32_spreading_activation

May 13, 2021

Open in Colab

[1]:
```
import matplotlib as mpl
mpl.rcParams['figure.dpi'] = 400
import matplotlib.pyplot as plt
plt.style.use('seaborn')
import seaborn as sns

import numpy as np
```

[2]:
```
def base_activation(pres_times, moments):
    base_act = np.zeros(len(moments))
    for idx in range(len(moments)):
        past_pres_times = pres_times[pres_times<moments[idx]]
        base_act[idx] = \
            np.sum(1/np.sqrt(moments[idx] - past_pres_times))
    non_zero_activations = np.not_equal(base_act, 0)
    base_act[non_zero_activations] = \
        np.log(base_act[np.not_equal(base_act, 0)])

    return base_act
```

[3]:
```
pres_times = np.linspace(0, 5000, 5)/1000
moments = np.arange(10000)/1000
base_act = base_activation(pres_times, moments)
```

0.1 The attentional weighting equation

In addition to base activation, a chunk's activation depends on the context in which it is needed. What counts as ''context" within the ACT-R cognitive architecture?

- context for cognitive processes is the information that is instantaneously available to the procedural module: all the buffers and the chunks that reside in them (basically, working memory, or focus of attention)
  - this is sometimes called ''the general context"; besides the general context, there is also a specific context, relevant for partial matching
    * partial matching is available in `pyactr`, but we won't cover it at all

    - see **Lebiere, Christian. 1999. The dynamics of cognition: An act-r model of cognitive arithmetic. *Kognitionswissenschaft* 8:5–19** for a detailed discussion of partial matching and an example of how it can be applied to cognition and learning

We know that chunks consist of slot-value pairs. To capture the role of context, ACT-R assumes that any chunk V that appears as the value of some slot in a buffer spreads activation to

- chunks in declarative memory that have V as one of their values
- chunks in declarative memory that are content-identical to V, i.e., they consist of the same set of slot-value pairs as V

This context-driven boost in activation for chunks in declarative memory is known as *spreading activation*.

An example will help shed more light on the workings of spreading activation.

Suppose that only one buffer carries a chunk, say, the imaginal buffer.

- the chunk in the imaginal buffer is the representation of the word *car*
- we assume that the chunk has four slots: FORM, MEANING, CATEGORY and NUMBER
- each of these slots, in turn, has a chunk as its value:
  - the form
  - the interpretation
  - the syntactic category
  - the morphological number
- each of these values comes with a weight, as shown below.

$$\text{WORD}\begin{bmatrix} \text{FORM:} & \text{car}\left(W_1\right) \\ \text{MEANING:} & [\![\text{car}]\!]\left(W_2\right) \\ \text{CATEGORY:} & \text{N}\left(W_3\right) \\ \text{NUMBER:} & \text{sg}\left(W_4\right) \end{bmatrix}$$

- the form *car* has weight W_1
- the meaning $[\![car]\!]$ has weight W_2
- the syntactic category N has weight W_3
- the number specification sg has weight W_4

Any chunk i in declarative memory that shares values with the imaginal chunk (or that consists of the same slot-value pairs, but we ignore this case for expositional simplicity) receives spreading activation proportional to ($\propto$) the weights W_j (for $j \in \{1, 2, 3, 4\}$) of the values that chunk i has in common with the imaginal chunk.

- chunk i receives an activation boost just by virtue of containing any of the four values in the context chunk, i.e.:
  - the form *car* (W_1)
  - $[\![car]\!]$ (W_2)
  - N (W_3)
  - sg (W_4)
- intuitively, sharing a value with a context chunk (like the *car* chunk in the imaginal buffer) 'connects' chunk i in declarative memory to the context chunk
- activation can now spread / flow along this connection, and this spreading activation is proportional to the weight W_j (in symbols: $\propto W_j$) of the connecting value

- note that these values are themselves chunks, but we will continue to refer to them as values and explicitly call 'chunk' only:
  - the chunk in declarative memory that receives spreading activation
  - the context chunk that is the source of the spreading activation

We keep insisting that spreading activation is proportional to weight W_j ($\propto W_j$), but not identical to it.

- this is because chunk i in declarative memory does not simply add W_j to its activation.

Every weight W_j, or *source activation*, is scaled by an *associative strength* S_{ji}.

- it is the product $W_j \cdot S_{ji}$ that gets added to the activation of chunk i

Intuitively, we can think of this associative strength as the strength (or the resistance, if you will) of the connection between:

- chunk i: the activation-receiving chunk in declarative memory
- the context chunk that is the source of spreading activation

Every value shared between chunk i and the context chunk 'creates' a connection along which activation W_j can spread / flow.

- this connection has a strength / resistance S_{ji} specific to the value j that 'created' the connection and to the activation-receiving chunk i

In our example, we have four weights / source activations:

- W_1, W_2, W_3, W_4

We also have four corresponding associative strengths:

- $S_{1i}, S_{2i}, S_{3i}, S_{4i}$

Associative, or 'connection', strength is basically a measure of how predictive any specific value in a context chunk is of chunk i. This idea of 'predictive' association will make more sense in a moment when we introduce the concept of fan.

In our example:

- the form *car* has weight W_1
  - but we don't simply add that to the base activation B_i of our chunk i in declarative memory
- we scale W_1 by the associative strength S_{1i}, which is the strength of the connection created by
  - the value *car*, which resides in the FORM slot of the imaginal buffer
  - chunk i, which resides in declarative memory and which has the same value *car* in one of its slots
- thus, the activation boost spreading to chunk i in declarative memory from the value *car* in the imaginal buffer is $W_1 S_{1i}$

The resulting total activation A_i for any chunk i in declarative memory will therefore be the sum of:

- its base activation B_i, which reflects its past history of usage

- whatever spreading activation it gets from the cognitive context, which in our example is restricted to the imaginal buffer

When activation spreads along all four values of the imaginal chunk (that is, chunk i in declarative has all these four values in its slots), we have:

- $A_i = B_i + W_1 S_{1i} + W_2 S_{2i} + W_3 S_{3i} + W_4 S_{4i}$

0.2 How do we set the weights / source activations and associative strengths?

How are we to set the weights and the associative strengths? One answer that immediately comes to mind is: empirically; we set some low-information / vague priors over the weights and the associative strengths and infer them from suitable experimental data.

- this is in fact what we will do (see Chapter 8 of the book)

For now, we will simply discuss some reasonable default values.

Every chunk in a buffer that spreads activation is assumed to have a total source activation W that gets evenly distributed among the values that reside in the slots of that chunk.

- W is by default set to 1
- in our example, this would mean that $W_1 = W_2 = W_3 = W_4 = \frac{1}{4}$

Default value for source activation (summary): - $W_j = \frac{W}{n}$, where - j goes from 1 to the number of slots n that carry a value - W is by default set to 1.

Let's turn now to the associative strengths S_{ji}, where

- i is the chunk in declarative memory that receives spreading activation
- j, which varies from 1 to n, is a value in the cognitive context, i.e., in the buffer that spreads activation
  - we assume that this buffer has n slots that carry a value

For these associative strengths S_{ji}, we want to capture the intuition that:

- the association should be 0 if j does not associate with i (it is not predictive of i in any way)
- it should be high if j uniquely associates with the chunk i (because j is then highly predictive of i), which would happen if there is no other chunk in declarative memory that is associated with j
- the association strength should decrease as more and more chunks in declarative memory are associated with j, since j becomes less predictive of any of these chunks |

This intuition is captured by the following formula:

- $S_{ji} \approx \log \frac{prob(i|j)}{prob(i)}$

That is, the strength of association between

- value j in a context buffer that spreads activation, and
- chunk i in declarative memory that receives activation

is approximately the log probability of needing chunk i from memory conditional on the fact that value j is present in the buffer.

- this probability is 'normalized' by the probability that chunk i is unconditionally needed

Formally, this is the pointwise mutual information (**pmi**) between

- the event that chunk i is needed / requested from declarative memory, and
- the event that j is a value in the activation-spreading context buffer, i.e., a chunk in one of the slots of that context buffer

Pointwise mutual information between two events i, j:

- $\mathbf{pmi}(i,j) = \log \frac{prob(i,j)}{prob(i)prob(j)} = \log \frac{prob(i|j)}{prob(i)} = \log \frac{prob(j|i)}{prob(j)}$
- **pmi** is a symmetric measure of association between single events (not an expectation, like mutual information)
- it can have both negative and positive values, and it is 0 if the events are independent

Understanding association strengths S_{ji} in terms of the **pmi** between the declarative memory chunk i and the value j in the cognitive context makes intuitive sense:

- strength of association is a measure of how predictive the contextual value j is of the need to retrieve chunk i from memory

ACT-R has developed a way to estimate the values in the equation above.

First, for cases in which i and j are not associated in any way, i.e., they are independent, the joint probability $prob(i,j)$ is the product of the marginals $prob(i)prob(j)$, so:

- $S_{ji} = \mathbf{pmi}(i,j) = \log \frac{prob(i,j)}{prob(i)prob(j)} = \log \frac{prob(i)prob(j)}{prob(i)prob(j)} = \log 1 = 0$

Another way to think about this is that, if i and j are independent, the contextual value j is not predictive at all of the need to retrieve chunk i from declarative memory, so:

- $prob(i|j) = \frac{prob(i,j)}{prob(j)} = \frac{prob(i)prob(j)}{prob(j)} = prob(i)$

Therefore, the contextual value j does not boost the activation of chunk i in any way.

To ensure that no activation spreads, we zero out the 'connection' strength:

- $S_{ji} = \log \frac{prob(i|j)}{prob(i)} = \log \frac{prob(i)}{prob(i)} = \log 1 = 0$

If j and i are not independent, i.e., there is an association between them, so j has some predictive value with regards to the need-probability of chunk i, the common estimate for $prob(i|j)$ is as follows:

- $prob(i|j) = \frac{1}{\text{fan}_j}$, where:
  - fan_j is the number of chunks associated with j in declarative memory, i.e.,
  - fan_j is the number of chunks that have j as a value in one of their slots

The intuition behind this common estimate is that a value j in the cognitive context is *equally predictive* of any chunk in declarative memory it is associated with.

Basically, in the past, whenever we had value j in the cognitive context, we were equally likely to need to retrieve any of the declarative memory chunks associated with j.

This kind of assumption is unrealistic in a naturalistic, 'ecologically valid' setting, but it is probably reasonable in the context of a counterbalanced experiment.

A common estimate for $prob(i)$ is:

- $prob(i) = \frac{1}{|dm|}$, where:
  - $|dm|$ is the size of declarative memory (dm): the number of chunks present in dm

Again, this is extremely unrealistic since it assumes that all the chunks in declarative memory have the same history of past usage (or no history of past usage), so they have the same probability of being needed/retrieved.

This estimate makes sense as a flat uniform prior used for convenience, perhaps in an experimental setting where frequentist and Bayesian posterior estimates of need probabilities for experimental items are intended to be identical.

With these two assumptions in place, associative strength S_{ji} can be estimated as follows:

- $S_{ji} = \log \frac{|dm|}{\text{fan}_j} = \log |dm| - \log \text{fan}_j$.
- the dependency on a specific declarative memory chunk i disappears because of the (unrealistic) uniformity assumptions built into the $prob(i|j)$ and $prob(i)$ estimates

It is hard to estimate the size of declarative memory, so the minuend $\log |dm|$ is often treated as a *free (hyper)parameter S*:

- with the requirement that S should be larger than all $\log \text{fan}_j$ for any value j

If this was not so, association could be negative in some cases, i.e., in some cases, association strength would yield negative spreading activation, decreasing (inhibiting) base activation for some items rather than simply failing to boost it.

In sum, the final form for associative strength that is commonly used in ACT-R modeling is as follows:

Associative strength equation between a value j and a chunk i:

$$S_{ji} = \begin{cases} S - \log \text{fan}_j & \text{if } i \text{ and } j \text{ are associated, i.e., } i = j \text{ or } j \text{ is a value of } i \\ 0 & \text{otherwise} \end{cases} \backslash$$

- where S is the maximum associative strength, a free (hyper)parameter

Putting all this together with base activation, we arrive at the activation equation below. This shows how spreading activation from just one buffer affects the total activation of elements in declarative memory.

Activation equation (simplified to one buffer): $\boxed{A_i = B_i + \sum_{j=1}^{m} W_j S_{ji}}$

- for any chunk i in declarative memory and all values j of the chunk in the buffer

This equation has three major components:

- base-level learning equation: $B_i = \log \left(\sum_{k=1}^{n} t_k^{-d} \right) = \log \left(\sum_{k=1}^{n} \frac{1}{\sqrt{t_k}} \right)$ (since usually $d = 0.5$), where t_k is the time since the k-th practice/presentation of chunk i
- attentional weight equation: see above
- associative strength equation: see above

Extending to more than one buffer is easy: we just sum up the spreading activation from all the buffers, as shown below.

Activation equation (generalized to all buffers): $\boxed{A_i = B_i + \sum_{k=1}^{n} \sum_{j=1}^{m_k} W_{kj} S_{ji}}$

- for any chunk i in declarative memory, all buffers k and all values j of the chunk present in buffer k, where the chunk in buffer k has m_k slots

This equation has the same three major components as the simpler one above. Differences:

- We sum over all buffers k, from 1 to n buffers in the cognitive context
- The weights/source activations W_{kj} are indexed with both the value j that is their source as well as the buffer k where value j is located

To understand this a little better, think of the typical scenario in which spreading activation, i.e., source activations scaled by associative strengths, is used

- the values j we typically consider are values stored in the slots of the imaginal or the goal chunk
- these buffers drive the cognitive process, so they provide a crucial part of the cognitive context in which we might want to retrieve items from memory
- when we have a goal or an imaginal chunk, we associatively bring to salience, i.e., spread activation to, chunks in declarative memory that are associated to the imaginal or goal chunk since they might be needed
- we operationalize this 'association' between a chunk in the cognitive context and a chunk i in declarative memory in terms of the chunks being content identical (consisting of the same same set of slot-value pairs) or sharing some value j in some of their slots
- this essentially results in increasing the activation of those declarative memory chunks that are related to the current cognitive context
  - i.e., ultimately, that are potentially relevant to the current stage of the cognitive process we are involved in
- the associative strength S_{ji} is really the probability that chunk i is relevant given a cognitive context in which we attend to the value j

One intuitive way to think about the activation of chunks in declarative memory and the additive relation between base activation and spreading activation is to imagine declarative memory was a sea of darkness with small rafts, i.e., chunks, floating everywhere on it.

- each raft has a small light
- the brightness of that light indicates its total activation:
  - the brighter that light is, the easier the raft is to find and grab
  - that is, we can retrieve it more accurately and more quickly

The light on each raft is powered by two power sources.

- one of them is a rechargeable battery stored on the raft itself (well, it's more like a capacitor, but let's ignore this)
- this reflects base activation, i.e., the history of previous usages of a chunk
- every time we use a chunk (retrieve a raft), we plug its 'local battery' in for a quick charge
- immediately after that, the battery will have more power, so the light will be brighter

The second source of power that can increase the brightness of the light on a raft is the current cognitive context, specifically the values held in the buffers.

- if these values are also stored on some of the rafts in declarative memory (that is, they are the values of some of the features of those chunks), they can act as wires delivering extra power to the lights on the rafts
- let's focus on a specific chunk in some buffer in our cognitive context
- each value j in that chunk has a set amount of battery power (these are the source activations, i.e., the W_j values)
- that power gets distributed to all the rafts in declarative memory that also store that value
- this immediately predicts that the more rafts a value in the cognitive context is connected with – in ACT-R parlance, the higher the 'fan' of a value –, the less power it will transmit to each individual raft
- this is called the 'fan effect'
- the amount of power / activation that 'spreads' from the goal/imaginal buffer (or any buffer in the cognitive context that we decide to spread activation from) depends not only on the 'battery power' W_j of each value j, but also on the specific 'wires' connecting the buffer and the rafts/chunks in declarative memory that share that value
- different wires have different 'resistance' characteristics S_{ij} , and the extra power boost W_j is modulated by the 'resistance'/strength of the connection.

Let us go through an example.

Suppose we have the word *car* in the imaginal buffer.

Also, our declarative memory consists of:

- two chunks x and y that are singular nouns (say, *book* and *pen*)
- one chunk z that is a plural noun (say, *books*)

The singular nouns x, y have two values in common with the *car* chunk in the imaginal buffer:

- the singular number
- the noun category

The plural noun z has only one value in common with the *car* chunk:

- the noun category

The activation of the plural noun z is calculated below. Recall that j are the values of the *car* chunk in the imaginal buffer:

- $j = 1$ for the form slot
- $j = 2$ for the meaning slot
- $j = 3$ for the syntactic category slot
- $j = 4$ for the number morphology slot

Since there are 4 total slots in the imaginal buffer chunk, the source activations are all set to $\frac{1}{4}$ (see above, with $W = 1, n = 4$).

Turning to association strengths, we note that only the value in the syntactic category slot (i.e., noun/N) spreads activation. This means that the association strengths for all the other values are zero:

- $S_{1z} = S_{2z} = S_{4z} = 0$

The fan of the value in the syntactic category slot, namely fan_3, is 4:

- the value is noun/N, and there are four nouns total in declarative memory: x, y, z and, we assume, also the *car* chunk currently in the imaginal buffer

The calculation, therefore, proceeds as follows:

$$\begin{aligned} A_z &= B_z + \sum_{j=1}^{m} W_j S_{jz} \\ &= B_z + W_1 S_{1z} + W_2 S_{2z} + W_3 S_{3z} + W_4 S_{4z} \\ &= B_z + \tfrac{1}{4} \cdot S_{1z} + \tfrac{1}{4} \cdot S_{2z} + \tfrac{1}{4} \cdot S_{3z} + \tfrac{1}{4} \cdot S_{4z} \\ &= B_z + \tfrac{1}{4} \cdot 0 + \tfrac{1}{4} \cdot 0 + \tfrac{1}{4} \cdot (S - \log \text{fan}_3) + \tfrac{1}{4} \cdot 0 \\ &= B_z + \tfrac{1}{4} \cdot (S - \log 4) \end{aligned}$$

In contrast, the activation of the singular noun x proceeds as shown below.

This time, the singular receives spreading activation both from the syntactic category value (N) and from the number specification (sg).

- the activation spreading from the number value is higher than the one spreading from the syntactic category value because there are 4 nouns total ($\text{fan}_3 = 4$), but only 3 of them are singular ($\text{fan}_4 = 3$)
- this makes intuitive sense: values that appear only in a handful of chunks are more predictive of these chunks, and should boost their activation more than values that are more frequent and therefore less discriminatory

$$\begin{aligned} A_x &= B_x + \sum_{j=1}^{m} W_j S_{jx} \\ &= B_x + W_1 S_{1z} + W_2 S_{2z} + W_3 S_{3x} + W_4 S_{4x} \\ &= B_z + \tfrac{1}{4} \cdot S_{1z} + \tfrac{1}{4} \cdot S_{2z} + \tfrac{1}{4} \cdot S_{3x} + \tfrac{1}{4} \cdot S_{4x} \\ &= B_x + \tfrac{1}{4} \cdot 0 + \tfrac{1}{4} \cdot 0 + \tfrac{1}{4} \cdot (S - \log \text{fan}_3) + \tfrac{1}{4} \cdot (S - \log \text{fan}_4) \\ &= B_x + \tfrac{1}{4} \cdot (S - \log 4) + \tfrac{1}{4} \cdot (S - \log 3) \end{aligned}$$

[ ]: