Alderete, John, Paul Tupper, Stefan A. Frisch. 2012. Phonotactic learning without *a priori* constraints: Arabic root co-occurrence restrictions revisited. In Proceedings of the 48th meeting of the Chicago Linguistics Society.

Math supplement

Both MaxEnt and c-net approaches to modeling acceptability judgements construct a function F with parameters p. The function takes a word x as input and then outputs a number indicating the acceptability of the word.

$a = F(x,p) = F(x)$.

There are two distinct aspects to the approach. One is the architecture of the network; the other is how the parameters p of the network are set by the training regime. Here we address the first aspect, and show that our c-net approach has greater expressiveness than Hayes and Wilson's MaxEnt architecture. We show that for any choice of constraints c_i and weights w_i used in the MaxEnt grammar, we can choose the number of hidden nodes and the weights on the connections so that the c-net agrees on acceptability judgements arbitrarily closely. We also give an example of a constraint that can be expressed in our network architecture but not in Hayes and Wilson's architecture.

The MaxEnt architecture developed in (Hayes and Wilson, 2008) can be characterized as follows:

$F(x)= \exp(- \sum_i w_i c_i(x))$

where c_i are constraint functions and w_i are the weights. Each c_i returns either 0, 1, 2, ... for each word x, corresponding to the number of violations of a given constraint. Constraints must be given in terms of natural classes and must be translation invariant, i.e. constraints are always constraints in adjacent segments, but it does not matter where in the word adjacent segments are located. Both c_i and w_i are determined by the learning algorithm.

Our architecture is:

$F(x) = \sigma(\delta - \sum_i w_i \ \sigma(b_i + (V x)_i))$

Here δ, w_i and b_i are scalars. V is an h x n matrix, where h indicates the number of hidden units and n is the length of the input x; b_1 and b_2 are vectors; σ is a sigmoid function that gives values between 0 and 1. (In our actual simulations we used a sigmoid running between -1 and 1, but we use this equivalent alternative here to simplify discussion.)

First, considering that we are primarily interested only in relative judgements of acceptability we omit the outer function evaluations of both values for F, since exp and σ are both monotonic functions. Let us call this G.

For Hayes and Wilson we have

$G(x)= -\sum_i w_i c_i(x)$.

However, we will interpret each of their constraints as one constraint for each segment in the word, in which case $c_i(x)$ is either 0 or 1.

For our model we have

$G(x) = -\sum_i w_i \sigma(b_i + (V x)_i)$.

We claim that a suitable choice of w_i, b_i and V can match our model arbitrarily close to their model. Choosing w_i to be the same as their w_i is straightforward. So it only remains to show that for each i.

$c_i(x) = \sigma(b_i + (V x)_i)$

for some b_i and V. Dropping the subscripts, for each constraint c of Hayes and Wilson, we need to show that there is a scalar b and a vector v such that

$c(x) = \sigma (b+ \sum_j v(j) x(j))$,

where v(j) denotes the jth entry of the vector v.

The expression on the right gives the output of a perceptron with a smooth activation function σ. Let us first imagine that sigma is the step function and then we will consider our smoother case. This means that $\sigma (b+ \sum_j v(j) x(j))=1$ if $b+ \sum_j v(j) x(j)>0$ and it is equal to 0 otherwise. As is well known, not all Boolean functions can be computed by a perceptron, exclusive OR being a prominent example (McLeod et al., 1998). However, they are capable of matching the limited set of Boolean functions of feature values used by Hayes and Wilson in their MaxEnt grammar, as we will show.

Hayes and Wilson use two types of constraints. The first prohibits a collection of feature values over one or more segments. For example, a constraint they propose for English onsets is

*[+ant, +strid][-ant]

which prohibits the cluster /sr/, among others. We will consider this constraint as applying to the first two segments of a form.

This constraint can be captured by a perceptron as follows. Let j_1,j_2, j_3 be the indices of the input nodes corresponding to [ant] for the first segment, [strid] for the first segment, and [ant] for the second segment, respectively. We want to find a scalar b and a vector v so that if $x(j_1)=1$, $x(j_2)=1$, and $x(j_3)=-1$, then $b+ \sum_j v(j) x(j)) >0$, but otherwise $b+\sum_j v(j) x(j))<0$. This is achieved by letting $v(j_1)=1$, $v(j_2)=1$, $v(j_3)=-1$, all other v(j)=0, and b=-2.5.

A similar idea works for any constraint of this type. Suppose a constraint is specified on J nodes with indices j_1 through j_J by saying that there is a violation if for all k=1,2,…,J, $x(j_k)=e_k$, where each e_k is -1 or 1. To capture this with a perceptron, let $v(j_k)=e_k$ for k=1,…,J and v(j)=0 otherwise. Let b= -J+1/2. If all J of the relevant nodes have their prohibited values then $\sum_j v(j) x(j) = J$, and $b+\sum_j v(j) x(j))>0$, causing the perceptron to return 1. However, if not all of the relevant nodes are active the perceptron will return 0 as required.

The other form of constraint considered by Hayes and Wilson is implicational. An example is

*[+lab][^+son, +cor],

which in their notation means that if a consonant is [+lab] it may only be followed by a consonant with [+son, +cor]. To capture this constraint with a perceptron, as before we let j_1 correspond to [lab] on the first consonant, j_2 correspond to [+son] on the second

consonant, and j_3 correspond to [+cor] on the second consonant. We then let $v(j_1)=1$, $v(j_2)=-1/2$, $v(j_3)=-1/2$, and $b=1/4$. Suppose $x(j_1)=1$. Then the only way to prevent the perceptron from being on $(b+\sum_j v(j)\, x(j))>0)$ is if both $x(j_2)$ and $x(j_3)$ are both 1.

More generally, suppose we have a constraint that $x(j_i)=e_i$ for $i=1,\ldots,K$ implies that $x(k_m)=f_m$ for $m=1,\ldots,J$. We assume that the indices j_i and k_m are distinct, as they are for the constraints used by Hayes and Wilson. We let $v(j_i)=e_i$ for all i, $v(k_m)=f_m/K$ for all m and all other entries of v be zero. Then we let $b= J-1+1/K$. It is straightforward to check that $b+\sum_j v(j)\, x(j))>0$ only if $x(j_i)=e_i$ for all I, but for some m, $x(k_m)$ is not equal to f_m.

The arguments above were all for the case where sigma is the step function. However, a perceptron with the step function can be approximated arbitrarily well with our sigmoid function if we multiply v and b by a sufficiently large positive constant.

To show that our architecture has strictly greater expressiveness than Hayes and Wilson's, consider a hypothetical constraint that requires at least one of the first two segments of a form to have the feature [lab]. This fits neither of the two constraint formats used in MaxEnt. To describe it in our architecture, let x(1), x(2) specify the feature [lab] for the first and second segment respectively. Then consider the constraint $c(x) = \sigma(-x(1) -x(2) +0.5)$. For the discontinuous σ, c(x) is 1 if both x(1) and x(2) are 0 and zero otherwise. For the continuous σ the constraint is approximately obtained by letting $c(x) = \sigma(K(-x(1) -x(2) +0.5))$ for some large constant K.