Personality and Individual Differences xxx (xxxx) xxx–xxx

Contents lists available at ScienceDirect

Personality and Individual Differences

journal homepage: www.elsevier.com/locate/paid

Using neural networks as models of personality process: A tutorial

Stephen J. Read*, Vita Droutman, Benjamin J. Smith, Lynn C. Miller

Department of Psychology, University of Southern California, Los Angeles, CA 90089-1061, USA

ARTICLE INFO

Keywords:
Neural networks
Computational modeling
Within-subjects variability
Connectionist modeling
Personality dynamics

ABSTRACT

This paper presents a tutorial for creating neural network models of personality processes. Such models enable researchers to create explicit models of both personality structure and personality dynamics, and to address issues of recent concern in personality, such as, "If personality is stable, then how is it possible that within subject variability in personality states can be as large as or larger than between subject variability in personality?" or "Is it possible to understand personality dynamics and personality structure within a common framework?" We discuss why one should want to use neural networks, review what a neural network model is, review a previous model we have constructed, discuss how to conceptualize issues in such a way that they can be computationally modeled, show how that conceptualization can be translated into a model, and discuss the utility of such models for understanding personality structure and personality dynamics. To build our model we use a neural network modeling package called *emergent* that is freely available, and a specific architecture called Leabra to build a runnable model that addresses one of the questions posed above: How can within subject variability in personality related states be as large as between subject variability in personality?

1. Introduction

The aim of this article is to provide interested researchers with a tutorial for how to conceptualize, translate, and implement a neural network model of personality processes. We use a specific published example of a neural network model of personality, (Read, Smith, Droutman, & Miller, 2017) article entitled, "Virtual Personalities: Using computational modeling to understand within-person variability", walking the reader through the process of developing such a model. In that article we leveraged the ability of neural network models to grapple with complex motivational and behavioral dynamics, to address personality puzzles that have long stymied the field. The puzzle we focused on is how personality traits can be both stable over time and simultaneously exhibit within-subject variability (in personality states) that is as large as between-subject variability in traits (Fleeson, 2001; Fleeson & Gallagher, 2009; Sherman, Rauthmann, Brown, Serfass, & Jones, 2015).

Despite what we think are the advantages of constructing neural network models for thinking about and modeling personality related processes, researchers face a steep learning curve in using these tools. Not only are the relevant concepts and software unfamiliar, there are few good examples of such models for personality psychologists. Cognitive models exist in abundance, but not personality related models. The intent of the current article is to help bridge that gap, affording a roadmap for scaffolding researchers in using neural network models, especially as they pertain to models of personality process.

In this tutorial, we: (1) start with an argument for why one should be interested in using computational models in personality research, and more specifically, why one might want to use a neural network model, (2) describe what a neural network model is, (3) review a "finished product" – an overview of the model presented in that work and what it tells us about personality processes, (4) take a "deep dive" into the process of conceptualizing the problem in the dynamic way necessary to build and implement such a computational model, (5) translate that conceptualization into an initial visual dynamic working space, (6) demonstrate the steps in implementing the model using the Leabra architecture in the *emergent* neural network software (Aisa, Mingus, & O'Reilly, 2008), and (7) discuss the meaning and utility of such models, and how to understand their limitations and constraints.

2. Theoretical/conceptual reasons for building a neural network model

Building a neural network model (or indeed any kind of computational model) has several benefits. First, it forces you to be explicit and specific about the constructs and processes in your model. Verbal (or mental) models make it very easy to gloss over gaps and holes in your model. Building a model frequently makes these gaps quite obvious (Oops! So how do X and Y interact??). For example, personality researchers often talk about models of person-situation interactions and

* Corresponding author.
E-mail address: read@usc.edu (S.J. Read).

https://doi.org/10.1016/j.paid.2017.11.015
Received 14 February 2017; Received in revised form 7 November 2017; Accepted 9 November 2017
0191-8869/ © 2017 Elsevier Ltd. All rights reserved.

give verbal descriptions of this process. But if you want to build a model of these interactions you must come to grips with exactly how situations and personality should be represented: What are the key features of each and how are they structured? And once we figure that out, how do personality and situation actually interact in a process? In our model that we discuss in the following, we conceptualize personality largely in terms of motivational systems and we conceptualize situations largely in terms of affordances for the pursuit of goals and motives. This conceptualization and implementation in our model leads to a natural process model concerned with how individual's motives interact with opportunities for the pursuit of those motives. With other possible conceptualizations of person and situation, if we tried to build a runnable model, we might very well discover that we don't have a clue as to how to build a runnable model in which these two things would interact in any reasonable way. Or we might build a model, but then discover it doesn't actually run. Either of these outcomes would force us to do some serious rethinking.

Second, once you have a runnable model you can then start testing it to see if it behaves as you predicted it should. Constructing a runnable model allows you to test the plausibility of your model and the assumptions you make about representations and about the processes that act on them. Having a runnable model does not “prove” that your conceptualization is correct, but it provides evidence that it “works.” If you cannot build a model that “works” then that casts doubt on how well you really understand the process.

But even if the model runs, you may discover a variety of different ways in which the model fails to capture real behavior. You may realize you left out key moderators or key processes. You may have misspecified functional relations in your model. You may have assumed additive relationships, when some other kind of relationship is a better fit. The exercise of actually running your model on a computer, rather than in your head, provides another check on whether you have adequately conceptualized the process in which you are interested.

In addition to the general benefits of building a computational model of personality related processes, we think there are a number of benefits to using a neural network architecture do so. Neural network models are well-established process models. They have been used extensively in cognitive psychology, cognitive science, and in cognitive neuroscience to model a wide range of cognitive and motivational processes. In the process, neural network researchers have come to understand a tremendous amount about the mechanisms involved in brain-like processing. For example, there is a growing understanding of how top-down and bottom-up processes interact, as a result of the massive bi-directional connectivity of the human brain. There is also a growing understanding of the dynamics of cognitive processing within a brain-like system. It is widely accepted that much of cognitive processing in the brain can be understood as a massively parallel constraint satisfaction process. Rather than being a largely serial process, cognitive processing is instead a parallel constraint satisfaction process in which multiple elements in the brain send activation back and forth to each other over weighted links, and the pattern of activation evolves over time until it reaches a stable state that represents the best solution to the constraints posed by the activation of all the connected systems and the weights between them. This processing is well understood in terms of the evolution of a dynamical system that evolves to what are called attractors, which are “low energy” states of the system that represent the solution to the constraints.

Further, learning is a key process in personality development and learning from experience is built into all modern neural network architectures. One can implement both differences in learning due to differences in experience, as well as individual differences in relevant biological parameters, such as learning rate. Thus, learning does not have to be separately implemented in a model, but one can instead take advantage of the existing infrastructure.

Because the processes and mechanisms within them are brain-like and relatively well understood, properly constructed neural network models draw upon a wide body of knowledge that has been gained about how the brain processes information, without having to explicitly implement them in the model. Building a model purely as a mathematical model does not provide a foundation based on this accumulated knowledge.

2.1. Why use a specific neural architecture such as Leabra?

There are several reasons why one might want to build a neural network model of personality processes in the specific Leabra neural network architecture that we use for our models. First, the Leabra architecture is biologically inspired and does a good job of providing a high level abstraction of neurobiological processes that are important in personality. It provides a natural way of capturing individual differences in such things as baseline motivation, individual differences in learning, and individual differences in synaptic efficacy that can be tied to such things as individual differences in density of neurotransmitter (e.g., dopamine, serotonin) receptors. Increasingly, work on personality processes and personality development is focusing on the underlying biological (genetic and neurobiology) bases of personality. Biologically inspired neural network architectures, such as Leabra, make it easier to build models that correspond to what we know of the underlying biological mechanisms. For example, Leabra has parameters that can be used to capture the impact of genetic factors on expression of synaptic receptors.

Finally, a key underlying theoretical mechanism in our model is that competition among motives and among alternative behaviors plays a key role in motivation and behavior. Such competition is built into the basic Leabra architecture, so it does not have to be separately implemented, although one can tweak how strongly things compete.

2.2. What are neural network models?

Neural network models are brain-inspired; their basic processing elements are based on the neurons in the brain and the synapses between them, and processing is modeled in term of the spread of activation among these simple elements. These kinds of models are very widely used in cognitive psychology, cognitive science, and neuroscience to model perceptual, cognitive, motivational and emotional processes. There are models of memory, learning, decision-making, language, motivation, emotions, and psychopathology. Among the psychological sciences, personality and social psychology are the least likely to use neural network models.

In a standard neural network model, the basic elements are nodes (representing neurons or clusters of neurons) and the weighted links between them. Processing occurs through the spread of activation from nodes along weighted links. The strength of the weight on the link between two nodes represents the strength of the influence of the sending node on the receiving node. Nodes in the network receive activations from other nodes that are connected to them. The level of activation received is a function of the input from each sending node multiplied by the weight on the link between the sending and the receiving node. The node sums the activation from all the sending nodes and then sends activation to subsequent nodes based on this sum. The output function for a node can take several typical forms. First, the output can be binary, such that the node sends activation if a threshold is exceeded and otherwise does not. Second, the output activation can be a linear function of the input. Third, it can be an S-shaped or sigmoidal function of the inputs, where activation rises sharply in the middle of the range, and then asymptotes. Some kind of S-shaped function is most typical, as it allows for more powerful learning and processing in the network.

Most neural networks have what are called “hidden layers” which are layers in the processing stream between input and output. Such hidden layers can learn more complex representations that are combinations of lower level features. It is well established that using a

Download English Version:

https://daneshyari.com/en/article/11016181

Download Persian Version:

https://daneshyari.com/article/11016181

Daneshyari.com