function as usually written, but it is equivalent to it. We can produce the usual form of the logistic function by dividing the numerator and the denominator by e^{net}, relying on the fact that $1/e^x = e^{-x}$:

$$\rho = \frac{1}{1 + e^{-net}}$$

This form of the function is used in simulators since it involves calling the `exp()` function only once, but they are both essentially the same function.

To summarize this section: The softmax function can compute according to Bayes' formula using biases and weights corresponding to the logs of key Bayesian quantities, while the logistic function computes according to Bayes' formula using biases and weights corresponding to logs of ratios of these quantities. The minus sign in the exponentiation in the logistic function reflects a simplification of the formula that slightly obscures the relationship to Bayes' formula but makes calculation quicker. It is also worth noting that the softmax and logistic functions could be used to describe the outcome of a process in which one simply chooses the alternative with the largest net input, subject to Gaussian noise.

Logistic Additivity

Here we discuss a characteristic of patterns of data we will call logistic additivity. This is the condition on the relationship between the posterior probability that some binary hypothesis h is true, as we manipulate two independent sources of evidence, under the assumption that the probability of each source of evidence are conditionally independent given h and given $\bar{h}$. It is also a condition on the expected output of the logistic function, given that each source of evidence has an independent, additive effect on the net input variable that is the input to this function. While a little outside the flow of our consideration of neural networks, we are relying on the relationships between exponentials and logarithms as above, and therefore from a mathematical point of view this is a good place for this material to come.

We will say that logistic additivity holds for the effects of two independent sources of evidence on the probability of some outcome when they have additive influences on the **logit** of the probability of the outcome. The logit of a probability p is defined as follows:

$$\text{logit}(p) = \log(p/(1-p))$$

With this expression defined, we can write the condition as:

$$\text{logit}(p(h_1|e_1, e_2)) = b + f_1(e_1) + f_2(e_2)$$

This result is nice for visualization purposes since it says that for for a factorial combination of different levels of e_1 and e_2, we should obtain parallel curves. While we won't develop this point further here, these parallel curves

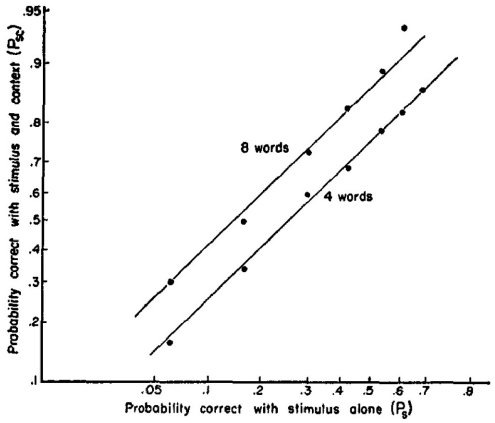

Figure 5: The joint effect of context and stimulus information on probability of identifying a word correctly, displayed on axes where points are spaced according to the logit of the indicated probabilities, from Morton (1969). The x axis corresponds to the logit of the probability of identifying a target word when presented without context; in the experiment (Tulving, Mandler, & Baumal, 1964), this probability was manipulated by using different exposure durations ranging from 0 to 120 msec. Two curves are plotted, one for cases in which an eight-word context was provided (e.g. for the target *raspberries*: "We all like jam made from strawberries and"), and one for the case in which only the last four words of the context was provided. The curves show that the context and stimulus information have additive effects on the logit of the probability of identifying the stimulus correctly.

can be turned into parallel straight lines by appropriate spacing of points along the x axis. In his excellent early analysis of context effects in word recognition, Morton(1969) used this approach. Further details are presented in Figure 5 and the corresponding caption.

We now show how logistic aditivity follows from Bayes formula for the case of two sources of evidence for hypotheses h and $\bar{h}$. We work from the following version of the formula:

$$p(h|e_1, e_2) = \frac{(S/\bar{S})}{1 + (S/\bar{S})}$$

If follows from this that

$$1 - p(h|e_1, e_2) = \frac{1}{1 + (S/\bar{S})} \ .$$

If you do not see this immediately, add the two quantities together — clearly they sum to 1. Dividing the first expression by the second, we obtain:

$$p(h|e_1, e_2)/[1 - p(h|e_1, e_2)] = S/\bar{S}$$

Taking logs of both sides, we obtain:

$$\text{logit}(p(h|e_1, e_2)) = \log \frac{p(h)}{p(\bar{h})} + \log \frac{p(e_1|h)}{p(e_1|\bar{h})} + \log \frac{p(e_2|h)}{p(e_2|\bar{h})}$$

The right-hand side of this equation exhibits logistic additivity, with $\log(p(h)/p(\bar{h}))$ corresponding to b, $\log(p(e_1|h)/p(e_1|\bar{h}))$ corresponding to $f_1(e_1)$, and $\log(p(e_2|h)/p(e_2|\bar{h}))$ corresponding to $f_2(e_2)$.

Working directly from the logistic function we can proceed in a similar vein to arrive at the formula expressing logistic additivity. Given that $\rho = \frac{e^{net}}{e^{net}+1}$ it follows that $1-\rho = \frac{1}{e^{net}+1}$. From these observations, it follows that $\rho/(1-\rho) = e^{net}$, since the denominators cancel. Taking logs of both sides and replacing *net* with its definition we have:

$$\text{logit}(\rho) = bias + a_1 w_1 + a_2 w_2$$

The idea that different sources of evidence – and in particular stimulus and context information – should exhibit logistic additivity was referred to as the *Morton-Massaro Law* by Movellan & McClelland (2001), and is a consequence of the assumptions of both Morton's and Massaro's (e.g., Massaro, 1989) models of how different sources of information are combined. Though neither model was explicitly formulated in Bayesian terms, it should be clear that these models follow from Bayes formula and from the assumption that context and stimulus information are conditionally independent sources of evidence about the identity of an item in context.

Posterior Probabilities from the Generative Model Used in the Multinomial IAM

With the above background, we are finally ready to apply the ideas we have explored so far to the multinomial IAM. The goal of perception, according to this model, is to infer the underlying states of the world that gave rise to observed features – in this case, the goal is to infer the identity of the word and of the four letters that generated the features that reach the input to the model in a trial of a perception experiment. Note that the answer cannot be determined with certainty, since the generative process that produces these features is assumed to be probabilistic. The multinomial IAM assumes that

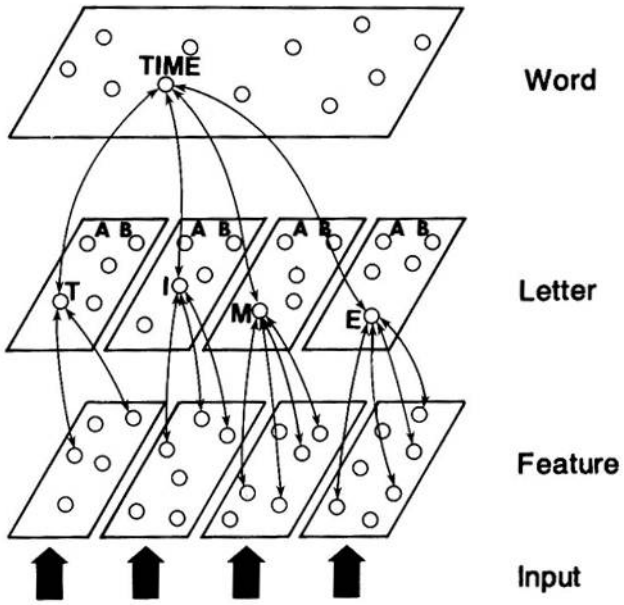

Figure 6: The architecture of the Multinomial Interactive Activation Model. Softmax is used in the model to calculate estimates of posterior probabilities for the word units and for each pool of letter units.

perception produces as its outcome a sample of a possible underlying state of the world that could have generated the observed features, and that alternative possible underlying states are sampled probabilistically, such that the probability of sampling each possible underlying state corresponds to the actual posterior probability that this was the underlying state that generated the observed features, according to the generative model embodied in its architecture and its connection weights. The model also provides a mechanism for doing so, based on a procedure we will describe below. First, we must establish what these posterior probabilities of different underlying states of the world are, given that we have observed a set of feature values in each position as our evidence.

The generative model of the multinomial IAM. The generative model of the multinomial IAM is described in the Mirman *et al.* article and in Khaitan and McClelland (2010). The generative model is thought of

as a characterization of the process that produces the set of input features received by a participant in a letter and word perception experiment. The figure depicting the model is reproduced here in Figure 6 for convenience. Note that the diagram shows some of the units and a small subset of the connections in the actual neural network model, but it can also be seen as corresponding to the generative model that the neural network embodies, with the proviso that the generative process runs strictly from the top down, while the connections in the network run in both directions.

The first step in the generative process is to select a word at random from a list of words that are four letters long, in accordance with a base rate for each word (represented $p(w_i)$); we then generate letters independently in each position with probability $p(l_{j_p}|w_i)$, where we use j_p to represent letter j in position p.[7] Given the above procedure, the probability of a particular word w_i and four letters $\{l_{j_p}\}$ is:

$$p(w_i, \{l_{j_p}\}) = p(w_i) \prod_p p(l_{j_p}|w_i) \ .$$

Now using the letter sampled in each position independently, we sample values for features for each letter. Recall that we treat the set of features as consisting of 14 separate feature dimensions, for each of which there are two explicitly represented possibilities, one that the feature is present and one that it is absent. Independently for each dimension, we select the value for a given feature dimension with probability $p(f_{v_{dp}}|l_{j_p})$.[8]

The generative process has produced a word, a letter in each position, and a value for each feature of each letter. We will call this set of elements a *path* $P_{i,\{j_p\},\{v_{dp}\}}$ of the generative process, and subscript it with the indices of all of the selected elements, one for the word (i), a set of four indices $\{j_p\}$ for the letters, where p runs over the four positions, and the set of 4×14 indices $\{v_{dp}\}$ each specifying the value v (*present*, *absent*) of each dimension d of each position p. The probability of a given path is:

[7] In principle, for each letter in each word we could have a full table of 26 times 4 entries, $p(l_{l_p}|w_i)$, but for simplicity we will assume (as in Mirman *et al*), that $p(l_{j_p}|w_i)$ is the same, and is equal to a fixed parameter value $c_{L|W}$ if l_j is the correct letter in position p of word i and that the remaining probability, $1 - c_{L|W}$, is split evenly among the remaining 25 letters. Thus if $c_{L|W} = .95$, the value of $p(l_{j_p}|w_i)$ for the incorrect letters will be $.05/25 = .002$. Using these numbers, with probability $.95^4 = \tilde{.}81$ all four letters generated from the chosen word will be the correct letters, but with probability $1 - .95^4 = \tilde{.}19$ there will be one or more incorrect letters.

[8] Again for simplicity, we use a single parameter for correct features, $c_{F|L}$; for incorrect features, the corresponding probability is $1 - c_{F|L}$.

$$p(P_{i,\{j_p\},\{v_{dp}\}}) = p(w_i) \prod_p p(l_{j_p}|w_i) \prod_d p(f_{v_{dp}}|l_{j_p}).$$

Simplify the notation slightly, using $p(\{v_{d_p}\}|l_{j_p})$ to represent $\prod_d p(f_{v_{dp}}|l_{j_p})$, this becomes:

$$p(P_{i,\{j_p\},\{v_{dp}\}}) = p(w_i) \prod_p p(l_{j_p}|w_i) p(\{v_{d_p}\}|l_{j_p}). \tag{7}$$

We will refer to this equation later as the *path probability equation*.[9]

We can now consider the posterior probability of a particular combination of unobserved word and letter variables, given an observed set of features, $p(w_i, \{l_{j_p}\}|\{v_{dp}\})$. This is just the path probability of the full path involving the given word, letters, and observed features, divided by the sum of the path probabilities of all of the paths that could have generated the observed features:

$$p(w_i, \{l_{j_p}\}|\{v_{dp}\}) = \frac{p(P_{i,\{j_p\},\{v_{dp}\}})}{Z_{\{v_{dp}\}}} \ .$$

The denominator represents a quantity called the partition function. It stands for the sum over all $n_w \times 26^4$ path probabilities, where n_w is the number of words in the word list of the generative model. The above equation is nothing more than an application of Bayes formula, but in a situation where the alternative hypotheses are the alternative *combinations* of possible word and letter identities that could have produced the given evidence, or ensemble of features.

Let us now consider how we could calculate the posterior probability that the word responsible for a given path was word i, given that we observed the set of features $\{v_{dp}\}$. This will be the sum, over all paths that can generate these features starting from the word i, of the probabilities of these paths, divided by the sum over all of the paths that could have generated the observed features:

$$p(w_i|\{v_{dp}\}) = \frac{\sum_{j_1,j_2,j_3,j_4} p(w_i) \prod_p p(l_{j_p}|w_i) p(\{v_{d_p}\}|l_{j_p})}{Z_{\{v_{dp}\}}}$$

The summation in the numerator is the sum over the 26^4 possible combinations of the 26 possible letters, one in each of the four letter positions, and $Z_{\{v_{dp}\}}$ is the partition function as above.

[9]Note the distinction between $\{v_{dp}\}$, the full set of indices of the active feature value units across all dimensions and all positions, and $\{v_{d_p}\}$, the set of indices of the active feature values in position p.

It is useful at this point to introduce the conceptual and terminological distinction between the *joint* posterior probability of a *combination* of unobserved variables and the *marginal* posterior probability of a single unobserved variable. $p(w_i, \{l_{j_p}\}|\{v_{dp}\})$ is an example of a joint posterior probability, whereas $p(w_i|\{v_{dp}\})$ is an example of a marginal posterior probability. There are also marginal posterior probabilities associated with the alternative letters in each position.

It will simplify further analysis to note that $p(w_i)$ is a constant that can be pulled out of the summation in the numerator above, and that we can use the distributive law[10] to rewrite $\sum_{j_1,j_2,j_3,j_4} \prod_p x_{j_p}$ as $\prod_p \sum_j x_{j_p}$. Using these two facts, the above reduces to:[11]

$$p(w_i|\{v_{dp}\}) = \frac{p(w_i) \prod_p \sum_{j_p} p(l_{j_p}|w_i) p(\{v_{d_p}\}|l_{j_p})}{Z_{\{v_{dp}\}}}$$

The value we obtain for each word i corresponds to the posterior probability of the word given the observed features.

Now, let us turn to the problem of calculating the marginal posterior probability that the letter in some arbitrary letter position is letter j, given the full set of feature values $\{v_{dp}\}$ over the four positions. This probability is just the sum of probabilities of all of the paths that involve letter j in position p and the given feature values in all four positions, divided by the sum of the probabilities of all of the paths that could have generated the given features. The expression below represents this summation. We focus on position 1 to simplify the notation – analogous expressions can be written replacing the index 1 with the index of any of the other letter positions.

$$p(l_{j_1}|\{v_{dp}\}) =$$

$$\frac{\sum_i \sum_{\{j_2,j_3,j_4\}} p(w_i) p(l_{j_1}|w_i) p(\{v_{d_1}\}|l_{j_1}) \prod_{p\neq 1} p(l_{j_p}|w_i) p(\{v_{d_p}\}|l_{j_p})}{\sum_{j'_1} \sum_i \sum_{\{j_2,j_3,j_4\}} p(w_i) p(l_{j'_1}|w_i) p(\{v_{d_1}\}|l_{j'_1}) \prod_{p\neq 1} p(l_{j_p}|w_i) p(\{v_{d_p}\}|l_{j_p})}$$

The expression looks complex, but if we approach it slowly it should make sense. Starting with the numerator, we start with the notation for the summation over all of the $n_w \times n_l^3$ possible paths that could have generated

[10] This law states that for all a, b, c, d: $(a + b)(c + d) = ac + ad + bc + bd$.

[11] It is worth noting that the simplification here dramatically speeds calculation. Instead of computing 26^4 separate products of four quantities and then adding these all up, we compute the product of four sums of 26 quantities, producing a speed up of 17,000:1. It is also easier to implement the add-then-multiply simplification as a parallel computation in a neural network.

the given features and that involve letter j in position 1. The probability of each of these paths is then the product of the prior probability for the word involved in the specific path, $p(w_i)$, times a corresponding expression for each of the letters involved in the path.

Once again we can simplify. Looking first at the numerator, we can pull out the expression $p(\{v_{d_1}\}|l_{j_1})$ from the summation over words and letter combinations, since this expression is constant with respect to these. Likewise, we can pull $p(l_{j_1}|w_i)p(\{v_{d_1}\}|l_{j_1})$ out of the summation over letter combinations, since it too is constant in all of these combinations. We can then use the distributive law to replace $\sum_{\{j_2,j_3,j_4\}} \prod_{p\neq 1} p(l_{j_p}|w_i)p(\{v_{d_p}\}|l_{j_p})$ with $\prod_{p\neq 1} \sum_{j_p} p(l_{j_p}|w_i)p(\{v_{d_p}\}|l_{j_p})$. In the denominator, we have partitioned the sum of the full set of path probabilities into subsets, one for each set of paths involving a different letter in position 1. We can apply the simplifications just described for the numerator to each such term in the denominator to obtain:

$$p(l_{j_1}|\{v_{dp}\}) =$$

$$\frac{p(\{v_{d_1}\}|l_{j_1}) \sum_i p(w_i)p(l_{j_1}|w_i) \prod_{p\neq 1} \sum_{j_p} p(l_{j_p}|w_i)p(\{v_{d_p}\}|l_{j_p})}{\sum_{j'_1} p(\{v_{d_1}\}|l_{j'_1}) \sum_i p(w_i)p(l_{j'_1}|w_i) \prod_{p\neq 1} \sum_{j_p} p(l_{j_p}|w_i)p(\{v_{d_p}\}|l_{j_p})}$$

The leftmost factor in the numerator $p(\{v_{d_1}\}|l_{j_1})$ now corresponds to the standard Bayesian quantity $p(\{e\}|h)$, where $\{e\}$ is the bottom-up evidence for the ensemble of features in position 1 and h is the hypothesis that the letter in position 1 is letter j. Everything else in the numerator specifies what we will call $p(l_{j_1}|c)$, the probability of letter j in position 1, given the context c, where the context is the set of features in all of the other letter positions. Thus we could rewrite the numerator as $p(\{v_{d_1}\}|l_{j_1})p(l_{j_1}|c)$. The denominator consists of a sum over all of the letters of corresponding quantities, so we can rewrite the above to express the posterior letter probability:

$$p(l_{j_1}|\{v_{dp}\}) = \frac{p(\{v_{d_1}\}|l_{j_1})p(l_{j_1}|c)}{\sum_{j'_1} p(\{v_{d_1}\}|l_{j'_1})p(l_{j'_1}|c)} \tag{8}$$

This equation once again looks like Bayes' formula, but this time, we use the probability of the item given the context in place of the prior or base rate. This should make sense: We can think of what we are doing here as using the context to set the "prior" probabilities of different letters to context-specific values, combining these context specific prior probabilities with the contribution of the evidence, to calculate the total support for each of the possible alternative letters.

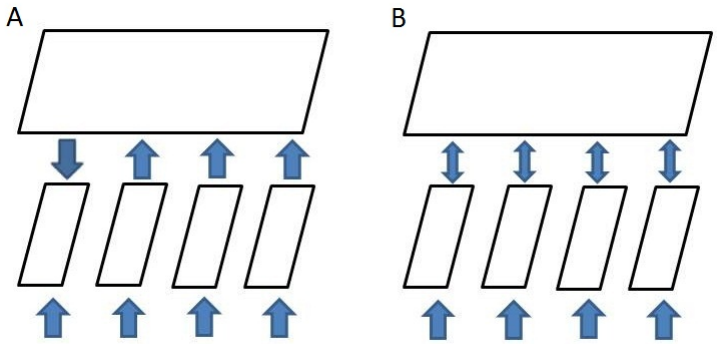

Figure 7: Schematic diagram of flow of computation for: (A) unidirectional procedure for calculating the posterior probabilities of letters in the first position and (B) bi-directional procedure for calculating the posterior probabilities of all four letters.

Calculating and Sampling Posterior Probabilities Given Observed Feature Arrays

The above can be thought of as a mathematical characterization of the true posterior joint and marginal probabilities of each word and of each letter in each position, conditional on observing some set of features $\{v_{dp}\}$, under the generative model. How might we calculate, or sample from, these quantities during perception?

We now describe two different ways to calculate the *marginal* posterior probabilities of words and letters – a *undirectional* method and an *interactive* method. After that we will describe how the updating procedure used in the multinomial IAM allows us to sample from the *joint* posterior distribution, and (as a byproduct) also the marginal posterior distribution over words and over letters in each position.

A unidirectional calculation method. Our first calculation method is completely non-interactive – information flows in a single direction between each pair of pools, as shown in Figure 7a.[12] Both Figure 7a and the text below apply to the particular case of calculating the posterior probability of possible letters in position 1, given the full set of features $\{v_{dp}\}$.

[12] In a sense the flow is still both bottom up and top-down, but there is no back-and-forth communication, which is the essence of interactive activation.

1. For each letter in each position, including position 1, we first calculate $p(\{v_{d_p}\}|l_{j_p})$. This corresponds to the arrows from each feature array to each letter array in Figure 7a.

2. For each word, we then calculate $p(w_i) \prod_{p \neq 1} \sum_{j_p} p(l_{j_p}|w_i) p(\{v_{d_p}\}|l_{j_p})$. This is the support for each word, given the feature information in all positions other than position 1, and we will thus call it $S_{i/1}$.[13] This corresponds to the three arrows from the position 2, 3, and 4 letter arrays to the word array in Figure 7a.

3. For each letter j in position 1, multiply each of the above word-specific terms by $p(l_{j_1}|w_i)$ and sum over words to obtain: $\sum_i p(l_{j_1}|w_i) S_{i/1}$. These quantities are the $p(l_1|c)$ terms we need, and the computation corresponds to the downward arrow in Figure 7a form the word level to the letters in position 1.

4. Finally, calculate $p(l_{j_1}|\{v_{dp}\})$ using the posterior letter probability equation (Eq. 8), taking $p(\{v_{d_1}\}|l_{j_1})$ from step 1 and the $p(l_{j_1}|c)$ from step 3.

The procedure can, of course, be applied to any letter position, just exchanging the roles of position 1 and any other position.

A drawback of the unidirectional method. The method just reviewed is basically consistent with the ideas of Norris & McQueen (2008) and Massaro (1989 and elsewhere). Both argue that when identifying the letter (or phoneme) in a particular string position, we must separately calculate context and stimulus support for each alternative, then combine these quantities only as the final step of our computation. The idea makes some sense when we think about using preceeding context to help recognize the next phoneme in a spoken word. We can imagine generating expectations based on the input received so far for the phoneme yet to come, applying this computation successively to each successive phoneme. However, experimental evidence (e.g., Rumelhart & McClelland, 1982) supports the view that perception of letters in every position of a briefly-presented word benefits from contextual influences from all other positions. The data indicates that the perception of each letter should benefit from the context provided by all of the other

[13] It could be worth noting that we have the option of normalizing the above quanties for each word by dividing them all by their sum. This step will not make any difference, since ratios of these quantities will be used later in calculating posterior probabilities at the letter level.

letters, and that these computations should be carried out in parallel, so that these influences can occur while the input is available for identification. (Subsequent context also affects phoneme perception, even though the context does not arrive until after the target phoneme; Ganong, 1980; Warren & Sherman, 1974, a key finding motivating the architecture of the TRACE model of speech perception, McClelland & Elman, 1986).

In general, to maximize the use of context, it seems desirable to calculate posterior probabilities for each letter using the context provided by all the other letters, and it could be useful to calculate posterior probabilities for words, based on all of the letters, as well. The original interactive activation model achieved something close to this, but not exactly – many of the complaints by Massaro and later by Norris et al., focused on the fact that the model did not get the posterior probabilities exactly right. Here we consider how the correct posterior probabilities can be calculated by an interactive procedure.

To calculate the posterior probabilities over words, we should of course include input from all four positions in the corresponding calculation of the S_i for each word. To calculate the context terms for a given position, position 1, say, we have to exclude its contribution to the word level to obtain the appropriate $S_{i/1}$ values. It would be possible to calculate S_i along with $S_{i/1}$, $S_{i/2}$, *etc.*, in parallel, but it seems redundant and computationally wasteful. Fortunately, there is a simple way to avoid the redundancy.

A parallel, interactive method. The approach we now consider is a specific instance of an approach proposed by Pearl (1982) – it is not the procedure we use in the multinomial IAM, but it is useful to understand both procedures, and the differences between them. Pearl's approach allows processing to occur in parallel for all four letter positions, relying on the bi-directional propagation of information, as shown in Figure 7b, minimizing the redundancy just noted. The key observation (specialized for our specific circumstances) is that the posterior probability for each word contains a product of terms, one from each letter position. The term from a given letter position p to each word unit i is $\sum_j p(l_{j_p}|w_i)p(\{v_{d_p}\}|l_{j_p})$. Suppose we call each of these quantities r_{i_p}. Then we can calculate the full bottom-up support for each word combining the r_{i_p} across all four positions, saving the r_{i_p} values so that we can divide them back out in calculating the $p(l_{j_p}|c)$ factors for each position. In more detail, here is the procedure:

1. For each letter in each position, calculate $p(\{v_{d_p}\}|l_{j_p})$.

2. For each word, then calculate $S_i = p(w_i) \prod_p r_{i_p}$ where r_{i_p} is as defined above, using the values calculated in step 1. S_i represents the total support for each word, given the feature information in all positions and the prior word probability, and can be used to calculate the posterior probability of each word by dividing through by the sum of all of the S_i.

3. For each letter position, we now calculate the appropriate top-down term by dividing S_i by r_{i_p} to obtain $S_{i/p}$. We then proceed to calculate, for each letter j, $p(l_{j_p}|c) = \sum_i p(l_{j_p}|w_i)S_{i/p}$ as in the unidirectional procedure.

4. For each position, we finally calculate $p(l_{j_p}|\{v_{dp}\})$, using the $p(\{v_{d_1}\}|l_{j_p})$ from step 1 and the $p(l_{j_p}|c)$ from step 3.

This procedure is a specific instance of the one proposed by Pearl (1982) for unidirectional causal graphical models (models in which causality propagates only in one direction, as in our generative model), subject to the constraint that each multinomial variable (i.e., each a set of mutually exclusive hypotheses) in the graph has at most one parent. The generative model underlying the multinomial IAM is an example of such a graph: In the generative model, the multinomial word variable has no parents; each of the four multinomial letter position variables depends only on the word variable; and each of the 14 binomial feature dimension variables in each letter position depends only on the letter variable for that position. The method allows for iterative, i.e., interactive updating; as new information arrives at any of the variables, it can be propagated through to update all of the other variables. There's an inherent sequentiality moving upward and then downward, but computation can occur in parallel in both directions. If feature information built up gradually over time, the process could easily be integrated, updating all of the variables as new evidence arises.

Pearl's method is an elegant and general method, and is now a long established part of the fabric of probabilistic computation. Interestingly, the idea did not come up in the development of the interactive activation model, even though the key idea of dividing back out one's contribution to a parent when receiving top-down input from the parent was proposed by Rumelhart (1977). Perhaps one reason why Rumelhart did not explore this idea in the original interactive activation model may be that Pearl's method requires each multinomial variable to keep separate track of its bottom-up and top-down values. What gets passed up in Pearl's algorithm is strictly feed-forward information; what gets passed down to a given multinomial variable

carefully excludes the information that came up through it, and must be kept distinct.[14] A feature Rumelhart found pleasing in the original interactive activation model was that the computation was entirely homogeneous. In an early talk on the model, he had on one of his transparencies: "activation is the only currency" transmitted between units in the network.

An important characteristic of Pearl's approach is that the posterior probabilities calculated for each variable are marginalized over all possible values of all of the other variables. To see what this means concretely, consider the set of features shown in Figure 8. The features in each position are consistent with two letters (H or F in the first position, E or O in the second position, and W or X in the third).[15] The features are also consistent with four words: FEW, FOX, HEX, and HOW.[16] Pearl's algorithm will allow us to calculate that these words and letters are the most likely. Ignoring differences in word frequency, the words would all be equally likely, and so would the letters. If we selected one word from those that are equally probable, and one letter from each position from those that are equally probable, we could end up with the conclusion that the word is FEW but that the letters are H, O, and X (letters that, together, don't even form a word).

The approach used in the multinomial IAM samples from the *full posterior* of the generative model. That is, it samples from the space of composite hypotheses consisting of one word and one letter in each position. For example, the joint hypothesis: WORD = FEW; LETTERS = F E W is one such hypothesis, while the hypothesis: WORD = FEW; LETTERS = H O X is another. The first of these joint hypotheses is far more likely than the second. The multinomial IAM assumes that perceivers select among alternative joint hypothesis weighted by their overall probability. We turn to the procedure used for doing this next.

[14]The importance of keeping these messages distinct becomes even more clear if the top-down signals need to be passed down more than one level, a possibility that arises in Pearl's general formulation.

[15]To the human eye, the features in position 1 seem consistent with the letter B as well as the letters F and H. However, in the Rumelhart & Siple font, B does not have a vertical on the left, so it is ruled out by the given features. Also, humans appear not to entertain the possibility of W in the third position, but again, the given features are equally consistent with W and X in the Rumelhart & Siple font.

[16]The word HEW is also consistent with one letter in each position, but it is very low in frequency and for the sake of the example we assume it is not known to the perceiver.

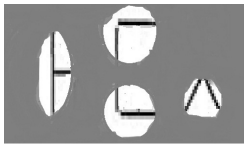

Figure 8: A few features from each position of a three-letter word. Based on the Rumelhart & Siple font, there are two consistent letters in each position, and four consistent words.

Sampling from the Joint Posterior in the Multinomial IAM

The final approach we will consider, which is the one used in the Multinomial IAM, goes by a number of names, including Gibbs sampling, and is closely related to the procedure used in the Boltzman Machine (Hinton & Sejnowski, 1983). We present the specific version of these ideas used in the multinomial IAM, which have been adapted and specialized for our case.

In the model, there are word, letter, and feature units as illustrated in Figure 6, and weights can be considered bi-directional, as in the figure, but their values are defined only in one direction. At the word level, each word unit has a bias term, corresponding to the log of its prior probability, $\log(p(w_i))$. The connection weight linking each word to each letter is set equal to $\log(p(l_{j_p}|w_i)$, and the weight linking each feature to each letter is set to $\log(p(f_{v_{dp}}|l_{j_p}))$.

We specify the input to the model by setting the values of the feature units of the MIAM to correspond to a specific input feature array. With all units at the letter and word levels initially off, we proceed as follows:

1. For each position, we calculate each letter's net input. Since the word units are all off, there will be no contribution from the word level, and each letter unit's net input will correspond to $\log p(\{v_{d_p}\}|l_{j_p})$.

2. Within each letter position we then select one unit to be on, according to the softmax function.

3. We then calculate the net input to each word unit, based on the active letter unit in each position.

4. We then select one word unit to be on, using the softmax function.

5. We then repeat steps 1-4 some number of times, noting that now, one unit at the word level is active on each iteration, providing top-

down input that affects the net input to each letter unit. We call this iterative process 'settling'.

6. Once we have iterated several times, we can stop, with one word unit active and one letter unit active in each position.

The state of activation in the network after these iterations is a sample from the joint posterior. That is, if we ran this procedure a very large number of times, and counted the number of times the pattern at the end of settling corresponded to each possible joint hypothesis (one word and one leter in each position), the proportion of times the network settled to each such pattern would correspond to the posterior probability of the corresponding joint hypothesis.

Running the above procedure hundreds of thousands of times would not be very efficient, but we would not propose that perception involves such a process. It makes more sense to imagine that each trial of a perceptual experiment involves a single instance of running the above procedure. Each such instance generates a single sample from the above process, capturing the probabilistic nature of performance in behavioral experiments. In a perceptual experiment where the task is to identify the letter in a specified position (as in most of the experiments modeled using the original IA model), we can simply imagine that the participant simply reads out the identity of the letter corresponding to the active unit it the appropriate position. Note that this procedure is a way to use a sample from the joint posterior as a sample from the marginal posterior for a particular multinomial variable (e.g. the letter in the specified position).

Relationship to Gibbs sampling and Boltzmann machines. The above procedure is related to the updating procedure proposed for Boltzmann machines by Hinton & Sejnowski (1983, 1986). One difference is that in the original Boltzmann machine, units are not organized into pools corresponding to multinominal random variables. Units are updated one at a time, selecting the next unit to update sequentially and at random; also each unit uses the logistic function. Our network is similar, but our units are organized into multinomial variables such that only one unit per multinomial variable is allowed to be active at one time. In both cases, after an initial burn in period (corresponding to what we called settling above), networks visit global states with probability proportional to $e^{G_s/T}$, where G_s is the goodness of the state and T corresponds to the temperature. The goodness

of a state is defined as:

$$G(s) = \sum_{i<j} a_i a_j w_{ij} + \sum_i a_i bias_i \ ,$$

where the summation runs over all pairs of units (with each pair of units counted only once) and w_{ij} corresponds to the bi-directional weight between the two units.[17] Additional terms can be included to represent external inputs to units, but these can be captured using weighted inputs from units whose activations are treated as clamped, as the set of input feature units are in our procedure.

For the multinomial IAM, the goodness of a state in which one word is active and one letter in each position is active, given a set of input feature values clamped onto the input units, is given by what we will call the MIAM goodness equation:

$$G(s|\{v_{dp}\}) = bias_i + \sum_p (w_{i,j_p} + \sum_d w_{j_p,v_{d_p}}) \tag{9}$$

Here i indexes the single word unit that is active at the word level, the four values of $\{j_p\}$ (one for each position p) index the active letter unit in position p, and the set of 4 times 14 values of $\{v_{d_p}\}$ represent the indices of the active feature-value units on each feature dimension in each feature position. The activations of the units involved are not expressed as variables because they are all equal to 1; no other terms occur in the goodness because all other units' activation values are 0.

As an exercise, the reader can check that the goodness of a state as represented by the MIAM goodness equation is equal to the log of the probability of the corresponding path under the generative model, by taking the log of the path probability equation (Eq. 7). Given that this is so, if we run our network at some temperature T, then the network will visit this state with probability proportional to $e^{log(p(P))/T}$, where $p(P)$ is the probability of the path corresponding to the state. The posterior probability of visiting this particular state given the particular feature values can be calculated by dividing through by the sum of the exponentiated and T-scaled goodnesses of all of the states that can be visited given the feature values:

$$p(S|\{v_{dp}\}/T) = \frac{e^{G_{s|\{v_{dp}\}}/T}}{\sum_{s'} e^{G_{s'|\{v_{dp}\}}/T}}$$

[17] Instead of goodness, Hinton & Sejnowski, 1986, included a minus sign and called the quantity energy, following Hopfield (1982), but the equation is otherwise the same.

For the case where T is equal to 1, we obtain:

$$p(S|\{v_{dp}\}) = \frac{e^{G_{s|\{v_{dp}\}}}}{\sum_{s'} e^{G_{s'|\{v_{dp}\}}}}$$

This is equal to the posterior probability of the state, given the evidence.

Hinton & Sejnowski focused on the task of finding the single best joint hypothesis using a process they call *simulated annealing*. In this process, one engages in a similar sequential update process to that described above, but with gradually reducing temperature. The procedure we have described operates at a fixed temperature. At lower temperatures, the preference for units with stronger net inputs is amplified, and as T goes to zero, the procedure will allocate all of the probability to the alternative with the largest net input. Gradually lowering the temperature corresponds to gradually increasing the relative probability of visiting the alternatives with the largest posterior probability. It may be worth noting that a gradual change in the clarity of evidence can have a similar effect as a gradual change in temperature, or that running the procedure when the evidence is very weak can be similar to running the procedure at very high temperature. Thus, perception with very weak stimuli may correspond approximately to running the model at very high temperature, and gradual buildup of information over time may correspond to simulated annealing. These ideas may be worth developing further in extensions of the multinomial IAM.

Why does the Multinomial IAM sample correctly from the posterior? So far we have stated without proof that 'after a burn in period' and at fixed temperature, states are sampled in proportion to $e^{G(s)/T}$. How do we know that this is true? For particular cases, we could demonstrate the validity of this result via stochastic simulation, and we did so for one specific case in Mirman *et al.*. The fact that it is true for all cases follows from the fact that the sampling procedure we are using is an instance of a *Gibbs sampling procedure*. The Gibbs sampler (named after the physicist J. W. Gibbs) is widely used to sample from the posterior probability distributions.

A Gibbs sampling procedure is a procedure that obtains samples from the joint posterior of a set of random variables by successively updating sampled values of individual probabilistic variables conditional on the values of other variables. Concretely in our case, we are updating the multinomial word variable based on the letter variables and eacn letter variable based on the word variable and the appropriate position specific feature variables. We can see our procedure as sampling from the conditional distribution of the

word variable based on the values of the feature and letter variables on each update at the word level, and as sampling from the conditional distribution of each letter variable, based on the values of the word and feature variables, on each update at the letter level. After burn in, the overall state after each update is a sample from the joint distribution over all of the variables. The statement that such states are samples from the joint posterior means that the probability of visiting each state (at equilibrium, i. e., after a burn-in period) is equal to the posterior probability of the state.

Two properties of our sampling procedure are necessary to ensure that it accurately samples from the posterior (Hastings, 1970). First, the process must be *ergodic* – it must be possible to get from any state to any other state in a finite number of steps. Taking the feature unit's values to be clamped, we are concerned only with states corresponding to a joint specification of a word and four possible letters. The process is ergodic if it is possible to get from any state of the word and letter units to any other state of these units. This property holds in our case, because all of the probabilities encoded in the weights are non-zero, making it possible (a) to visit any possible state of the letter units given an active word unit and a set of active feature values, and (b) to then visit any possible state of the word units given a set of active letter values. In our case, then, it is possible in principle to get from any state to any other state in one update cycle, consisting of one letter update and one word update. So our model is ergodic.[18]

The second critical property is that the updating process exhibits *detailed balance*. A stochastic updating process is said to have detailed balance with respect to a particular probability distribution $\{\pi\} = \{\ldots, \pi_i, \ldots, \pi_j \ldots\})$ over possible states if the probability of being in state i and transitioning to state j is equal to the probability of being in state j and transitioning to state i:

$$\pi_i p(i \rightarrow j) = \pi_j p(j \rightarrow i) \ ,$$

or equivalently,

$$\frac{p(j \rightarrow i)}{p(i \rightarrow j)} = \frac{\pi_i}{\pi_j} \ ,$$

If a stochastic updating process has this property, it will converge to the equilibrium distribution $\{\pi\}$ in which the probabilities of states i and j are π_i and π_j respectively; ergodicity ensures that we can get to the equilibrium

[18]It should be noted that some of the transition probabilities can be very small, and thus many of the transitions are highly unlikely. As we shall see below, we will not be relying on moving widely accross the state space during processing of a sing item.

distribution from any starting state.[19]

Intuitively, the dynamic balance condition can be seen as a way of expressing what it means for a probability distribution to be at equilibrium, or stationary. Referring to the first version of the equation, we can read it as saying that at equilibrium, the probability of being in state i and then transitioning to state j should be equal to the probability of being in state j and transitioning to state i. If this is true for all pairs of states, and if we can get from any state to any other state, then the distribution over states will stay constant as we continue to update. It is important to note that it is not the states themselves but the *distribution over states* that is stationary. On each update, the state may change, but the probability distribution over states, conceived of as the proportion of times an infinitely large ensemble of instances of the process is in each possible state, can still remain stationary. This is so because in the ensemble, the dynamic balance condition stipulates that the probability of being in state i and transitioning to state j is equal to the probability of being in j and transitioning to i.

We have just seen that if we are already at equilibrium (i.e., if the ratios of probabilities of particular states are equal to the ratios of the corresponding transition probabilities) we will stay there. But what if, at a certain time, the distribution of states is not yet at equilibrium? In that case, if the transition probability ratios are equal to the equilibrium probability ratios, the transitions will tend to move the distribution toward the stationary distribution. We will not prove this statement but we will consider an example a bit later showing how movement toward the correct stationary distribution does occur.

To show that our updating procedure will sample from the posterior distribution of the multinomial IAM, we must show that its state transitions are balanced with respect to the posterior probabilities of the paths associated with these states, i.e., that the transition probabilities between states i and j are in balance with the posterior path probabilities. To do so, it is easier to work with the second version of the statement of the detailed balance condition. Working with this version, we would like to show that the ratio of the transition probabilities between any two states is equal to the ratio of the posterior probabilities of the generative paths corresponding to these states. Designating these states and the probabilities of the corresponding

[19] Being ergotic is an in-principle matter, and some of the transition probabilities can be very small, but the starting state we start from – all units off except the clamped feature units – makes for easy access to all of the states that are plausible given the input.

paths with the subscripts i and j, this corresponds to the expression

$$\frac{p(S_i \rightarrow S_j)}{p(S_j \rightarrow S_i)} = \frac{\pi_i}{\pi_j} \ .$$

For concreteness, let's consider a specific case. Suppose that the input features are the correct values of the features of the word TIME, and that the correct word is active at the word level, and the correct letter is active in positions 2, 3, and 4. Let state S_I be the state in which, in addition to the above, the letter I is active in the first position and let state S_T be the state in which, in addition to the above, the letter T is active in the first position, and let π_I and π_T represent the probabilities of the corresponding paths of the generative model. Using these indices, the above would then correspond to:

$$\frac{p(S_T \rightarrow S_I)}{p(S_I \rightarrow S_T)} = \frac{\pi_I}{\pi_T} \ .$$

Based on the visual similarity between I and T in the Rumelhart & Siple font, the paths associated with these states should be among the most probable, although state I should be less probable that state T. Now, suppose we are in state I and we are about to update the state of the first-position letter variable. We calculate the net input to each letter unit based on the active features and the active letters, and we then select the letter to activate according to the softmax function. The probability of transitioning to state T, i.e. of selecting T as the next letter, is $\frac{e^{net_T}}{\sum_j e^{net_j}}$, where the net input to the unit for letter T is

$$log(p(l_{T_1}|w_{TIME}) + \sum_d \log(p(f_{v_{d_1}}|l_{T_1}))$$

so that e^{net_T} is $p(l_{T_1}|w_{TIME}) \prod_d p(f_{v_{d_1}}|l_{T_1})$. Similarly, suppose we are in state T and we are about to update the state of the first-position letter variable. We proceed as before, and find that the probability of transitioning to state I is $\frac{e^{net_I}}{\sum_j e^{net_j}}$, where the net input to the unit for letter I is

$$log(p(l_{I_1}|w_{TIME}) + \sum_d \log(p(f_{v_{d_1}}|l_{I_1}))$$

and e^{net_I} is $p(l_{I_1}|w_{TIME}) \prod_d p(f_{v_{d_1}}|l_{I_1})$. The ratio of these two transition probabilities, $\frac{p(S_I \rightarrow S_T)}{p(S_T \rightarrow S_I)}$ is then

$$\frac{p(l_{T_1}|w_{TIME}) \prod_d p(f_{v_{d_1}}|l_{T_1})}{p(l_{I_1}|w_{TIME}) \prod_d p(f_{v_{d_1}}|l_{I_1})}$$

They have the same denominator, which cancels out. This ratio is the same as the ratio of the posterior probabilities of each of the two paths, since the path probabilities share all of the other factors in common as well as the same denominator, and again everything else cancels out.

It would be tedious to repeat the above analysis for all possible pairs of states that might arise in the course of our sampling process. Luckily, there was noting special about the particular case we just considered. The same argument can be applied for any pair of states differing only by the letter that is active in one of the four letter positions, given any set of clamped features. Furthermore, an analogous argument can be applied for any two states differing only by the word that is active. Since all the transitions are from one letter in a given position to another letter, or from one word to another word, this covers all of the transitions.

This completes the proof that the multinomial IAM exhibits detailed balance, and we previously saw that it was ergodic. It follows, then, that the multinomial IAM samples states with probabilities corresponding to the posterior probabilities of the corresponding paths through the generative model.

In the context of the example we were working with above, we can now observe that the distribution of states tends to move toward the correct equilibrium distribution, at least in a simple specific case. Consider, for concreteness, an ensemble of 1,000 separate instances of our network, and let an arbitrary fraction be in state I and the rest be in state T just before we update the multinimial variable for the first letter position in each of these 1,000 instances. As one possibility, all of the networks could be in the I state. Now, we note that our update procedure is unaffected by the previous active letter in the first letter position (it depends only on the state of the feature and word units – the other multinomial variables in the system). Relying on the same reasoning we worked through above, it should be clear that the update in each network will put the system in state T with a probability proportional to $\pi_T = p(P_T)$, and in state I with a probability proportional to $\pi_I = p(P_I)$, and thus the ratio of the proportion of networks in states I and T will tend toward $\frac{p(P_T)}{p(P_I)}$ after the update. Thus, in this case, we can move from a distribution far from equilibrium to something much closer to it in just a single update step. We have not considered what would happen in a full range of cases, but perhaps this example helps support the intuition that, in general, the distribution of states will tend to move toward the equilibrium distribution, if the transition probability ratios are in balance with the posterior path probabilities.

A few practicalities. It is important to be aware of two things when using Gibbs sampling and related procedures. First, it takes some time for the settling process to reach the stationary distribution. It is difficult to know how many iterations of settling to allow before taking the state of the network as a valid sample from the stationary distribution, and testing for stationarity is not easy. Second, while in principle it is possible to transition from any state to any other state, in practice adjacent states tend to be correlated, and it may take a long time to make a transition between quite different, but equally good possibilities. For example, for the display in Figure 8, the time needed to transition from the interpretation WORD = FEW; LETTERS = F E W to the interpretation WORD = HEX; LETTERS = H E X may be quite long. It may, in fact, be quicker to get a set of decent samples by restarting from a blank starting place several times. This is essentially how we proceeded to sample from the multinomial IAM. This is appropriate for our purposes, given that we think of each trial in a perceptual experiment as corresponding to a single case of settling to a perceptual interpretation, corresponding to a sample from the posterior.

Learning an Interactive Model of Perception

In the above we have shown how an explicit generative model can be embedded in a perceptual system, so that it can sample from the generative model's posterior distribution. For this case, we have had the advantage of working in a domain – the domain of printed words and letters – where the relevant underlying units (words and letters) and the generative process (selecting a word, then the letters based on the spelling of the word) are assumed to be known in advance. Although the generative model is of course a fiction, it could actually be used to generate displays in a psychology experiment. Real scenes that we are called upon to perceive are of course far more complex. There may be several objects in a display at the same time – so rather than a single underlying cause, there can be several. The underlying causes my be partially, but perhaps not completely, independent. The objects may take various poses and scales, be subject to various occluders and misleading lighting effects, etc. The objects themselves might not be fully characterized by mutually exclusive discrete identities, as words and letters are. To handle such cases, one can well imagine that no explicit generative model could ever do full justice to the actual probabilities involved.

A solution to this problem may involve using a network in which the units and connection weights are not pre-assigned, but learned. The network could still be viewed as instantiating a generative model, but without the

stipulation of the correct set of units or connections. This is the approach taken in the deep belief networks introduced by Hinton & Salakhutdinov (2006).

Conclusion

This tutorial has covered a lot of ideas related to Bayesian inference, generative models, and neural networks. The primary goal was to provide the background necessary for readers unfamiliar with these ideas to understand the proposition that the multinomial interactive activation model samples from the posterior distribution of the paths through its generative model, and to have an intuitive appreciation for why this proposition is true. I hope that the tutorial fulfills this goal, and I also hope that it will be of broader use. The concepts are used extensively in probabilistic modeling and in psychology and cognitive science, and in recent neural network modeling appraches such as deep belief networlks. As such, whatever the reader has learned about them may prove useful in gaining a foothold in this literature.