Canadian Journal of Experimental Psychology / Revue canadienne de psychologie expérimentale
2018, Vol. 72, No. 3, 153–170

1196-1961/18/$12.00 http://dx.doi.org/10.1037/cep0000135

Key-Finding by Artificial Neural Networks That Learn About Key Profiles

Michael R. W. Dawson and Jasen A. Z. Zielinski
University of Alberta

We explore the ability of a very simple artificial neural network, a perceptron, to assert the musical key of novel stimuli. First, perceptrons are trained to associate standardized key profiles (taken from 1 of 3 different sources) to different musical keys. After training, we measured perceptron accuracy in asserting musical keys for 296 novel stimuli. Depending upon which key profiles were used during training, perceptrons can perform as well as established key-finding algorithms on this task. Further analyses indicate that perceptrons generate higher activity in a unit representing a selected key and much lower activities in the units representing the competing keys that are not selected than does a traditional algorithm. Finally, we examined the internal structure of trained perceptrons and discovered that they, unlike traditional algorithms, assign very different weights to different components of a key profile. Perceptrons learn that some profile components are more important for specifying musical key than are others. These differential weights could be incorporated into traditional algorithms that do not themselves employ artificial neural networks.

Keywords: artificial neural networks, key-finding, perceptrons

A critical component of listening to music is identifying its musical key. When a piece of music is in a particular musical key, certain tones or musical pitches are more stable than are others. That is, a musical key establishes the tonality of a piece. Human listeners, whether musically trained or not, are able to identify musical key very rapidly (Butler, 1989). Not surprisingly there is considerable interest in proposing procedures for musical key-finding, both to contribute to theories of music perception and to develop computer algorithms for automatically asserting the keys of musical stimuli (Albrecht & Shanahan, 2013; Frankland & Cohen, 1996; Handelman & Sigler, 2013; Holtzman, 1977; Longuet-Higgins & Steedman, 1971; Sapp, 2005; Shmulevich & Yli-Harja, 2000; Temperley, 1999, 2004, 2007; Temperley & Marvin, 2008; Tillmann, Bharucha, & Bigand, 2000; Vos & Van Geenen, 1996).

One model of key-finding is derived from empirical results concerning the interrelationships between different pitches in a given musical context. Human listeners, even those without formal musical training, organize musical events according to tonal hierarchies (Krumhansl, 1979, 1990; Krumhansl & Shepard, 1979). A tonal hierarchy represents the relative fit or stability of one pitch to another in a given musical context. Tonal hierarchies can be empirically determined using the probe tone method. This paradigm has subjects use a seven-point scale to rate how well a tone fits into the context of preceding tones heard before this probe is presented; a rating of 1 indicates *very bad* whereas a rating of 7 indicates *very good*.

When the musical context establishes a particular tone as being central, the probe tone method produces a regular pattern of ratings (see Table 1, first two rows). From this table it can be seen that when the context establishes a particular major key, the tonic note of the established key (i.e., degree 0, which is the pitch C in the key of C major, C# in the key of C# major, and so on) is the most stable, because it receives the highest rating of 6.35. The next most stable tone is a perfect fifth (7 semitones) above the tonic, with the second highest rating of 5.19. The tones 1, 10, and 3 semitones above the tonic are the least stable, because they receive the lowest ratings in the probe tone method. A related tonal hierarchy is obtained when a context establishes a minor key in the probe tone method, as is shown in the second row of Table 1. Importantly, the same basic tonal hierarchy is obtained for each major key (or for each minor key) when a different tonal center is established. For instance, two different major keys will generate the same basic pattern of ratings; the difference between the two is simply that the pattern shifts to align with the tonal center of the particular key.

Tonal hierarchies provide the foundation for one influential model of key-finding proposed by Krumhansl and Schmuckler (Krumhansl, 1990). The model uses tonal hierarchies to create a set of standardized tone profiles, one for each of the 12 major and for each of the 12 minor musical keys. In the Krumhansl/Schmuckler key-finding algorithm, a to-be-analyzed musical stimulus is also represented as a tone profile. This is accomplished by determining the total duration of each pitch-class in the stimulus. Pitch-class is an equivalence class of different musical pitches that are related by the musical interval of an octave. For instance, middle C, the C an octave higher, and the C two octaves higher all belong to the pitch-class 'C.' To use the Krumhansl/Schmuckler algorithm, one tabulates the total number of beats in the stimulus that involve hearing the pitch class A, the total number of beats of the pitch class A#, and so on. Once the stimulus is represented in this fashion, correlations are computed between the stimulus' profile and each of the 24 standardized tone profiles. The algo-

This article was published Online First May 8, 2017.
Michael R. W. Dawson and Jasen A. Z. Zielinski, Biological Computation Project, Department of Psychology, University of Alberta.
Correspondence concerning this article should be addressed to Michael R. W. Dawson, Department of Psychology, University of Alberta, Edmonton, Alberta, Canada T6G 2P9. E-mail: mdawson@ualberta.ca

Table 1
The Three Sets of Key Profiles Used in Simulation 1: The Major and Minor Profiles From the Krumhansl-Schmuckler Algorithm Provided by Krumhansl (1990; Source Indicated as KS), the Major and Minor Profiles From Temperley (2007; Source Indicated as T), and the Major and Minor Profiles From Albrecht and Shanahan (2013; Source Indicated as AS)

		Degree of pitch-class relative to tonal center of context											
Source	Type	0	1	2	3	4	5	6	7	8	9	10	11
KS	Maj	6.35	2.23	3.48	2.33	4.38	4.09	2.52	5.19	2.39	3.66	2.29	2.88
	Min	6.33	2.68	3.52	5.38	2.60	3.53	2.54	4.75	2.98	2.69	3.34	3.17
T	Maj	.748	.060	.488	.082	.670	.460	.096	.715	.104	.366	.057	.400
	Min	.712	.084	.474	.618	.049	.460	.105	.747	.404	.067	.133	.330
AS	Maj	.24	.01	.11	.01	.14	.09	.02	.21	.01	.08	.01	.08
	Min	.22	.01	.10	.12	.02	.10	.01	.21	.06	.02	.06	.05

Note. Scale degree 0 is assumed to be the tonic pitch, etc.

rithm identifies the standardized profile that produces the highest correlation, and asserts that this is the key of the musical stimulus. Krumhansl (1990) reports this algorithm performs very well. It may also serve as a model of the cognitive processes involved when human listeners establish tonal centers (Frankland & Cohen, 1996; Schmuckler & Tomovski, 2005).

However, the performance of the Krumhansl-Schmuckler algorithm is not perfect. For example, when examining performance on a test set of 492 selections of classical compositions, Albrecht and Shanahan (2013) observed that Krumhansl-Schmuckler correlation algorithm is only 74.2% accurate. Second, the performance of the algorithm varies depending upon whether it is presented stimuli in major or in minor keys. In Albrecht and Shanahan's test set, the Krumhansl-Schmuckler algorithm generated 69.0% accuracy to major key compositions, while it was 83.2% accurate in assigning minor keys.

Some researchers have investigated variations of the Krumhansl-Schmuckler algorithm in an attempt to improve key-finding performance. In general, these variations explore two different avenues. The first involves replacing the Krumhansl-Schmuckler tone profiles with alternative profiles. For example, both Temperley (2004) and Albrecht and Shanahan (2013) derive new tone profiles from statistical analyses of large corpuses of music pieces. These alternative tone profiles are also provided in Table 1.

The second avenue for exploring variations of the Krumhansl-Schmuckler algorithm involves comparing inputs to tone profiles using some method other than correlation. For instance, Temperley uses the Bayesian probability equation to choose the most probable key to assign to an input stimulus. In contrast, Albrecht and Shanahan assume that the tone profiles for each musical key, and the to-be-classified stimulus, are all points located in a 12-dimensional space; they choose the key whose point in this space is the shortest distance from the point representing the stimulus.

The current manuscript explores a new variation of a key-finding algorithm that employs tone profiles. Its general purpose is to determine whether simple artificial neural networks trained on tone profiles can serve as plausible methods for key-finding. It describes the performance of artificial neural networks that assert musical keys after learning the keys associated with different sets of tone profiles.

Training Key-Finding Networks on Tone Profiles

Artificial Neural Networks

An artificial neural network is a computer simulation of a system of simple processors that send signals to one another through weighted connections (McClelland & Rumelhart, 1986; Rumelhart & McClelland, 1986). In general, an artificial neural network consists of layers of processing units; signals pass through weighted connections from one layer to the next. The function of a typical network is to generate a desired response to each stimulus in a set of training patterns. Each stimulus is encoded as a pattern of activity in a layer of input units. The network's response to the input is represented as a pattern of activity in a layer of output units. Intervening layers of processors in the system, called hidden units, may be required to detect more complex features of the stimulus that allow the network to make a correct response.

An output unit in an artificial neural network is analogous to a neuron, and behaves as follows: First, it computes the total signal being received from other processors in the network, called the *net input*. Second, the output unit converts this total signal into a numerical value that represents its activity or response. This is accomplished using an equation called the *activation function*. In the current manuscript, the output units use the logistic function (Rumelhart, Hinton, & Williams, 1986) as the activation function. The logistic activation function is a sigmoid-shaped function that squashes net input, which can in principle range from positive to negative infinity, into activity that ranges between 0 and 1. The logistic equation is $a = 1/(1 + exp(-net + \theta))$; in this equation a is output unit activity, net is net input, and θ is a parameter called *bias* that is analogous to the output unit's threshold.

An artificial neural network's response to an input pattern is defined by its set of weighted connections between the various units in the network. However, artificial neural networks are generally not programmed in any conventional sense. Instead, they are taught. Networks receive a sequence of input patterns, and learn by adjusting their connection weights based on feedback about their responses. In the networks considered below, learning is error-driven. After a network responds to a presented stimulus, error—the difference between the desired and actual response for each output unit—is calculated. A gradient descent learning rule (Dawson, 2008) then uses the computed errors to modify the network's connection weights. These modifications change weights (and biases) in such a way that network error is reduced. With repeated presentations of the stimuli in a training set, a network's error is reduced to the point that it can be claimed that the network has learned the desired stimulus-response mapping.

Networks and Music Cognition

Artificial neural networks have been applied to a wide variety of problems in music and in musical cognition (Bharucha, 1999; Fiske, 2004; Griffith & Todd, 1999; Todd & Loy, 1991). A variety of network architectures have been used to study to such topics as classifying pitch and tonality, assigning rhythm and meter, classifying and completing melodic structure, and composing new musical pieces. Let us briefly consider some examples of musical connectionism.

Connectionist networks can accomplish a variety of tasks that require classification of basic elements of Western music (e.g., pitch, tonality, and harmony). Artificial neural networks have been trained to classify chords (Laden & Keefe, 1989; Yaremchuk & Dawson, 2005, 2008), to assign notes to structures similar to the tonal hierarchy (Leman, 1991; Scarborough, Miller, & Jones, 1989), to model the effects of musical expectations on musical perception (Bharucha, 1987; Bharucha & Todd, 1989), to add harmony to melodies (Berkeley & Raine, 2011; Shibata, 1991), to determine the musical key of a melody (Griffith, 1995), to identify a melody even when it has been transposed into a different key (Benuskova, 1995; Bharucha & Todd, 1989; Page, 1994; Stevens & Latimer, 1992), and to detect the chord patterns in a composition (Gjerdingen, 1992). Artificial neural networks can also handle other important aspects of music that are independent of tonality, such as assigning rhythm and meter (Desain & Honing, 1989; Griffith & Todd, 1999; Large & Kolen, 1994) or generating preferences for, or expectancies of, particular rhythmic patterns (Gasser, Eck, & Port, 1999).

The examples cited above generally involve using artificial neural networks to detect properties of existing music. The ability of networks to process tonality, harmony, meter, and rhythm also permits them to generate new music. Composition has in fact been of the most successful applications of musical connectionism. Networks can compose single-voiced melodies on the basis of learned musical structure (Mozer, 1991; Todd, 1989), can compose harmonized melodies or multiple voiced pieces (Adiloglu & Alpaslan, 2007; Bellgard & Tsang, 1994; Hoover & Stanley, 2009; Mozer, 1994), can improvise when presented new jazz melodies and harmonies (Franklin, 2006), and can improvise by composing variations on learned melodies (Nagashima & Kawashima, 1997).

The ability of artificial neural networks to exploit similarity relationships positions them to capture regularities that are difficult to express in language or using formal rules (Loy, 1991). This permits networks to solve musical problems that involve very abstract properties. For example, human subjects can accurately classify the genre or style of a short musical selection within a quarter of a second (Gjerdingen & Perrott, 2008). The notion of style or genre is too vague to be formalized in a fashion suitable for a classical rule governed system (Loy, 1991). However, neural networks are up to the task, and can classify musical patterns as belonging to the early works of Mozart (Gjerdingen, 1990), can classify selections as belonging to different genres of Western music (Mostafa & Billor, 2009), can evaluate the affective aesthetics of a melody (Cangelosi, 2010; Coutinho & Cangelosi, 2009; Katz, 1995), and that can even predict the possibility that a particular song has "hit potential" (Monterola, Abundo, Tugaff, & Venturina, 2009).

Clearly artificial neural networks are popular models for studying music cognition. Why might this be so? It has been argued that artificial neural networks provide five different advantages for this research domain (Bharucha, 1999). First, artificial neural networks can account for how music is learned. Second, connectionist theories of such learning are biologically plausible. Third, networks provide accounts of music perception phenomena, such as contextual effects and the filling-in of incomplete information. Fourth, networks exploit similarity-based regularities that are important in theories of musical cognition. Fifth, networks may discover regularities that elude other analyses.

The current paper describes simulations that to a certain extent react against this final point of Bharucha (1999). These simulations belong to a broader research program (Dawson, in press) that does not agree that a main goal of musical connectionism is to capture *informal* regularities. Instead, this program uses musical networks to reveal new *formal* properties of music. It does so by training simple networks on basic musical tasks, such as identifying the tonic or mode of a scale, or classifying chords into types. After training is complete, the internal structure of a network is examined in an attempt to determine how it solves the problem that it has been taught. In many cases, this interpretation reveals new insights into musical regularities.

The current paper illustrates this approach by training a very simple type of network, called a perceptron, to assert musical key by training it using stimuli related to the tonal hierarchy. This stimuli are taken from some traditional (i.e., nonconnectionist) models of key-finding (Albrecht & Shanahan, 2013; Krumhansl, 1990; Temperley, 2007). After this training, a perceptron is presented a set of 296 new musical stimuli to assess its ability to assert their musical keys. In general, we find that these networks perform quite well on these new stimuli. We then examine the connection weights of the trained network, and discover that these weights indicate that the different components of a tonal hierarchy are not equally informative signals of musical key. That is, the networks learn that some stimulus components are more important than others are, and assign them much stronger weights.

The remainder of this paper proceeds as follows: First, we introduce the basic properties of a perceptron, and describe how it can be trained for key-finding. Second, we describe the results of this training, and evaluate the ability of trained perceptrons to assert the musical keys of novel stimuli. Third, we examine the internal structure of these networks, and discuss how these weights can be used to inform some current models of key-finding. We end with a general discussion of the implications of these results, as well as how networks like the ones discussed below might be situated in the experimental study of musical cognition.

A Perceptron for Key-Finding

Most previous studies of how neural networks process musical tonality or musical key have employed complex, dynamic networks whose components interact over time. Such networks evolve into stable states that reflect responses to stimuli, and are typically used to model changes in musical qualities over time.

For instance, one such model is a network of interacting oscillators whose frequencies shift into stabilized patterns (Large, Kim, Flaig, Bharucha, & Krumhansl, 2016). These patterns can be interpreted as indicating that some tonal relationships are more stable than are others in the context, and can be related to the tone profiles introduced earlier. Another model is a self-organizing map whose surface is arranged around the shape of a torus, and which is trained on the 24 Krumhansl-Schmuckler profiles (Toivainen, 2005; Toiviainen & Krumhansl, 2003). When this network is combined with a short-term memory (STM), it can represent dynamic changes in tonality as a musical stimulus changes over time. A similar dynamic network (Bharucha, 1987) uses different processing units to represent various chords, musical keys, and musical tones. Signals flow back and forth between these units; changes in unit activity reflect changes over time of such attributes as key or stable chords within a particular context (i.e., the various stimuli that have been presented to the network, which are remembered for a brief period).

In contrast to these complex dynamic networks, the current manuscript explores a much simpler network: the perceptron (Rosenblatt, 1958, 1962). A perceptron is a network that consists of a layer of input units that are directly connected to a layer of output units. Perceptrons do not include any intermediate layers of processors. Perceptrons also do not settle into stable states, because they do not permit signals to move back and forth between processors over a period of time. Instead, perceptrons are feedforward systems in which signals generated by activating input units are scaled by weighted connections and then cause output unit activity. In a trained perceptron, this output activity is the response to the presented stimulus. Below we explore the behavior of such networks when they learn a mapping between tone profiles and musical keys, and are then faced with asserting the musical keys of novel stimuli.

There are three main reasons for choosing the perceptron as the type of network for investigating key-finding. First, the type of learning rule used to train perceptrons is formally and empirically related to psychological models of associative learning (Dawson, 2008), and perceptrons have successfully been used to model behaviors observed in animals and humans (Dawson, Kelly, Spetch, & Dupuis, 2010). Thus perceptrons are psychologically plausible models. Second, and as discussed in more detail below, if artificial neural networks are to contribute to musical cognition, then these contributions will come from interpreting the internal structure of trained networks (Dawson, in press). Perceptrons are straightforward to interpret because of their simple structure. Third, when a perceptron's output units use the logistic activation function, their outputs and their connection weights are strongly related to elements of probability theory (e.g., Dawson & Gupta, 2017). Probability theory also has a growing importance in accounts of music cognition (e.g., Temperley, 2007).

The general structure of a key-finding perceptron is illustrated in Figure 1. It consists of 12 input units, each of which is associated with a particular pitch class of Western music. This layer of input units can represent a profile for a particular musical key, or a pitch class profile for a musical composition. The perceptron also includes 24 output units, each of which is associated with a different musical key. The task for the perceptron is to learn the mapping from an input pattern to an output key. For example, once training is complete it is expected that if the perceptron is presented the tone profile for the key of A major then it will generate high activity in the A major output unit while generating low activity in each of the other output units.

In the study presented below, three different types of key-finding perceptrons were trained. The first was trained on tone profiles taken from the Krumhansl-Schmuckler algorithm (Krumhansl, 1990), the second was trained on tone profiles taken from the Temperley (2007) algorithm, and the third was trained on

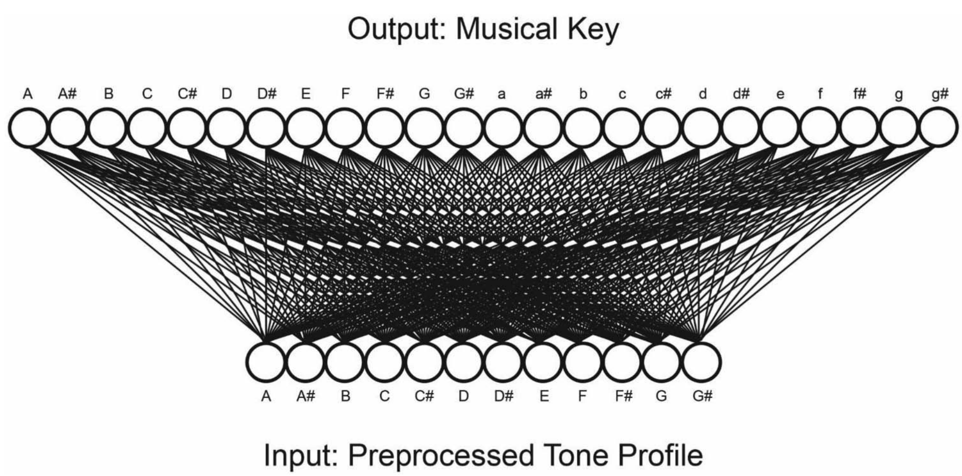

Figure 1. A perceptron that can be trained to generate the musical key associated with a key profile that is presented to its input units. The 12 input units can represent a key profile or a musical stimulus represented as an analogous profile. The 24 output units each represent a different musical key. The lines between input and output units represent connections; the weights of these connections are modified by training the perceptron on a set of stimulus-response pairs.

tone profiles taken from the Albrecht and Shanahan (2013) algorithm.

These perceptrons represent variations of the original Krumhansl-Schmuckler correlational algorithm. Consider a perceptron trained on the Krumhansl-Schmuckler key profiles. Even though it uses the same set of key profiles, this perceptron differs from the Krumhansl-Schmuckler algorithm in two ways. First, it does not directly match tone profiles to musical keys. Instead, it uses its connection weights to transform tone profiles before their signals reach the output units. Second, this network does not use correlations to match stimuli with desired musical keys. Instead, the output units take the signals from (transformed) input profiles and then further transform them by applying a nonlinear activation function, the logistic equation. Because of these two types of differences, a perceptron trained on the Krumhansl-Schmuckler key profiles may respond differently than will the Krumhansl-Schmuckler correlation algorithm itself.

The Logic of Perceptrons

Before turning to our simulations, let us briefly consider an important question: Why might we be interested in studying key-finding with perceptrons? In spite of their simple structure, the perceptrons described below illustrate several of the advantages that have been claimed for connectionist explorations of music cognition (Bharucha, 1999).

First, perceptrons offer a natural avenue for considering the role of learning in studies of key-finding and tonality. The tonal hierarchies derived from probe tone studies are highly correlated with the frequency of occurrence of various pitch-classes in musical corpuses; it has been hypothesized that human listeners learn pitch-class distributions from the music that they hear (Krumhansl, 1990). However, to our knowledge the literature on key-finding has not proposed how this learning might occur.

The learning rule used to train perceptrons is strongly related to the psychological literature. It is formally equivalent to the seminal Rescorla-Wagner model of classical conditioning (Rescorla & Wagner, 1972); perceptrons have provided many insights into this type of associative learning (Dawson, 2008; Gluck & Bower, 1988; Sutton & Barto, 1981). Perceptrons have also successfully modeled learning to navigate from environmental cues and learning to match response probabilities to the likelihood of events occurring in the world (Dawson & Dupuis, 2012; Dawson, Kelly, Spetch, & Dupuis, 2008; Dawson et al., 2010; Dupuis & Dawson, 2013a).

Of course, this is not to say that the particular approach to learning keys that is described below—the supervised pairing of each musical key with a tone profile—is a realistic account of how humans actually learn about musical keys. Some experimental results suggest that human key-finding cannot be equated with the typical key-finding process that assigns a precise key to a musical stimulus.

For instance, in one psychological study subjects listened to different melodies that were randomly generated so that their tonal content was consistent with a specific tonal profile (Temperley & Marvin, 2008). That is, stimuli were stochastically generated so that the likelihood of a tone being added to a stimulus matched the likelihood of its occurrence in the profile. For each stimulus, subjects indicated one of 12 tones to specify the tonic of the stimulus, as well as whether the stimulus was in the major or minor mode. Temperley and Marvin found that while subject responses were significantly better than chance, they were not as accurate as expected. In general, subjects were only about 52% accurate in indicating the expected key of a stimulus, and there were not high levels of agreement between the judgments of different subjects.

Results such as those reported by Temperley and Marvin (2008) raise questions about what exactly human listeners learn when they experience tonality. From our perspective, the simulations described below are a first step in permitting such issues to be explored with artificial neural networks. In this first step, we determine whether a very simple kind of network can learn the same kind of mappings employed in other theories of key-finding (i.e., a relationship between a particular tone profile and a specific musical key), and can then adequately generalize this learning in terms of asserting the musical keys of novel stimuli. If this capability can be demonstrated, then this would suggest that associative learning mechanisms are worthy of further exploration in an attempt to develop more realistic learning theories. Such an exploration could involve exploring a wider variety of responses related to tonality (e.g., judgments of mode only, judgments of whether two stimuli are in the same key or not, etc.). The performance of such networks could then be related to analogous human judgments, for instance by using a methodology similar to that employed by Temperley and Marvin, but having subjects make different sorts of judgments.

The second characteristic of perceptrons that is related to Bharucha's (1999) list of connectionist advantages is that the performance of our perceptrons demonstrates the power of exploiting similarity-based regularities. We create our perceptrons by teaching them the mapping between musical keys and a small number of tone profiles taken from key-finding theories. Then the perceptrons exploit the similarity relationships between these mappings in order to assign musical keys to a large number of novel stimuli.

A third connectionist advantage cited by Bharucha (1999) is that networks may discover regularities that elude other analyses. The networks that we describe below do provide an interesting novel discovery about using tone profiles to assert musical key. Network interpretation plays a key role in the simulations described below. By examining the connection weights of key-finding perceptrons, we discover that some components of a single tone profile are far more important for asserting key than are others, as was noted above. Importantly, this difference in importance is over and above the different sizes of the components that are already present in the tone profile (e.g., the different values across a row of Table 1). This result suggests possible alterations to existing key-finding algorithms (Albrecht & Shanahan, 2013; Krumhansl, 1990; Temperley, 2007).

Method

Design

This simulation study proceeds in two phases. First, a network is trained to identify musical key using one of the three possible sets of mean-centered normalized tone profiles described in more detail below (see Table 2). Second, after successfully completing the first phase, a network is tested on its ability to assert the musical key of 296 different musical selections. These test stimuli

Table 2
The Three Sets of Mean-Centered Normalized Key Profiles Used in the Simulation

Source	Type	Degree of pitch-class relative to tonal center of context											
		0	1	2	3	4	5	6	7	8	9	10	11
KS	Maj	.655	−.286	−.001	−.263	.205	.139	−.220	.390	−.250	.041	−.273	−.138
	Min	.668	−.234	−.026	.433	−.253	−.024	−.268	.278	−.160	−.231	−.071	−.113
T	Maj	.442	−.330	.151	−.305	.355	.119	−.289	.405	−.280	.014	−.333	.052
	Min	.423	−.308	.146	.313	−.348	.130	−.283	.463	.064	−.327	−.251	−.022
AS	Maj	.575	−.287	.103	−.287	.199	.040	−.250	.485	−.276	−.012	−.280	−.009
	Min	.563	−.318	.086	.164	−.264	.082	−.293	.538	−.087	−.252	−.091	−.128

Note. These are all preprocessed versions of the original profiles presented in Table 1.

represent four different sources. The first is the collection of 48 preludes and 48 fugues from both books of J.S. Bach's *Well-Tempered Clavier*. These compositions are a typical test bed for key-finding algorithms (Albrecht & Shanahan, 2013; Temperley, 1999). The second is a collection of 24 preludes composed by Frederic Chopin as his *Opus 28*. One prelude was written in each musical key. These too are often used to test the accuracy of key-finding theories; for instance, they pose some difficulty for the Krumhansl-Schmuckler algorithm (Krumhansl, 1990). The third test set is the 24 *Preludes, Op. 67* composed by Johann Hummel. As is the case for the Chopin preludes, Hummel composed one prelude in each musical key. The fourth test set, the only selection of music not from the classical genre, consists of 152 Nova Scotia folk songs from *Songs and Ballads from Nova Scotia* (Creighton, 1932).

Training Patterns

One advantage that the Krumhansl-Schmuckler algorithm has via its use of correlations to compare stimuli to tone profiles is that this operation is not affected by stimulus magnitude. This is useful because when a long musical selection is summarized as a pitch-class profile, the magnitude of each value in its profile is expected to be larger than would be observed when a shorter musical selection is summarized in the same way. This is simply because on average one would expect to find more instances of each pitch-class in a longer piece than in a shorter one. Fortunately, correlation is not sensitive to stimulus magnitude. Instead, it computes the similarity of two patterns by only considering their relative directions, and ignoring their relative magnitudes.

As the artificial neural networks to be considered below do not use correlations, their outputs can be affected by differences in stimulus magnitude. For this reason each pattern presented to a network is first mean-centered and then normalized to unit magnitude. This is accomplished by considering each pattern to be a 12-dimensional vector, with each entry in a vector being a profile value. To mean-center this vector, the mean of its entries is computed, and is then subtracted from each entry. To normalize the mean-centered vector, its magnitude is computed, and then each entry in the vector is divided by this value. When normalized in this fashion every pattern presented to the network can be described as a vector whose magnitude is equal to one. Note that the operations used to mean-center and normalize a profile will not affect the correlation between the profile and any set of numbers with which it is being compared. Table 2 presents the three different sets of tone profiles from Table 1 after they have been mean-centered and normalized.

To build a training set for a perceptron trained on the Krumhansl-Schmuckler profiles, the mean-centered normalized major tone profile in the first row of Table 2 was used to generate a key profile for each of the 12 major keys by associating the values given in Table 2 with the appropriate input unit. That is, for the key of C major the value of 0.655 was presented to the input unit representing the pitch-class C, the value of −0.286 was presented to the input unit representing the pitch-class C#, and so on. Similarly the stimulus for the key of C# major involved presenting the value of 0.655 to the input unit representing the pitch-class C#, the value of −0.286 was presented to the input unit representing the pitch class D, and so on. A similar procedure created input stimuli for the 12 different minor keys using the normalized minor key profile in the second row of Table 2. As a result, the training set consisted of 24 different input patterns, one for each musical key. A second set of 24 training patterns was created by applying this method with the two mean-centered normalized Temperley profiles from Table 2, and a third set of 24 training patterns was created by applying this method with the two mean-centered normalized Albrecht-Shanahan profiles from Table 2.

To use these profiles for training, each input pattern is paired with a desired pattern of responses. For a particular stimulus, this desired response pattern is one in which the output unit associated with the stimulus' key is given a value of 1, and all other output units are given values of 0. When presented a stimulus, the perceptron learns to turn a particular output unit on (i.e., assert a particular musical key) and to turn all of the other output units off.

Testing Patterns and Methodology

As noted earlier, 296 test stimuli were taken from four different collections of musical compositions. Each of these four collections of musical selections is available in the kern file format at http://kern.humdrum.org/. As a result, they can be analyzed using the Humdrum toolkit (Huron, 1999). Humdrum's *key* command can be used to represent each of the 296 test stimuli in the format required by the Krumhansl-Schmuckler algorithm. That is, the total duration of each of the 12 Western pitch-classes was computed for each stimulus, producing a 12-entry vector representation. Each of these vectors was then mean-centered and normalized to unit magnitude prior to being presented to a trained network. This preprocessing renders the test stimuli into a format that is identical to that used to represent the training stimuli. It also ensures that network performance is not affected by the varying magnitudes of each of these test patterns.

During the test phase, network performance is assessed using a procedure analogous to the Krumhansl-Schmuckler algorithm. That is, when a test stimulus is presented to a network it produces activity in 24 different output units, each one representing a different musical key (see below). The output unit that generates the highest activity is taken to indicate the musical key being asserted by the network. The accuracy of this assertion is then compared to the key in which the stimulus was actually composed (information that is provided as part of the test stimulus' kern file).

Network Architecture, Training, and Testing

Each network is a perceptron of the type illustrated in Figure 1. Networks are trained with a gradient descent rule for perceptrons (Dawson, 2004, 2008) using the Rosenblatt software program (Dawson, 2005); this program is available as freeware from the first author's website. Before training begins all connection weights in the network are set to random values between −0.1 and 0.1. The biases (θ) of the output units are all initialized to 0, but are then modified by the learning rule during training. A learning rate of 0.50, which is typical for this architecture (Dawson, 2005) is employed. Networks are trained epoch by epoch, where each of the 24 training patterns is presented once each epoch. The order of pattern presentation is randomized each epoch. Training continues until the network generates a 'hit' for every output unit for each of the 24 patterns in the training set. A hit is defined as an activity of 0.90 or higher when the desired output is 1 or as an activity of 0.10 or lower when the desired output is 0. Once a network converges to a solution for the training patterns it is then presented each of the 296 test patterns without additional training, and its responses to these patterns are recorded.

Because each network is trained from a random configuration of small initial set of connection weights, it is possible that different networks can achieve qualitatively different end states via training. For this reason, 10 different perceptrons are trained on each of the three sets of training patterns, making a total of 30 different perceptrons being studied in this simulation. Each of these perceptrons can be viewed as a different 'subject' in a simulation experiment.

Results

Training

All 30 perceptrons successfully learned to map mean-centered normalized tone profiles onto musical keys. On average the 10 perceptrons trained on the Krumhansl-Schmuckler profiles converge after 1354.3 epochs of training ($SD = 0.823$). On average the 10 perceptrons trained on the Temperley profiles converge after 1453.4 epochs of training ($SD = 1.17$). On average the 10 perceptrons trained on the Albrecht-Shanahan profiles converge after 1864.0 epochs of training ($SD = 1.25$).

As 10 different versions of each perceptron were trained, and as each of these began from a different set of small randomly selected initial weights, are there any qualitative differences between the solutions reached by different versions of the same perceptron type? Interestingly, it appears that each perceptron trained on the same set of profiles reaches essentially the same solution (i.e., the same set of connection weights, as is detailed later), and generates essentially the same responses to stimuli. For example, we computed the standard deviation (across the 10 different versions of each perceptron) of the responses of each output unit to each of the 296 test patterns (i.e., a total of 7,104 different standard deviations). On average, the standard deviation of one output unit to one test pattern was 0.0003; the maximum standard deviation observed was 0.01. This was true for each of the three different types of perceptrons. In other words, despite starting from different initial states, different perceptrons trained on the same profiles converged on nearly identical sets of connection weights, and generated nearly identical responses (across all 24 output units) to each of the 296 test stimuli.

Key-Finding Accuracy for Test Stimuli

We conducted testing by presenting a trained perceptron each member of a set of 296 musical stimuli, each of which represented as a mean-centered and normalized tone profile. The output unit that generates the highest activity is used to assert the musical key of the presented stimulus. This response is then compared to the known key of the stimulus (i.e., the asserted key in the kern file for the stimulus). The general performance of each of the three types of perceptron for these novel stimuli is presented in Table 3, which also presents the performance of the Krumhansl-Schmuckler correlation algorithm for these same stimuli.

Because the responses discussed above are binary (a perceptron is either correct or not in asserting the key of a stimulus), and there are a large number of test stimuli that can be used to assess overall accuracy, differences in performance of the algorithms can be statistically compared using a test of proportions (Hays, 1981). The test of proportions takes advantage of the fact that for large samples (i.e., greater than 20 cases) the binomial distribution can be approximated by a normal distribution. It uses this approximation to convert the difference between two proportions into a z-score.

With respect to overall performance on the 296 test stimuli, tests of proportions indicate that the perceptron trained on the Krumhansl-Schmuckler profiles performed significantly poorer than the Krumhansl-Schmuckler correlation algorithm ($z = -6.498$, $p <$ .001), significantly poorer than the Temperley perceptron

Table 3
Overall Accuracy in Percentage Correct for Various Key-Finding Methods

Key-finding method	All test stimuli	Classical test stimuli only
KSP	74.4	77.1
TP	88.4	94.4
ASP	88.6	93.4
KS Corr	87.1	93.8

Note. Accuracy is provided for all 296 test stimuli as well as for only the 144 stimuli belonging to the classical genre (i.e. this column excludes the Nova Scotia folk songs). KSP indicates Krumhansl-Schmuckler perceptron, TP indicates the Temperley perceptron, ASP indicates the Albrecht-Shanahan perceptron, and KS Corr indicates the Krumhansl-Schmuckler correlation algorithm.

($z = -7.535$, $p < .001$), and significantly poorer than the Albrecht-Shanahan perceptron ($z = -7.705$, $p < .001$). The performance of the perceptron trained on the Temperley profiles did not differ significantly from the Krumhansl-Schmuckler correlation algorithm ($z = 0.689$, $p < .31$) or from the Albrecht-Shanahan perceptron ($z = -0.111$, $p < .40$). The performance of the Albrecht-Shanahan perceptron did not differ significantly from the Krumhansl-Schmuckler correlation algorithm ($z = 0.795$, $p <$ .29).

An identical pattern of statistical significance is revealed when performance on just the 144 classical test stimuli is considered. Tests of proportions indicated that the perceptron trained on the Krumhansl-Schmuckler profiles performed significantly poorer than the Krumhansl-Schmuckler correlation algorithm ($z = -8.310$, $p < .001$), significantly poorer than the Temperley perceptron ($z = -9.029$, $p < .001$), and significantly poorer than the Albrecht-Shanahan perceptron ($z = -7.878$, $p < .001$). The performance of the perceptron trained on the Temperley profiles did not differ significantly from the Krumhansl-Schmuckler correlation algorithm ($z = 0.299$, $p < .38$) or from the Albrecht-Shanahan perceptron ($z = 0.483$, $p < .6$). The performance of the Albrecht-Shanahan perceptron did not differ significantly from the Krumhansl-Schmuckler correlation algorithm ($z = -0.199$, $p < .39$).

Joshua Albrecht (personal communication) ran the Albrecht-Shanahan algorithm on a subset of 288 of the test stimuli, and reported that it identified the keys of classical genre stimuli with 97.1% accuracy, and generated 91% accuracy to the entire test set. Tests of proportions indicate that this performance on the entire test set is significantly better than that of the perceptron trained on the Krumhansl-Schmuckler profiles ($z = -9{,}829$, $p < .001$), as well as of the Krumhansl-Schmuckler correlation algorithm ($z = -2.319$, $p < .03$). However, this overall performance was not significantly better than either the perceptron trained on the Temperley profiles ($z = -1.523$, $p < .125$), or the perceptrons trained on the Albrecht-Shanahan profiles ($z = -1.401$, $p < .150$). With respect to performance on the 136 classical stimuli that Albrecht examined, the Albrecht-Shanahan model performs significantly better than the Krumhansl-Schmuckler perceptron ($z = -13.899$, $p < .001$), the Temperley perceptron ($z = -2.571$, $p < .01$), and the Krumhansl-Schmuckler correlation algorithm ($z = -2.293$, $p < .03$). However, it did not perform significantly better than the Albrecht-Shanahan perceptron on these stimuli ($z = -1.876$, $p <$ .07. It should be pointed out that Albrecht's test set did not include eight of the preludes from the second book of the *Well-Tempered Clavier*; only one of the perceptrons (the Albrecht-Shanahan perceptron) made a mistake in asserting the key for one of these omitted tests, and it only made a single mistake.

Table 4 provides a more detailed summary of performance of the key-finding methods, providing accuracy rates for each of the four different sets of test materials. For each set, accuracy is first given as the percent correct for the entire subset of stimuli; accuracy is then provided for only the major key stimuli and for only the minor key stimuli.

One observation to make from Table 4 is that perceptrons trained on tone profiles demonstrate different key-finding performance depending upon whether stimuli are in major or in minor keys. Interestingly, the perceptron that performs best on minor key stimuli is the one trained on the Albrecht-Shanahan profiles; for classical genre stimuli, it is 100% accurate. One of the motivations that Albrecht and Shanahan (2013) provided for their profiles was the goal of improving key-finding for minor key stimuli.

Another observation to make from Table 4 is that performance on classical genre stimuli is much better—for both perceptrons and for the correlation algorithm—than it is for the Nova Scotia folk songs. This may be due to a variety of factors. For instance, the folk songs are generally short univocal selections, while the classical pieces are generally longer and include harmony. As a result, there may be more reliable information about key in the classical selections than in the folk songs. Table 4 indicates that particular sets of tone profiles for key-finding might have more success for some genres, or for at least some subsets of stimuli, than for others.

At first glance, it appears that the algorithms that employ tone profiles derived from experimental data (i.e., the Krumhansl-Schmuckler tonal hierarchies) perform poorer than do the algorithms that employ the other two types of profiles, both of which are based on corpus data. This might suggest that the source from which tonal hierarchies are derived might determine its ability to assert key across musical genres. However, tests of proportions do not provide enough evidence to warrant this possibility. Tests of proportions were conducted on the overall accuracy of the various methods for the 152 folk song stimuli. This revealed, for example, that the Albrecht-Shanahan perceptron's performance on these stimuli was significantly better than was the performance of the Krumhansl-Schmuckler perceptron ($z = 2.060$, $p < .04$), but was not significantly better than the Krumhansl-Schmuckler correlation algorithm ($z = 1.897$, $p < .06$). There were no significant differences between the performance of the Temperley perceptron

Table 4
The Average Percent Accuracy of Classification of the Three Perceptrons Trained on Three Different Mean-Centered and Normalized Key Profiles

	Bach WTC			Chopin preludes			Hummel preludes			Nova Scotia folk songs		
Algorithm type	All	Major	Minor	All	Major	Minor	All	Major	Minor	All	Major	Minor
KSP	89.58	89.58	89.58	54.17	75.00	33.33	87.50	100.00	75.00	66.45	70.37	35.29
TP	95.83	97.92	93.75	95.83	91.67	100.00	91.67	91.67	91.67	70.39	78.52	5.88
ASP	96.88	93.75	100.00	83.33	66.67	100.00	100.00	100.00	100.00	74.34	79.26	35.29
KS Corr	93.75	87.50	100.0	87.50	83.33	91.67	100.0	100.0	100.0	67.11	71.85	29.41

Note. KSP indicates the perceptron trained on the Krumhansl-Schmuckler profiles; TP indicates the perceptron trained on the Temperley profiles, and ASP indicates the perceptron trained on the Albrecht-Shanahan profiles. The final row (KS Corr) provides the performance of the Krumhansl-Schmuckler correlation algorithm for purposes of comparison.

and the performance of the Krumhansl-Schmuckler perceptron ($z = 1.029$, $p < .24$), the Krumhansl-Schmuckler correlation algorithm ($z = 0.861$, $p < .28$), or the Albrecht-Shanahan perceptron ($z = -1.115$, $p < .21$).

One other observation to make, supported by the results in both Tables 3 and 4, is that when the same set of profiles is used, but processed with a different method, different performance is observed. To be more particular, the Krumhansl-Schmuckler perceptron generates significantly different levels of accuracy than does the Krumhansl-Schmuckler correlation algorithm, even though both of these methods use the same profiles. This shows that the perceptron is processing the profiles in a different fashion than is the case when correlation is used. The nature of this different type of processing is discussed in more detail below when the internal structure of perceptrons are discussed.

Network Errors

The preceding paragraphs have discussed the accuracy of perceptron responses. Another aspect of their performance to consider concerns the errors that they make. Any given musical key is more strongly related to some other musical keys than it is to others. The degree of this relationship can be quantified by counting the number of pitch-classes shared by the two keys in the set of seven pitch-classes that define the scale for each key, where for major keys the scale is taken to be the major, and for minor keys the scale is taken to be the harmonic minor. The key of C major, for instance, is most strongly related to both the keys of F major and G major, because its scale shares six pitch-classes with the scales of each of these keys; the maximum number of shared pitch-classes is seven, which only occurs when a key is compared with itself. The C major scale also shares six pitch-classes with the A harmonic minor scale; this strong relationship makes musical sense because A minor is called the relative minor of C major, because both keys have the same key signature. C major is slightly less related to its parallel minor, C minor, because their scales share only five pitch-classes. C minor is the parallel minor to C major because both keys are based on the same tonic (C), but they each have different key signatures. C major also shares five pitch-classes with A# major, D major, D minor and E minor. C major is least related to the keys of B major, C# major, and F# major; it shares only two pitch-classes with the scales of each of these keys. The average number of shared pitch-classes between a key and the 23 other different keys is 3.96 with a standard deviation of 1.26.

Table 5 provides the frequencies of different types of errors for each of the perceptrons, as well as for the Krumhansl-Schmuckler correlation algorithm, for the 296 test stimuli. In this table, error type is assigned in terms of the number of pitch-classes shared between the incorrect key asserted by the perceptron and the correct key of the stimulus. For instance, the first row of Table 5 indicates that the Krumhansl-Schmuckler perceptron generates three errors in which the incorrect key shares three pitch-classes with the correct key, generates 24 errors in which the incorrect key shares four pitch-classes with the correct key, and so on.

In general, Table 5 indicates that when the various perceptrons make mistakes, these mistakes involve asserting an incorrect key that is similar to the correct key. For instance, for the Krumhansl-Schmuckler perceptron and for the Albrecht-Shanahan perceptron, the most common error involves asserting a key that is very strongly related to the correct key, because the asserted key shares six pitch-classes with the desired key. In contrast, Table 5 also reveals that incorrect keys that share less than four pitch-classes with the correct key are very rarely asserted.

Table 5
Frequency of Error Types Made by the Different Key-Finding Algorithms, Where Errors Are Classified in Terms of the Number of Pitch-Classes Shared by the Correct Musical Key and the Incorrectly Asserted Musical Key

Algorithm type	Number of pitch-classes shared by incorrectly asserted musical key						
	2	3	4	5	6	Mean	*SD*
KSP	0	3	24	14	34	5.05	.43
TP	0	0	21	11	20	4.98	.40
ASP	0	0	16	10	20	5.09	.39
KS Corr	0	4	15	16	29	5.09	.43
Proportion	.13	.26	.26	.22	.13		

Note. KSP indicates the perceptron trained on the Krumhansl-Schmuckler profiles; TP indicates the perceptron trained on the Temperley profiles, and ASP indicates the perceptron trained on the Albrecht-Shanahan profiles. The final row (KS Corr) provides the performance of the Krumhansl-Schmuckler correlation algorithm for purposes of comparison. The final two columns provide the mean number of shared pitch-classes between the correct and the incorrectly asserted keys for each algorithm, as well as the standard deviation of the mean. The final row in the table provides the proportion of each type of error found when considering the 23 musical keys that differ from a selected musical key.

To statistically test whether the errors made by the perceptrons were systematic, χ^2 tests were performed. The final row of Table 5 provides the proportion of each type of error found in the set of 23 musical keys that are different from a selected key. These proportions were used to generate the expected frequency of each type of error for a particular algorithm by multiplying the total number of errors by the proportion. As there are five categories of errors, there were four degrees of freedom for each test. This comparison revealed that the Krumhansl-Schmuckler perceptron ($\chi^2 = 85.09$, $p < .001$), the Temperley perceptron ($\chi^2 = 50.19$, $p < .001$), the Albrecht-Shanahan perceptron ($\chi^2 = 52.00$, $p < .001$), and the Krumhansl-Schmuckler correlation algorithm ($\chi^2 = 69.58$, $p < .001$) all deviated significantly from this distribution. This implies that each of these algorithms is generating errors whose distribution differs significantly from that which would be obtained if an error were randomly sampled from the set of possible incorrect keys. In short, when all of the algorithms make errors, these errors tend to be musically related to the correct musical key.

Discrimination

One limitation with comparing the various algorithms using the test of proportions is that this only examines one (i.e., the highest) of the 24 responses made by the algorithm to an input stimulus. This is unsatisfactory because it ignores the other 23 responses of the algorithm made to each test stimulus. In addition to accuracy, another desirable feature of a key-finding algorithm is its ability to discriminate amongst competing keys. That is, a key-finding algorithm should generate a high response to the correct key, but should also generate much lower responses to all the other 23 incorrect keys. The greater the difference between response magnitudes associated with the correct key and those associated with the incorrect keys, the greater is the algorithm's ability to discriminate between possible keys.

The possibility that different algorithms might differ with respect to this type of discrimination is illustrated in Figure 2. This figure was created by examining the responses of two algorithms to the 57 different test stimuli that belonged to the key of F major. These stimuli were selected because this was the largest subset of test stimuli that all belonged to the same musical key. For each of these stimuli, the correlations between the stimulus' profile and each of the 24 Krumhansl-Schmuckler key profiles was calculated; the graph on the left of Figure 2 presents the mean correlation for each key profile, averaged over the 57 stimuli. Note that the heights of the bars in this graph appear to be highly variable; several of the bars (e.g., those for C major, F minor, C minor, D minor, and A minor) are of similar height to the bar associated with the correct key. The standard deviation of these 24 average correlations in the graph is 0.41.

For comparison, the graph on the right of Figure 2 presents the responses in the output units of the perceptron trained on the Krumhansl-Schmuckler profiles; these responses are averaged over the same 57 test stimuli. Note that this graph seems much less variable than the one plotted beside it; most of the average responses are near zero, a handful have weak positive values that are clearly smaller than the highest average response in the output unit that represents the correct key. The standard deviation of these bars is 0.12, about 3.4 times smaller than the standard deviation of the graph on the left. In short, the bar for the correct key F on the right in Figure 2 is much easier to discriminate from the other bars than is the case for the graph on the left in Figure 2. This suggests that the perceptron is a better discriminator between correct and incorrect keys than is the correlation algorithm, because the differences between the highest response and the remaining responses are greater for it than is the case for the correlation algorithm.

The analysis above uses a single musical key to illustrate higher discrimination in the perceptron. To statistically test the hypothesis that perceptrons are more discriminating than is the Krumhansl-Schmuckler algorithm across the range of musical keys, we proceeded as follows: First, we created a 24-entry vector for each of the 296 test stimuli, with each entry associated with a possible musical key. The entry associated with the correct key of each stimulus was given a value of 1; the other 23 entries in the vector were given a value of 0. Second, we computed the squared Pearson correlation between this stimulus vector and the 24 responses generated by each algorithm to each stimulus (i.e., the 24 output unit activities of each of the perceptrons, or the 24 correlations generated by the Krumhansl-Schmuckler algorithm). The higher the squared correlation between the stimulus vector and the response vector, the closer the match between the two. An algorithm that is more discriminating would be expected to generate significantly higher squared correlations to these test stimuli than would an algorithm that is less discriminatory. The results of this analysis are provided in Table 6.

An inspection of Table 6 indicates that on average the squared correlations computed for the Krumhansl-Schmuckler algorithm are markedly smaller than those for the three different types of perceptron. These three differences are all statistically significant; for instance, the comparison between the Krumhansl-Schmuckler correlation algorithm and the perceptron trained on the Krumhansl-Schmuckler profiles produces $t = 29.49$, $df = 295$, $p < .0001$. The squared correlations for the perceptron trained on

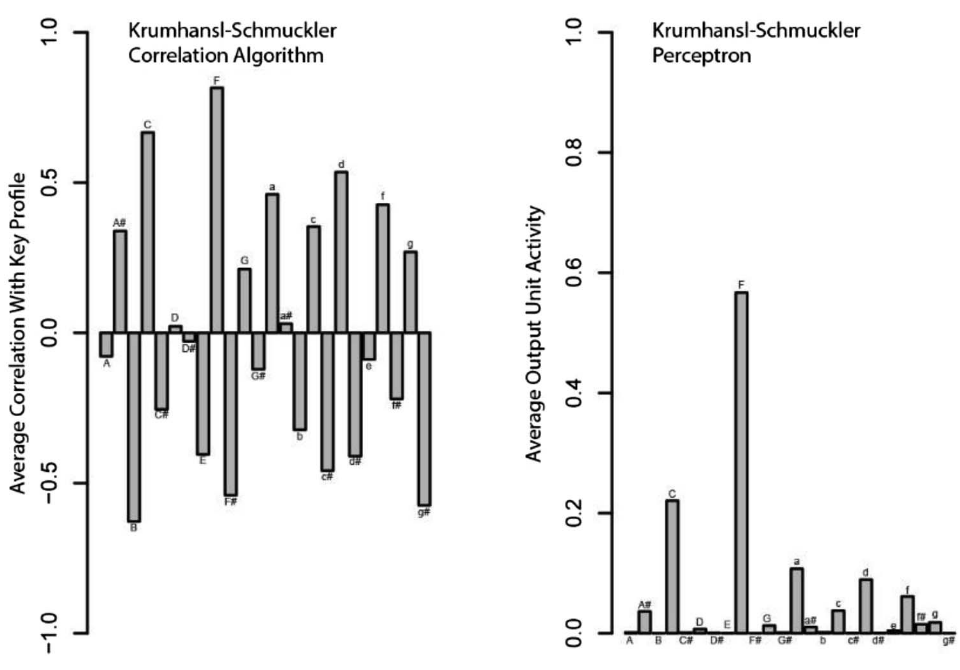

Figure 2. Average responses of the Krumhansl-Schmuckler correlation algorithm (left) and of the perceptron trained on the Krumhansl-Schmuckler profiles (right) when presented test stimuli from the key of F major. See text for details.

Table 6
The Average Squared Correlation (With Standard Deviations in Parentheses) Between the 24 Responses of a Key-Finding Algorithm and the 24-Unit Vector Encoding of the Desired Musical Key for the 296 Test Stimuli

Krumhansl-Schmuckler perceptron	Temperley perceptron	Albrecht-Shanahan perceptron	Krumhansl-Schmuckler algorithm
.64 (.32)	.73 (.34)	.77 (.32)	.15 (.05)

the Krumhansl-Schmuckler profiles are significantly smaller than those of the perceptron trained on the Temperley profiles ($t = -5.491$, $df = 295$, $p < .001$), as well as those for the perceptron trained on the Albrecht-Shanahan profiles ($t = -9.222$, $df = 295$, $p < .001$). The squared correlations for the perceptron trained on the Temperley profiles are significantly smaller than are those for the perceptron trained on the Albrecht-Shanahan profiles ($t = -3.678$, $df = 295$, $p < .001$).

It might be argued that encoding the desired vector with a 1 and with 0s puts the Krumhansl-Schmuckler algorithm at a disadvantage, because its output values can fall in principle into the range from −1 to 1, whereas the perceptron responses fall between 0 and 1. However, this is not the case. One can convert the vector of 1 and 0s into a vector of 1s and −1s with a linear transformation (multiply by 2 and subtract 1). This puts the desired vector into a range more consistent with the range of Krumhansl-Schmuckler correlations. However, because this transformation is linear it has no effect on the squared correlations that are computed, and does not change the results of Table 6.

Of course, one might wonder what advantage is offered by having greater discrimination. For example, the Krumhansl-Schmuckler perceptron has significantly greater discriminatory power than does the Krumhansl-Schmuckler correlation algorithm, but also has significantly poorer accuracy in asserting keys for the 296 test stimuli! However, it is important to realize that asserting a musical key is not the only task that is related to key-finding, as is noted later in the General Discussion. For instance, it might be of interest to determine whether two selections are in the same musical key, or to determine when a change in musical key has occurred. Such tasks involve comparing the musical keys of stimuli, but do not require absolute key assertion—for instance, one might easily say that two selections are in the same musical key, but not know what key they are in. Such tasks might plausibly benefit by comparing representations delivered by algorithms with higher discrimination, and the Table 6 results suggests that perceptrons could provide such a capability.

Network Structure

One of the advantages of using artificial neural networks as models is that they can provide new insights into the tasks that they have been trained to accomplish. These insights are discovered by examining the internal structure of networks after training has been completed (Berkeley, Dawson, Medler, Schopflocher, & Hornsby, 1995; Dawson, 2004, 2013; Dawson, Medler, McCaughan, Willson, & Carbonaro, 2000; Dawson & Piercey, 2001). Even though the perceptrons described above are very simple networks, analyses of their connection weights at the end of training raise some interesting issues about using key profiles for key-finding.

To begin, let us consider the structure of a perceptron trained on the mean-centered normalized Albrecht-Shanahan key profiles. An examination of weights of the connections between its input units and each of the output units that represents a major key reveals the same pattern of weights for each output unit when input units are coded not in terms of pitch-class, but instead in terms of degree within the key (i.e., by taking the input unit associated with the tonic as 0, the input unit associated with the tone a semitone higher than the tonic as 1, and so on). In other words, if inputs are considered in terms of degree away from the tonic (and not in terms of pitch class), then each major key output unit has nearly identical patterns of connection weights feeding into it. As is detailed below, a similar pattern is true for output units that represent minor keys, but the specific values of the weights for these units differ from those that feed into output units for major keys.

Furthermore, each of the 10 perceptrons that were trained converged on nearly the identical connection weight structure. We calculated the standard deviation of each connection weight across the 10 different perceptrons trained on these profiles (i.e., 288 different standard deviations were computed). On average, the standard deviation of a weight across the different networks was 0.04, while the highest standard deviation was 0.07 and the smallest standard deviation was 0.01. The high degree of similarity between the structures of the different perceptrons permitted an average perceptron to be created by averaging the biases and weights of the 10 trained networks.

Table 7 presents the details of the major key connection weights from this average perceptron. The first row of Table 7 provides the mean-centered normalized major key profile from the Albrecht-Shanahan model (also presented earlier in Table 2). The second row provides the average connection weight from each input unit (represented in terms of degree from tonic) to a major key output unit, along with the average bias of the output unit (which is analogous to the output unit's threshold). These structural components were averaged across all of the different instances of this type of perceptron. The third row shows how the stimulus profile is distorted when the perceptron multiplies each profile value by its associated weight.

If the perceptron did not need to transform the input key profiles when it was trained to map these profiles into musical keys, then each of its connection weights would equal 1. This is clearly not the case: the second row of Table 7 reveals that connection weights range from over 7 to about −5. This indicates that different components of stimulus profile have different degrees of importance in determining musical key.

To be more specific, when perceptrons use the logistic activation function in their output units, the perceptron is functionally equivalent to logistic regression, and its weights can be interpreted as being natural logarithms of odds ratios (Schumacher, Rossner, & Vach, 1996). This means that the more extreme the weight, the greater the likelihood that a signal from the input unit will affect

Table 7
Comparison of Albrecht-Shanahan Mean Centered Normalized Profiles to the Weight Structure of the Albrecht-Shanahan Perceptron for Both Major Key and Minor Key Profiles

		Degree of pitch-class in relation to key											
Source	Bias	0	1	2	3	4	5	6	7	8	9	10	11
Major profile		.57	−.29	.10	−.29	.20	.04	−.25	.49	−.28	−.01	−.28	−.01
Major weights	−8.28	7.29	.44	−.92	−5.04	4.83	−.76	−1.12	4.31	−2.43	−2.45	−3.78	−.28
Product		4.19	−.13	−.09	1.45	.96	−.03	.28	2.09	.67	.03	1.06	.00
Minor profile		.56	−.32	.09	.16	−.26	.08	−.29	.54	−.09	−.25	−.09	−.13
Minor weights	−7.94	5.70	−2.18	.48	4.10	−5.62	−.29	−.52	5.09	−.70	−4.13	−3.00	1.08
Product		3.21	.69	.04	.67	1.48	−.02	.15	2.74	.06	1.04	.27	−.14

Note. See text for details.

output unit activity. For the major weights in Table 6, this means that the above-average presence of pitch-classes associated with degrees of 0, 4, and 7 provide strong evidence that a stimulus belongs to the output unit's (major) key, because these profile components are associated with large positive weights. Similarly, the below-average presence of pitch-classes associated with degrees of 3, 8, 9, and 10 also provides strong evidence that a stimulus belongs to the output unit's key, because these profile components are associated with large negative weights. Importantly, inputs associated with degrees of 1, 2, 5, 6, or 11 provide much weaker evidence that an input stimulus belongs to the output unit's key, because these weights are much smaller than are the others in the table.

It is important to realize that these weightings of input unit signals are *in addition* to the different sized signals that are reflected in the profiles themselves. For instance, the mean-centered normalized Albrecht-Shanahan major key profile itself indicates that the most common component in the profile is degree 0, because its profile value is 0.57. However, the perceptron further amplifies this value by multiplying it by 7.29, indicating that this high profile value is itself extremely informative relative to the other profile values. Similarly, in the Albrecht-Shanahan major profile the values of degree 1 and degree 3 both equal −0.29. However, the perceptron weights these two values quite differently, multiplying the degree 1 signal by a weight of 0.44, while multiplying the degree 3 signal by a weight of −5.04. This indicates that the perceptron has learned that decreased occurrence of degree 3 pitch-classes is far more important for establishing the major key than is decreased occurrence of degree 1 pitch-classes.

Another way to consider the degree to which input profiles are transformed by perceptron weights is to correlate perceptron structure with the key profile. For Table 7, the correlation between the profile (first row) and the weights (second row) is 0.86. When the profile is modified by the weights via multiplication (third row), the correlation between the profile and the weighted profile is only 0.67. In short, the perceptron does not use the Albrecht-Shanahan profiles as given when it learns to map profiles to keys; it distorts the profiles with its weights because it learns that some components of the profile are more informative than are others. This distortion is reflected in the fact that these correlations are not equal to 1.

A similar result holds when connection weights between input units and output units that represent minor keys are examined for the Albrecht-Shanahan perceptron. Once again, the same pattern of weights is used by each minor key output unit when input signals are represented in terms of degree from tonic instead of pitch-class. The final three rows of Table 7 provide the profile, the average weights, and the weighted profile for a minor key output unit in this perceptron. Again, the fact that these weights are not all equal to 1 indicates that the perceptron transforms the input profiles by weighting their components differently. For minor keys, the correlation between the profile and the weights is 0.86, while the correlation between the profile and the weighted profile is 0.64.

With respect to minor keys, Table 7 indicates that the above-average presence of pitch-classes associated with degrees of 0, 3, and 7 provide strong evidence that a stimulus belongs to the output unit's (minor) key, because these degrees are associated with large positive weights. Similarly, below-average presence of pitch-classes associated with degrees of 2, 4, 9, and 10 also provide strong evidence that a stimulus belongs to the output unit's key, because these degrees are associated with large negative weights. Finally, the presence or absence of pitch-classes associated with degrees of 1, 5, 6, 8, or 11 provide much weaker evidence about an input stimulus' belonging to the output unit's key, because these weights are much smaller than are the others in the table.

The discussion of Table 7 in the preceding paragraphs indicates that not all components of a key profile are equally important. To explore this observation, we created a 'sparse profile' for each major and minor key. This was done by removing the five least important the 12 components of each of the 24 Albrecht-Shanahan profiles; these components were removed by setting their profile values equal to 0. For each major key profile, we removed the profile component from degrees 1, 2, 5, 6, and 11, because these five components were associated with the smallest weights in Table 6. For each minor key profile, we removed the profile component from degrees 2, 5, 6, 8, and 11; these components had the smallest weights for this profile in Table 7. We then trained 10 different perceptrons to assert musical key using these sparse profiles, employing the same training method described earlier. On average, they converged after 1008.5 sweeps of training. After training, we tested their performance using the same test stimuli and procedure as was described earlier.

Table 8 provides a summary of average performance of the perceptrons trained on the sparse Albrecht-Shanahan profiles when presented the various test stimuli. Overall, these perceptrons achieved 82% accuracy on the total test set, and 90% accuracy on the subset of the test set comprised of classical selections. For

Table 8
Average Performance (in Percent Correct) of Perceptrons Trained on Sparse Albrecht-Shanahan Profiles When Presented the Mean-Centered Normalized Profiles of the 296 Test Stimuli

Bach WTC			Chopin preludes			Hummel preludes			Nova Scotia folk songs		
All	Major	Minor	All	Major	Minor	All	Major	Minor	All	Major	Minor
89.58	79.17	100.00	83.33	66.67	100.00	95.83	100.00	91.67	75.59	79.04	48.24

overall accuracy, tests of proportions reveals that this performance is significantly poorer than the Albrecht-Shanahan perceptron ($z = -3.598$, $p < .001$), significantly poorer than the Krumhansl-Schmuckler correlation algorithm ($z = -2.611$, $p < .01$), and significantly poorer than the Albrecht-Shanahan algorithm ($z = -5.86$, $p < .001$). Given that this perceptron was trained on profiles that had about 40% of their components removed, this poorer performance is not surprising.

What is surprising is that removing this much of each profile used in training did *not* result in *poorer* performance. An examination of Table 8 indicates that it is slightly poorer on both the Bach and Hummel stimuli, equal in performance on the Chopin stimuli, and slightly better on the Nova Scotia folk songs when compared to the Albrecht-Shanahan perceptron results in Table 4. In short, perceptron performance is only about 8% poorer than the Albrecht-Shanahan perceptrons from Table 7, even though 40% of the profiles were set to zero prior to training.

To this point, we have considered in detail the internal structure of the perceptron trained on the Albrecht-Shanahan profiles. Very similar accounts could be provided for the connection weights in the other two types of perceptrons as well. Table 9 presents the weights between a major key output unit and the 12 input units, with input units coded in terms of degree instead of pitch-class, for each of the three types of perceptron. Each column of weights in the perceptron is averaged across the 10 different perceptrons of that type that were trained. Table 8 also arranges the weights in order from the most positive to the most negative.

The Albrecht-Shanahan perceptron weights in Table 9 have already been discussed in detail. Table 9 indicates that the other two types of perceptrons have similar patterns for their major key weights. The weights are highly variable, indicating that some profile components are more important for asserting major keys than are others. Clearly all of the perceptrons discovered the need to transform input profiles when they learn to use them to assert keys.

Table 9 indicates some interesting differences between perceptron types. In particular, the three columns indicating the source of input signals are not in the same order. This indicates that each type of perceptron does not assign the same degree of importance to each source of input. For instance, the most important source of evidence for the Temperley perceptron is degree 4, while for the other two perceptrons it is from degree 0.

Table 10 provides the same kind of information as Table 9, but in this case, for an output unit that represents a minor key. It, like Table 9, indicates that each type of perceptron learns that it has to transform stimulus profiles in order to use them to assert keys, as shown in the variety of weights in each column. The different order in each of the source columns indicates that each type of perceptron assigns different levels of importance to the various degrees of an input profile.

Table 9
Weights Between Inputs Units (Coded in Terms of Degree) and a Major Scale Output Unit for Three Different Types of Perceptron

Albrecht-Shanahan perceptron		Temperley perceptron		Krumhansl-Schmuckler perceptron	
Source	Weight	Source	Weight	Source	Weight
0	7.27	4	5.67	0	6.68
4	4.84	7	4.52	4	3.57
7	4.33	0	3.67	7	3.44
1	.48	5	1.77	5	1.57
11	−.32	2	1.41	6	.39
5	−.78	1	−.76	1	−.63
2	−.95	11	−.77	8	−1.60
6	−1.13	6	−2.57	2	−1.63
9	−2.44	8	−2.57	9	−2.07
8	−2.46	3	−2.78	11	−2.39
10	−3.77	9	−3.63	10	−3.45
3	−5.04	10	−3.92	3	−3.78
Bias	−8.28	Bias	−7.47	Bias	−7.15

Note. See text for details.

Table 10
Weights Between Inputs Units (Coded in Terms of Degree) and a Minor Scale Output Unit for Three Different Types of Perceptron

Albrecht-Shanahan perceptron		Temperley perceptron		Krumhansl-Schmuckler perceptron	
Source	Weight	Source	Weight	Source	Weight
0	5.73	3	3.30	0	6.46
7	5.05	0	3.13	3	4.61
3	4.08	7	2.76	11	1.28
11	1.13	2	1.80	2	.41
2	.46	11	1.74	7	.21
5	−.32	5	1.65	1	.11
6	−.59	8	.60	6	−.67
8	−.70	6	−.50	5	−1.69
1	−2.20	1	−1.71	10	−2.22
10	−2.97	9	−3.03	9	−2.30
9	−4.06	4	−4.14	8	−2.84
4	−5.66	10	−5.57	4	−3.35
Bias	−7.94	Bias	−6.14	Bias	−6.05

Note. See text for details.

With the logistic activation function, perceptron outputs can literally be interpreted as conditional probabilities (Bishop, 1995, 2006; Dawson & Dupuis, 2012; Dawson, Dupuis, Spetch, & Kelly, 2009; Dawson & Gupta, 2017; Dupuis & Dawson, 2013a; Hastie, Tibshirani, & Friedman, 2009). In our simulations, this is the probability that a particular input unit represents the correct musical key given the presence of a particular input pattern. Furthermore, with the logistic activation function the weights of a perceptron can be interpreted as being identical to the coefficients of logistic regression and are therefore equivalent to the natural logarithms of odds ratios (Hosmer & Lemeshow, 2000; Schumacher et al., 1996). The implication of this is straightforward: each connection weight in a trained perceptron is an effect size, where the effect is attributable to the presence of a certain value of an independent variable (i.e., an input value), and the size of the effect reflects the degree to which this independent variable alters the state of the output unit.

Why do perceptrons learn to differentiate incoming signals in terms of their effectiveness at signaling key? It is important to remember that when learning begins, each training stimulus is presented to the same set of input units (see Figure 1), so all of these input units are activated both when the profile associated with a particular output unit's key and when the other 23 incorrect profiles are presented. As a result, the perceptron must discover a set of weights that cause a specific output unit to turn on when its profile is presented. However, it must also use these same weights to turn the output unit off when any of the other profiles are presented. The gradient descent learning rule discovers an optimal set of weights for accomplishing both of these goals, in the sense that the connection weights at the end of training minimize network response errors. However, one property of this configuration is that not all input signals are given the same weights.

Discussion

Summary

The purpose of the current paper was to explore the suitability of using very simple artificial neural networks, perceptrons, to assert musical key. Perceptrons were trained to map mean-centered normalized tone profiles to specific musical keys. Three different sets of tone profiles (Albrecht & Shanahan, 2013; Krumhansl, 1990; Temperley, 2007) were used to train three different types of perceptron. One main result was that when performance of these perceptrons was tested on a set of 296 novel stimuli, two types of perceptron—those trained on the Temperley profiles and those trained on the Albrecht-Shanahan profiles—generated overall accuracy that was not statistically different from two standard algorithms (Albrecht & Shanahan, 2013; Krumhansl, 1990). Interestingly, perceptrons trained on the Krumhansl-Schmuckler profiles were significantly less accurate than were any of the other algorithms.

A second interesting finding was that all three different types of perceptrons exhibit greater discriminatory power than does the Krumhansl-Schmuckler correlation algorithm. For all of these algorithms, the goal is to have a value associated with the desired key (i.e., an output unit activity or a correlation) achieve a greater magnitude than the same value obtained for the remaining 23 incorrect keys. Our analyses revealed that all three types of perceptrons generate, on average, significantly higher differences between the correct value and the remaining incorrect values than is the case for the correlation algorithm.

A third result of the simulation studies was that when a perceptron is trained on one set of tone profiles, it can generate different responses than does a correlation algorithm which employs the same profiles. In particular, perceptrons trained on the Krumhansl-Schmuckler profiles perform significantly poorer in terms of accuracy, and significantly better in terms of discrimination, than does the Krumhansl-Schmuckler correlation algorithm. One reason for this is that the logistic activation function used in a perceptron's output units implements a nonlinear transformation of an input signal that is not captured by correlation.

However, a more important reason for the difference in performance is that the connection weights of all three types of perceptrons dramatically distort input signals. One of the key discoveries revealed by our simulations is that when tone profiles are used to train a key-finding perceptron, it learns that not all components of the profile are equally important with respect to signaling key. The perceptrons adjust their weights to modify the input signals to emphasize the information provided by important profile elements, and to de-emphasize the information provided by less important elements. The discovery of these distortions is an example of the contributions that these networks can make to key-finding algorithms as well as to the experimental psychology of musical cognition.

Contributions to Key-Finding Algorithms

That perceptrons do not weight all components of a tonal hierarchy equally illustrates how the structure of a trained perceptron can suggest possible modifications to other key-finding algorithms that do not employ artificial neural networks.

For example, the Albrecht-Shanahan algorithm positions each tone profile as a point in a 12-dimensional space, positions a stimulus profile in the same space, and assigns the stimulus the key that is represented by the point to which the stimulus is closest to (Albrecht & Shanahan, 2013). However, this model assumes that each dimension in the space has the same importance. The Albrecht-Shanahan perceptron interpretation (see Table 6) indicates that the space in which the Albrecht and Shanahan algorithm measures distances could be distorted, with some dimensions being stretched out (i.e., those associated with important profile components), and with others being shortened in size (i.e., those associated with less important profile components). The weights of the perceptron provide magnitudes for distorting the Albrecht-Shanahan space before points are plotted and distances are measured; it would be interesting to see what effect such distortions would have on the algorithm's performance. A similar approach could be taken to incorporate weights into other algorithms that are based on tone profiles by using them to scale profile components. Of course, the weights in Table 6 would likely not be the precise ones to use, because they arise from a network trained on mean-centered normalized tone profiles. However, a perceptron trained to map musical keys to tone profiles that have not been preprocessed (i.e., those presented earlier in Table 1) would provide weights that could be incorporated into another algorithm.

The preceding discussion is an example of how the perceptrons described in the current paper can contribute to key-finding algorithms. Importantly, the neural network approach taken here is flexible enough to be used to conduct different simulations that can provide other contributions.

For instance, the three sets of tone profiles (Albrecht & Shanahan, 2013; Krumhansl, 1990; Temperley, 1999, 2007) used in the current simulations are derived from different origins, are typically employed using different algorithms, and involve different magnitudes of values. Given these differences, it is perhaps not surprising that no one has attempted to improve key-finding by combining all three sets of profiles together in a single algorithm. However, perceptrons can easily be used to perform key-finding judgments using combinations of profiles. A single perceptron could learn all three sets of tone profiles, and then use this combined learning for key-finding. Perhaps this network could use its combined knowledge to more accurately deal with stimuli that provide problems to networks that only have knowledge of a single type of profile. Related to this approach, one could use all three different perceptrons from the simulations reported above and pool their responses to assert musical key. Such an architecture is called a committee of networks; committees of networks have been shown to be superior to single networks in a variety of pattern classification tasks (Buus et al., 2003; Das, Reddy, & Narayanan, 2001; Guo & Luh, 2004; Marwala, 2000; Medler & Dawson, 1994; Zhao, Huang, & Sun, 2004).

As a second example, perceptrons could be trained to use tone profiles to make judgments key-finding judgments that are different from the key assertions made by the current networks and by traditional algorithms. For instance, a network could be presented pairs of tone profiles and learn to use a single output unit to indicate whether the two profiles were from the same musical key. The network could then be used to judge whether pairs of stimuli belonged to the same musical key.

As a third example, one clear result from the simulation study is that algorithm performance on the Nova Scotia folk songs is poorer than is its performance on classical stimuli. One approach to improving this situation would be to investigate whether perceptron performance improved when different tone profiles, based on different sources, were used as training stimuli. One approach to conducting this research would be to train perceptrons on stimuli derived from representations of musical selections (like the various test stimuli discussed earlier) instead of on tone profiles taken from existing algorithms. This approach could be used to identify the kinds of stimuli that are better suited for making key judgments about folk songs, and an interpretation of such networks could lead to the discovery of new kinds of tone profiles.

As a fourth example, perceptrons can easily be used to explore the impact of alternative representations of inputs or of responses. The simulations reported above employed 24 output units to employ a response representation that could be directly compared with that used by existing key-finding algorithms. However, networks are flexible enough to explore alternative output representations; it is unclear whether this is possible for other current key-finding algorithms. For instance, a network could use 13 output units – 12 to represent the tonic of a musical key (A, A#, and so on) and an additional unit to represent whether the key is major (by turning this unit on) or not (by turning this unit off). Note that this network is functionally equivalent to those described earlier, in the sense that it learns the same task. The only difference is with respect how the key-finding assertion is represented.

Pilot studies have shown that though a 13-output learns to assert key tonics, but is not capable of learning to assert the correct key mode for every tone profile. This result is similar to other results that have examined training networks to identify properties of musical scales (Dawson, in press): a perceptron can identify the tonic of a scale, but cannot identify its mode. To identify mode, a more complex network that employs hidden units is required. The pilot studies on 13-output unit perceptrons indicate that it is a simpler information processing task to retrieve a key's tonic from a tone profile than it is to identify a key's mode.

Contributions to Musical Cognition

One advantage of the simulation approach used in the current paper is that simulations can be used to explore a variety of different manipulations and tasks. The purpose of this exploration is to seek interesting predictions that can then be pursued experimentally using subjects. Some of our previous research using perceptrons has demonstrated this paradigm. For instance, after using simulations and formal analyses to establish strong general links between perceptrons and associative learning (Dawson, 2008), and to then use these networks to model phenomena in a particular associative learning task, reorientation (Dawson et al., 2010), perceptrons generated interesting hypotheses concerning reorientation which were then explored and confirmed experimentally (Dupuis & Dawson, 2013b).

The current paper has not explored the experimental predictions of the simulations, because its primary concern was whether these networks could serve as plausible key-finding algorithms. However, the demonstrated success of the current networks at asserting musical key suggests that they too can serve to generate new research questions to be evaluated via further study with human subjects. Three such questions immediately arise from the simulation results that have been reported above.

First, three different key-finding theories were used as the sources of the tone profiles that were used to train our perceptrons (Albrecht & Shanahan, 2013; Krumhansl, 1990; Temperley, 2007). However, these theories differ not only in their specific tone profiles, but also in their method of processing tone profiles to assert musical key. To our knowledge, the current paper is the first example in which these different profiles were used to assert keys by employing a common method. This permits the efficacy of the different profiles themselves to be compared. Our results demonstrated that perceptron performance on the test stimuli depended upon which tone profiles were used (see Tables 3 and 4 above, for instance). One empirical question that this result raises is whether one of these profile pairs provides a better account of human performance on various stimuli than do the others.

Second, perceptron performance varied significantly depending upon the type of test stimulus that was used. In particular, perceptrons were much better at asserting the musical key of novel classical genre stimuli than they were at asserting the key of the Nova Scotia folk songs. Do human listeners display a similar pattern of results?

Third, one of the main results of the current simulations was that some components of a tone profile are more informative than are others when a perceptron uses these components to assert musical key. Is this true of human listeners as well?

These three experimental questions, all generated from the performance of our perceptrons, could be explored using a variation of the probe tone method. For instance, Schmuckler and Tomovski (2005) investigated whether one characteristic of the Krumhansl-Schmuckler algorithm—better performance on Bach preludes than on Chopin preludes—was also true of human listeners. They tested this possibility by using various short segments taken from the two different types of preludes as contexts in the probe tone paradigm, and discovered that their listeners also performed less ably when presented contexts from Chopin. This sort of variation of the probe tone methodology could easily be adapted to explore the three research questions that were introduced above.

Importantly, it would be very easy to modify our simulation methodology to train perceptrons on tasks that are related to key-finding, but which do not necessarily involve the direct assertion of musical key. One example was discussed earlier, where in exploring alternative encodings of perceptron responses, we separate judgments of a key's mode from judgments of a key's tonic. As discussed earlier, pilot simulations indicate that tone profiles can be used by a perceptron to detect a key's tonic, but identifying whether a profile represents a major or a minor mode poses problems for perceptrons. Exploring this issue with human experiments would be interesting. Similarly, it is very straightforward to present pairs of tone profiles to a network, and have it use a single output unit to judge whether the two profiles are the same or different with respect to key, to tonic, or to mode. Of interest in this simulation would be determining whether a network has more trouble learning to distinguish certain pairs of stimuli. Of course, of further interest would be testing human subjects to see whether they have similar difficulties in distinguishing certain types of stimuli.

Both of the avenues of research suggested in the previous paragraph would have to be experimentally explored by adopting a methodology that begins to depart from the probe tone method. For instance, one new methodology would involve presenting listeners two stimuli in sequence and then making a same/different judgment with respect to a musical attribute of interest (key, tonic, or mode). From this perspective, changing the task that a network learns to perform in turn leads to developing novel experimental methodologies that can be used to compare human judgments to those of the networks.

Résumé

Nous explorons ici la capacité d'un très simple réseau neural artificiel, un perceptron, à affirmer la clé musicale de nouveaux stimuli. Premièrement, les perceptrons sont formés pour associer des profils de clés standardisés (prélevés parmi une à trois différentes sources) avec différentes clés musicales. Une fois la formation achevée, nous avons mesuré l'exactitude avec laquelle les perceptrons affirmaient les clés musicales pour 296 nouveaux stimuli. Selon les profils clés utilisés pendant la formation, les perceptrons peuvent produire les mêmes résultats que les algorithmes de sélection de clés lors de cette tâche. Des analyses plus poussées indiquent que les perceptrons génèrent plus d'activité dans une unité qui représente une clé sélectionnée et beaucoup moins d'activité dans les unités qui représentent les clés concurrentes qui n'ont pas été sélectionnées, comparativement à un algorithme traditionnel. Finalement, nous avons examiné la structure interne des perceptrons formés et découvert qu'ils, contrairement aux algorithmes traditionnels, attribuaient de très différents poids aux différentes composantes d'un profil-clé. Les perceptrons apprennent que certaines composantes de profil sont plus importantes dans la spécification de clés musicales que d'autres. Ces poids différentiels pourraient être incorporés dans des algorithmes traditionnels qui eux-mêmes n'emploient pas de réseaux neuraux artificiels.

Mots-clés : réseaux neuraux artificiels, sélection de clé, perceptrons.

References

Adiloglu, K., & Alpaslan, F. N. (2007). A machine learning approach to two-voice counterpoint composition. *Knowledge-Based Systems, 20,* 300–309. http://dx.doi.org/10.1016/j.knosys.2006.04.018

Albrecht, J. D., & Shanahan, D. (2013). The use of large corpora to train a new type of key-finding algorithm: An improved treatment of the minor mode. *Music Perception, 31,* 59–67. http://dx.doi.org/10.1525/mp.2013.31.1.59

Bellgard, M. I., & Tsang, C. P. (1994). Harmonizing music the Boltzmann way. *Connection Science, 6,* 281–297. http://dx.doi.org/10.1080/09540099408915727

Benuskova, L. (1995). Modeling transpositional invariancy of melody recognition with an attractor neural network. *Network (Bristol, England), 6,* 313–331. http://dx.doi.org/10.1088/0954-898X_6_3_001

Berkeley, I. S. N., Dawson, M. R. W., Medler, D. A., Schopflocher, D. P., & Hornsby, L. (1995). Density plots of hidden value unit activations reveal interpretable bands. *Connection Science, 7,* 167–186. http://dx.doi.org/10.1080/09540099550039336

Berkeley, I., & Raine, R. (2011). An old-fashioned connectionist approach to a Cajun chord change problem. *Connection Science, 23,* 209–218. http://dx.doi.org/10.1080/09540091.2011.597500

Bharucha, J. J. (1987). Music cognition and perceptual facilitation: A connectionist framework. *Music Perception, 5,* 1–30. http://dx.doi.org/10.2307/40285384

Bharucha, J. J. (1999). Neural nets, temporal composites, and tonality. In D. Deutsch (Ed.), *The Psychology Of Music* (2nd ed., pp. 413–440). San Diego, CA: Academic Press. http://dx.doi.org/10.1016/B978-012213564-4/50012-3

Bharucha, J. J., & Todd, P. M. (1989). Modeling the perception of tonal structure with neural nets. *Computer Music Journal, 13,* 44–53. http://dx.doi.org/10.2307/3679552

Bishop, C. M. (1995). *Neural networks for pattern recognition*. Oxford, UK: Oxford University Press.

Bishop, C. M. (2006). *Pattern recognition and machine learning*. New York, NY: Springer.

Butler, D. (1989). Describing the perception of tonality in music: A critique of the tonal hierarchy theory and a proposal for a theory of itervallic rivalry. *Music Perception, 6,* 219–241. http://dx.doi.org/10.2307/40285588

Buus, S., Lauemøller, S. L., Worning, P., Kesmir, C., Frimurer, T., Corbet, S., . . . Brunak, S. (2003). Sensitive quantitative predictions of peptide-MHC binding by a 'Query by Committee' artificial neural network approach. *Tissue Antigens, 62,* 378–384. http://dx.doi.org/10.1034/j.1399-0039.2003.00112.x

Cangelosi, A. (2010). Connectionist modelling of music emotions. *Physics of Life Reviews, 7,* 37–38. http://dx.doi.org/10.1016/j.plrev.2010.01.005

Coutinho, E., & Cangelosi, A. (2009). The use of spatio-temporal connectionist models in psychological studies of musical emotions. *Music Perception, 27,* 1–15. http://dx.doi.org/10.1525/mp.2009.27.1.1

Creighton, H. (1932). *Songs and ballads from Nova Scotia*. Toronto, Canada: J. M. Dent.

Das, A., Reddy, N. P., & Narayanan, J. (2001). Hybrid fuzzy logic committee neural networks for recognition of swallow acceleration signals. *Computer Methods and Programs in Biomedicine, 64,* 87–99. http://dx.doi.org/10.1016/S0169-2607(00)00099-7

Dawson, M. R. W. (2004). *Minds and machines: Connectionism and psychological modeling*. Malden, MA: Blackwell. http://dx.doi.org/10.1002/9780470752999

Dawson, M. R. W. (2005). *Connectionism: A hands-on approach* (1st ed.). Malden, MA: Blackwell. http://dx.doi.org/10.1002/9780470694077

Dawson, M. R. W. (2008). Connectionism and classical conditioning. *Comparative Cognition and Behavior Reviews, 3* (Monograph), 1–115.

Dawson, M. R. W. (2013). *Mind, body, world: Foundations Of cognitive science*. Edmonton, Canada: Athabasca University Press.

Dawson, M. R. W. (in press). *Connectionist Representations of Tonal Music: Discovering Musical Patterns by Interpreting Artificial Neural Networks*. Edmonton, Canada: Athabasca University Press.

Dawson, M. R. W., & Dupuis, B. (2012). Equilibria of perceptrons for simple contingency problems. *IEEE Transactions on Neural Networks and Learning Systems, 23,* 1340–1344. http://dx.doi.org/10.1109/TNNLS.2012.2199766

Dawson, M. R. W., Dupuis, B., Spetch, M. L., & Kelly, D. M. (2009). Simple artificial neural networks that match probability and exploit and explore when confronting a multiarmed bandit. *IEEE Transactions on Neural Networks, 20,* 1368–1371. http://dx.doi.org/10.1109/TNN.2009.2025588

Dawson, M. R. W., & Gupta, M. (2017). Probability matching in perceptrons: Effects of conditional dependence and linear nonseparability. *PLoS ONE, 12,* e0172431. http://dx.doi.org/10.1371/journal.pone.0172431

Dawson, M. R. W., Kelly, D. M., Spetch, M. L., & Dupuis, B. (2008). Learning about environmental geometry: A flaw in Miller and Shettleworth's (2007). operant model. *Journal of Experimental Psychology: Animal Behavior Processes, 34,* 415–418. http://dx.doi.org/10.1037/0097-7403.34.3.415

Dawson, M. R. W., Kelly, D. M., Spetch, M. L., & Dupuis, B. (2010). Using perceptrons to explore the reorientation task. *Cognition, 114,* 207–226. http://dx.doi.org/10.1016/j.cognition.2009.09.006

Dawson, M. R. W., Medler, D. A., McCaughan, D. B., Willson, L., & Carbonaro, M. (2000). Using extra output learning to insert a symbolic theory into a connectionist network. *Minds and Machines, 10,* 171–201. http://dx.doi.org/10.1023/A:1008313828824

Dawson, M. R. W., & Piercey, C. D. (2001). On the subsymbolic nature of a PDP architecture that uses a nonmonotonic activation function. *Minds and Machines, 11,* 197–218. http://dx.doi.org/10.1023/A:1011237306312

Desain, P., & Honing, H. (1989). The quantization of musical time: A connectionist approach. *Computer Music Journal, 13,* 56–66. http://dx.doi.org/10.2307/3680012

Dupuis, B., & Dawson, M. R. W. (2013a). Differentiating models of associative learning: Reorientation, superconditioning, and the role of inhibition. *Journal of Experimental Psychology: Animal Behavior Processes, 39,* 273–286. http://dx.doi.org/10.1037/a0032174

Dupuis, B., & Dawson, M. R. W. (2013b). Get out of the corner: Inhibition and the effect of location type and number on perceptron and human reorientation. *Learning & Behavior, 41,* 360–378. http://dx.doi.org/10.3758/s13420-013-0111-0

Fiske, H. E. (2004). *Connectionist models of musical thinking*. Lewiston, NY: E. Mellen Press.

Frankland, B. W., & Cohen, A. J. (1996). Using the Krumhansl and Schmuckler key-finding algorithm to quantify the effects of tonality in the interpolated-tone pitch-comparison task. *Music Perception, 14,* 57–83. http://dx.doi.org/10.2307/40285709

Franklin, J. A. (2006). Jazz melody generation using recurrent networks and reinforcement learning. *International Journal of Artificial Intelligence Tools, 15,* 623–650. http://dx.doi.org/10.1142/S0218213006002849

Gasser, M., Eck, D., & Port, R. (1999). Meter as mechanism: A neural network model that learns metrical patterns. *Connection Science, 11,* 187–216. http://dx.doi.org/10.1080/095400999116331

Gjerdingen, R. O. (1990). Categorization of musical patterns by self-organizing neuron-like networks. *Music Perception, 7,* 339–369. http://dx.doi.org/10.2307/40285472

Gjerdingen, R. O. (1992). Learning syntactically significant temporal patterns of chords: A masking field embedded in an ART-3 architecture. *Neural Networks, 5,* 551–564. http://dx.doi.org/10.1016/S0893-6080(05)80034-6

Gjerdingen, R. O., & Perrott, D. (2008). Scanning the dial: The rapid recognition of music genres. *Journal of New Music Research, 37,* 93–100. http://dx.doi.org/10.1080/09298210802479268

Gluck, M. A., & Bower, G. H. (1988). From conditioning to category learning: An adaptive network model. *Journal of Experimental Psychology: General, 117,* 227–247. http://dx.doi.org/10.1037/0096-3445.117.3.227

Griffith, N. (1995). Connectionist visualization of tonal structure. *Artificial Intelligence Review, 8,* 393–408. http://dx.doi.org/10.1007/BF00849727

Griffith, N., & Todd, P. M. (1999). *Musical networks: Parallel distributed perception and performance*. Cambridge, MA: MIT Press.

Guo, J. J., & Luh, P. B. (2004). Improving market clearing price prediction by using a committee machine of neural networks. *IEEE Transactions on Power Systems, 19,* 1867–1876. http://dx.doi.org/10.1109/TPWRS.2004.837759

Handelman, E. J., & Sigler, A. (2013). Key induction and key mapping using pitch-class set assertions. In J. Yust, J. Wild, & J. A. Burgoyne (Eds.), *Mathematics and computation in music* (pp. 115–127). New York, NY: Springer. http://dx.doi.org/10.1007/978-3-642-39357-0_9

Hastie, T., Tibshirani, R., & Friedman, J. H. (2009). *The elements of statistical learning* (2nd ed.). New York, NY: Springer. http://dx.doi.org/10.1007/978-0-387-84858-7

Hays, W. L. (1981). *Statistics* (3rd ed.). New York, NY: Holt, Rinehart and Winston.

Holtzman, S. R. (1977). Program for key determination. *Interface: Journal of new music research, 6,* 29–56. http://dx.doi.org/10.1080/09298217708570231

Hoover, A. K., & Stanley, K. O. (2009). Exploiting functional relationships in musical composition. *Connection Science, 21,* 227–251. http://dx.doi.org/10.1080/09540090902733871

Hosmer, D. W., & Lemeshow, S. (2000). *Applied logistic regression* (2nd ed.). New York, NY: Wiley. http://dx.doi.org/10.1002/0471722146

Huron, D. (1999). *Music research using humdrum: A user's guide*. Stanford, CA: Center for Computer Assisted Research in the Humanities.

Katz, B. F. (1995). Harmonic resolution, neural resonance, and positive affect. *Music Perception, 13,* 79–108. http://dx.doi.org/10.2307/40285686

Krumhansl, C. L. (1979). Psychological representation of musical pitch in a tonal context. *Cognitive Psychology, 11,* 346–374. http://dx.doi.org/10.1016/0010-0285(79)90016-1

Krumhansl, C. L. (1990). *Cognitive foundations of musical pitch*. New York, NY: Oxford University Press.

Krumhansl, C. L., & Shepard, R. N. (1979). Quantification of the hierarchy of tonal functions within a diatonic context. *Journal of Experimental Psychology: Human Perception and Performance, 5,* 579–594. http://dx.doi.org/10.1037/0096-1523.5.4.579

Laden, B., & Keefe, B. H. (1989). The representation of pitch in a neural net model of pitch classification. *Computer Music Journal, 13,* 12–26. http://dx.doi.org/10.2307/3679550

Large, E. W., Kim, J. C., Flaig, N. K., Bharucha, J. J., & Krumhansl, C. L. (2016). A neurodynamic account of musical tonality. *Music Perception, 33,* 319–331. http://dx.doi.org/10.1525/mp.2016.33.3.319

Large, E. W., & Kolen, J. F. (1994). Resonance and the perception of musical meter. *Connection Science, 6,* 177–208. http://dx.doi.org/10.1080/09540099408915723

Leman, M. (1991). The ontogenesis of tonal semantics: Results of a computer study. In P. M. Todd & D. G. Loy (Eds.), *Music and connectionism* (pp. 100–127). Cambridge, MA: MIT Press.

Longuet-Higgins, H. C., & Steedman, M. J. (1971). On interpreting Bach. In B. Meltzer & D. Michie (Eds.), *Machine intelligence* (Vol. pmed6, pp. 221–239). Edinburgh, UK: Edinburgh University Press.

Loy, D. G. (1991). Connectionism and musiconomy. In P. M. Todd & D. G. Loy (Eds.), *Music and connectionism* (pp. 20–36). Cambridge, MA: MIT Press.

Marwala, T. (2000). Damage identification using committee of neural networks. *Journal of Engineering Mechanics, 126,* 43–50. http://dx.doi.org/10.1061/(ASCE)0733-9399(2000)126:1(43)

McClelland, J. L., & Rumelhart, D. E. (1986). *Parallel distributed processing* (Vol. 2). Cambridge, MA: MIT Press.

Medler, D. A., & Dawson, M. R. W. (1994). Training redundant artificial neural networks: Imposing biology on technology. *Psychological Research, 57,* 54–62. http://dx.doi.org/10.1007/BF00452996

Monterola, C., Abundo, C., Tugaff, J., & Venturina, L. E. (2009). Prediction of potential hit song and musical genre using artificial neural networks. *International Journal of Modern Physics C, 20,* 1697–1718. http://dx.doi.org/10.1142/S0129183109014680

Mostafa, M. M., & Billor, N. (2009). Recognition of Western style musical genres using machine learning techniques. *Expert Systems with Applications, 36,* 11378–11389. http://dx.doi.org/10.1016/j.eswa.2009.03.050

Mozer, M. C. (1991). Connectionist music composition based on melodic, stylistic, and psychophysical constraints. In P. M. Todd & D. G. Loy (Eds.), *Music and connectionism* (pp. 195–211). Cambridge, MA: MIT Press.

Mozer, M. C. (1994). Neural network music composition by prediction: Exploring the benefits of psychoacoustic constraints and multi-scale processing. *Connection Science, 6,* 247–280. http://dx.doi.org/10.1080/09540099408915726

Nagashima, T., & Kawashima, J. (1997). Experimental study on arranging music by chaotic neural network. *International Journal of Intelligent Systems, 12,* 323–339. http://dx.doi.org/10.1002/(SICI)1098-111X(199704)12:4<323::AID-INT5>3.0.CO;2-Q

Page, M. P. A. (1994). Modeling the perception of musical sequences with self-organizing neural networks. *Connection Science, 6,* 223–246. http://dx.doi.org/10.1080/09540099408915725

Rescorla, R. A., & Wagner, A. R. (1972). A theory of Pavlovian conditioning: Variations in the effectiveness of reinforcement and nonreinforcement. In A. H. Black & W. F. Prokasy (Eds.), *Classical conditioning II: Current research and theory* (pp. 64–99). New York, NY: Appleton-Century-Crofts.

Rosenblatt, F. (1958). The perceptron: A probabilistic model for information storage and organization in the brain. *Psychological Review, 65,* 386–408. http://dx.doi.org/10.1037/h0042519

Rosenblatt, F. (1962). *Principles of neurodynamics*. Washington, DC: Spartan Books.

Rumelhart, D. E., Hinton, G. E., & Williams, R. J. (1986). Learning representations by back-propagating errors. *Nature, 323,* 533–536. http://dx.doi.org/10.1038/323533a0

Rumelhart, D. E., & McClelland, J. L. (1986). *Parallel distributed processing* (Vol. 1). Cambridge, MA: MIT Press.

Sapp, C. S. (2005). Visual hierarchical key analysis. *ACM Computers In Entertainment, 3,* 1–19. http://dx.doi.org/10.1145/1095534.1095544

Scarborough, D. L., Miller, B. O., & Jones, J. A. (1989). Connectionist models for tonal analysis. *Computer Music Journal, 13,* 49–55. http://dx.doi.org/10.2307/3680011

Schmuckler, M. A., & Tomovski, R. (2005). Perceptual tests of an algorithm for musical key-finding. *Journal of Experimental Psychology: Human Perception and Performance, 31,* 1124–1149. http://dx.doi.org/10.1037/0096-1523.31.5.1124

Schumacher, M., Rossner, R., & Vach, W. (1996). Neural networks and logistic regression: Part 1. *Computational Statistics & Data Analysis, 21,* 661–682. http://dx.doi.org/10.1016/0167-9473(95)00032-1

Shibata, N. (1991). A neural network-based method for chord note scale association with melodies. *NEC Research & Development, 32,* 453–459.

Shmulevich, I., & Yli-Harja, O. (2000). Localized key finding: Algorithms and applications. *Music Perception, 17,* 531–544. http://dx.doi.org/10.2307/40285832

Stevens, C., & Latimer, C. (1992). A comparison of connectionist models of music recognition and human performance. *Minds and Machines, 2,* 379–400. http://dx.doi.org/10.1007/BF00419420

Sutton, R. S., & Barto, A. G. (1981). Toward a modern theory of adaptive networks: Expectation and prediction. *Psychological Review, 88,* 135–170. http://dx.doi.org/10.1037/0033-295X.88.2.135

Temperley, D. (1999). What's key for key? The Krumhansl-Schmuckler key-finding algorithm reconsidered. *Music Perception, 17,* 65–100. http://dx.doi.org/10.2307/40285812

Temperley, D. (2004). Bayesian models of musical structure and cognition. *Musicae Scientiae, 8,* 175–205. http://dx.doi.org/10.1177/102986490400800204

Temperley, D. (2007). *Music and probability*. Cambridge, MA: MIT Press.

Temperley, D., & Marvin, E. W. (2008). Pitch-class distribution and the identification of key. *Music Perception, 25,* 193–212. http://dx.doi.org/10.1525/mp.2008.25.3.193

Tillmann, B., Bharucha, J. J., & Bigand, E. (2000). Implicit learning of tonality: A self-organizing approach. *Psychological Review, 107,* 885–913. http://dx.doi.org/10.1037/0033-295X.107.4.885

Todd, P. M. (1989). A connectionist approach to algorithmic composition. *Computer Music Journal, 13,* 27–43. http://dx.doi.org/10.2307/3679551

Todd, P. M., & Loy, D. G. (1991). *Music And connectionism*. Cambridge, MA: MIT Press.

Toivainen, P. (2005). Visualization of tonal content with self-organizing maps and self-similarity matrices. *ACM Computers in Entertainment, 3,* 1–10.

Toiviainen, P., & Krumhansl, C. L. (2003). Measuring and modeling real-time responses to music: The dynamics of tonality induction. *Perception, 32,* 741–766. http://dx.doi.org/10.1068/p3312

Vos, P. G., & Van Geenen, E. W. (1996). A parallel-processing key-finding model. *Music Perception, 14,* 185–223. http://dx.doi.org/10.2307/40285717

Yaremchuk, V., & Dawson, M. R. W. (2005). Chord classifications by artificial neural networks revisited: Internal representations of circles of major thirds and minor thirds. *Artificial Neural Networks: Biological Inspirations - Icann 2005, 3696,* 605–610.

Yaremchuk, V., & Dawson, M. R. W. (2008). Artificial neural networks that classify musical chords. *International Journal of Cognitive Informatics and Natural Intelligence, 2,* 22–30. http://dx.doi.org/10.4018/jcini.2008070102

Zhao, Z. Q., Huang, D. S., & Sun, B. Y. (2004). Human face recognition based on multi-features using neural networks committee. *Pattern Recognition Letters, 25,* 1351–1358. http://dx.doi.org/10.1016/j.patrec.2004.05.008

Received January 7, 2017
Accepted April 3, 2017 ■