Behavior Research Methods
2009, 41 (1), 154-162
doi:10.3758/BRM.41.1.154

ChoiceKey: A real-time speech recognition program for psychology experiments with a small response set

Christopher Donkin, Scott D. Brown, and Andrew Heathcote
University of Newcastle, Newcastle, Australia

Psychological experiments often collect choice responses using buttonpresses. However, spoken responses are useful in many cases—for example, when working with special clinical populations, or when a paradigm demands vocalization, or when accurate response time measurements are desired. In these cases, spoken responses are typically collected using a voice key, which usually involves manual coding by experimenters in a tedious and error-prone manner. We describe ChoiceKey, an open-source speech recognition package for MATLAB. It can be optimized by training for small response sets and different speakers. We show ChoiceKey to be reliable with minimal training for most participants in experiments with two different responses. Problems presented by individual differences, and occasional atypical responses, are examined, and extensions to larger response sets are explored. The ChoiceKey source files and instructions may be downloaded as supplemental materials for this article from brm.psychonomic-journals.org/content/supplemental.

Many psychology experiments require participants to complete hundreds of trials using a small response set. For example, memory experiments often require participants to respond only "old" or "new" (e.g., Rubin, Hinton, & Wenzel, 1999), and choice response time (RT) tasks often require participants to make one of two responses, such as "one" and "two," or "high" and "low" (e.g., Ratcliff & Rouder, 1998). The typical means of collecting responses for this type of experiment is a buttonpress, usually via a keyboard, mouse, specially developed buttonbox, or a touch screen. However, there are a number of reasons that an experimenter might instead prefer to collect spoken responses. For these cases we offer an open source speech recognition package, ChoiceKey. In the following we show that ChoiceKey reliably identifies a small number of response alternatives and that it gives precise estimates of vocal RT; but first, we discuss a few of the circumstances in which one might prefer vocal responses to alternative data collection methods.

RT measurement is an important aspect of many psychology experiments, and the precision and accuracy of RT estimates from different response tools has been documented extensively in this journal. Keyboard responses are often imprecise due to buffering issues (Plant, Hammond, & Turner, 2004; Shimizu, 2002; Voss, Leonhart, & Stahl, 2007), as are mouse-button clicks (Beringer, 1992; Crosbie, 1990; Plant, Hammond, & Whitehouse, 2003). Precise RT measurements can be obtained using buttonboxes connected via the PC parallel port (e.g., Stewart, 2006; Voss et al., 2007), but these solutions require specialized hardware, which can be expensive and is not always well supported. We show that ChoiceKey also yields precise measurements of RT, with the advantages of being simple to set up and inexpensive, requiring only a microphone and a sound card, standard equipment for most PCs.

Aside from RT measurements, making responses via buttons can be problematic because it requires the participant to learn a response-to-button mapping. Although some of these mappings are relatively natural, such as "left" and "right" using the left and right arrows of the keyboard, other response sets have no intuitive button mapping. For example, Rubin et al. (1999) mapped the responses "old" and "new" to keys chosen by the experimenters, and participants had to learn this mapping and maintain it throughout the experiment. If participants were able to speak aloud the responses "old" or "new," the learning of this mapping could be avoided.

Experimental research with clinical populations unable to give manual responses via a buttonpress might also benefit from the ability to easily collect spoken responses and the associated RTs. In particular, an automatic speech recognition program might benefit experimenters working with people with schizophrenia, people with intellectual disabilities, or people with psychomotor disabilities. Trewin and Pain (1999) have shown that people with these types of psychological and/or motor disabilities display a wide range of errors when using a mouse or keyboard. The use of spoken responses might help avoid some of these measurement errors.

C. Donkin, chris.donkin@newcastle.edu.au

Even if participants are able to give manual responses, Vidulich and Wickens (1985) show that spoken responses are most appropriate when central processing is required for a verbally oriented task. It is also possible that the need for spoken responses is implicit, given the experimental procedure or paradigm being used, as in Stroop-like tasks that investigate the cause of interference due to response modality (Simon & Sudalaimuthu, 1979; Wang & Proctor, 1996). In these situations, ChoiceKey offers a way of both identifying response and recording RT.

Among others, Lacouture and Marley (2004) allowed participants to respond vocally in an absolute identification experiment. Participants gave spoken responses via microphone, and RTs were obtained using a "voice key," a device that measures RT in terms of the time taken for sound energy to cross a threshold. However, a voice key requires response choices to be manually coded by an experimenter, an inevitably time-consuming, tedious, and error-prone task. Speech recognition software can help alleviate these problems.

Speech recognition software is in common use; most people have had the experience of placing an order, or giving personal details, over the phone. However, the accuracy of these systems can be far from perfect, and is unlikely to be acceptable for experimental measurement. Microsoft Windows and Macintosh OS X both come with inbuilt speech recognition functionality that can be adapted by training to individual users' voices. However, we found these inbuilt speech recognition packages to be far too inaccurate for use in experiments, even under ideal conditions with only two different responses and extensive training. This is likely because the programs are intended to recognize a very large number of different responses in environments where the cost of an incorrect recognition event is low.

As an alternative, we offer an exemplar-based speech recognition program designed exclusively to recognize only those responses that are to be used in an experiment. The program, ChoiceKey, is an open-source library for the software package MATLAB that can be called by a MATLAB script that controls the experiment. Appendix A outlines an example script for a simple experiment in which one of two stimuli is presented and the participant is required to name it. ChoiceKey was developed using the Data Collection and Voicebox toolboxes under MATLAB v7.5.0 (R2007b) and Reynolds, Quatieri, and Dunn's (2000) Gaussian mixture models for speaker identification. Details about the contents of the ChoiceKey library are outlined in Appendix B.

The underlying structure of ChoiceKey is based on leading models of speaker verification. Bimbot et al. (2004) offered a detailed and complete discussion of the extensive work in this area. A graphical summary of how ChoiceKey works is presented in Figure 1. Sound card outputs are captured using the inbuilt MATLAB Data Collection Toolbox. Audio capture begins when input from a microphone reaches a threshold, and terminates 1.5 sec later. Both the threshold and recording time can be altered by the user. The recorded data are first passed through front-end processing, transforming the time-varying amplitude input from the microphone into a set of features represented as a vector of "cepstral" coefficients. These feature vectors are then modeled statistically to create a set of training exemplar models. Later, during the experiment, participants' responses are turned into feature vectors, and the likelihood that these vectors came from each training exemplar model is calculated. The most likely model is the chosen response. We now discuss each of these aspects in more detail.

Front-End Processing

In the front-end processing stage, the sound input is broken up into 20-msec windows using a 10-msec frame rate, ensuring 50% overlap between segments. Only those windows containing enough sound energy to be considered not silent are kept. This is done relative to the noise in the signal, so as to lower the probability that speech is discarded. The Mel scale cepstral feature vectors are then calculated for each of the 20-msec windows. This is done by first taking the fast Fourier transform of the speech segment. The resulting spectrum is smoothed using a series of bandpass frequency filters which are convolved with the spectrum to get an average value for each frequency band. These filters are spaced on the Mel scale, which has the property of being close to the frequency scale of the human ear (Stevens, Volkmann, & Newman, 1937). A discrete cosine transformation is applied to the log of the values produced by the frequency filters to yield cepstral coefficients.

Reynolds et al. (2000) suggested that all but the 0th cepstral coefficient are best used in speaker recognition. However, for ChoiceKey we desire speech, not speaker, recognition, and we have achieved greater accuracy by retaining the 0th coefficient. Reynolds et al. also suggested that a number of normalization transformations be made to compensate for mismatched microphone conditions between training and testing. ChoiceKey does not use any such normalizing transformations, since we assume that training and testing will be done under identical conditions.

Statistical Modeling

The cepstral coefficients are modeled using Gaussian mixture models (GMMs), which have been shown to be successful in the domain of speaker recognition (Reynolds, 1992). GMMs have the desirable properties of being able to capture the behavior of a distribution without assuming a very specific (e.g., Gaussian) form. They are also computationally simple, facilitating real-time processing. A GMM's density is the weighted linear combination of M Gaussian densities, each parameterized by a mean and variance term for each cepstral coefficient vector. The number of Gaussian densities used, M, can be altered by the user. During development, we found that five Gaussian densities gave the best overall performance. However, individual differences in the optimal value of M did exist, so improved individual accuracy may be obtained by setting M based on an individual's data.

The Decision

During ChoiceKey training, the participant will speak aloud each of k response words N times. Each of these

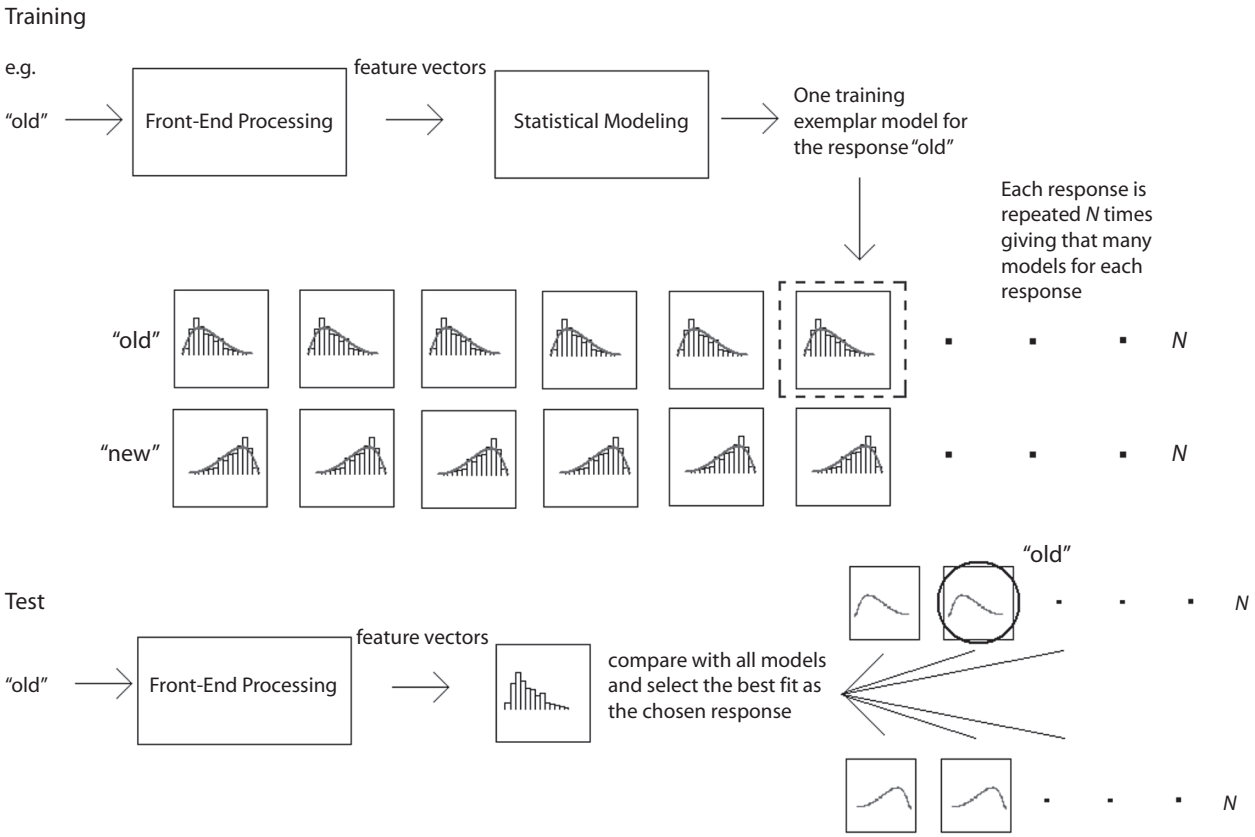

Figure 1. Graphical representation of ChoiceKey's operation. During training, the response "old" is given on a particular trial. Features are extracted and a statistical model is created for that training exemplar. The model is then stored with the rest of the models created in training. During testing, the participant speaks the word "old" during one of the trials of the experiment. The features of the word are extracted and then compared with all the models created at training. The most likely model is chosen, which happens to be one of the trained exemplars for the response "old," so ChoiceKey chooses that response.

training words is modeled using a GMM, giving *N* exemplar models for each of the *k* responses at the end of training. On any particular trial of the experiment proper, the participant will make a new and unknown response. The Mel scale cepstral feature vectors are calculated for this new response. ChoiceKey then calculates the log likelihood of observing the cepstral feature vectors, given the parameters of the GMM for each of the *N* exemplars and *k* responses. The exemplar with the largest log likelihood is selected as the given response.

During development, we tried a range of alternative response selection rules, such as selecting the response set with the largest summed log likelihood across all exemplars, and more sophisticated classifiers such as backpropagation neural networks and support vector machines. For the small response-set sizes in our experiments, the more sophisticated selection rules did not provide any benefit, but this may not be the case for larger response-set sizes. The simple selection rule used by ChoiceKey has the advantage of reduced computational cost, particularly during training. In other settings, ChoiceKey can be easily adapted by the user to implement alternative training and decision algorithms.

Using ChoiceKey

A typical experiment using ChoiceKey involves a short training session (less than 5 min), where participants speak aloud the words in the response set (e.g., old,new) a number of times (typically between 10 and 30), training ChoiceKey to identify their voices. The experiment then proceeds as normal, with responses made by voice, using ChoiceKey to return the response that it calculates to be the one most likely spoken by the participant, and the RT. We now report the results of experiments examining the accuracy of these measurements. The first experiment investigates the accuracy of the RT measured by ChoiceKey, by comparing it with RTs manually measured from audio waveforms recorded in real time. In the second experiment, we investigate how accurately ChoiceKey identifies responses from a variety of response sets.

EXPERIMENTS

Method

RT. An AMD computer, with an AthlonXP 64-bit 2.33-GHz processor and 2 GB of RAM running Windows XP SP2 with a SoundBlaster Live! v5.10 sound card, was set up to play a loud tone

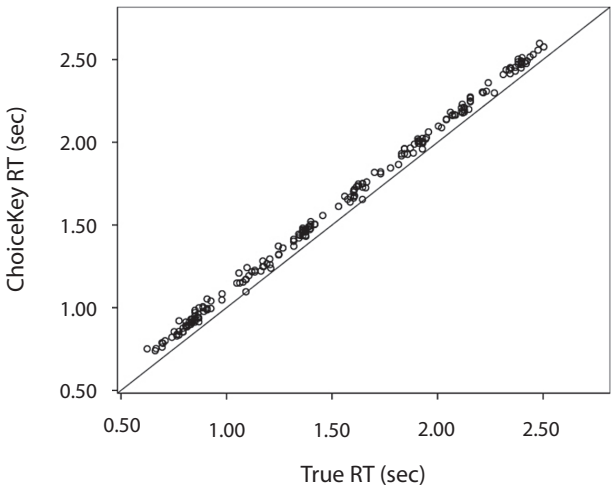

Figure 2. Response time (RT) as recorded by ChoiceKey (*y*-axis) as a function of RT calculated manually from sound waveforms (*x*-axis). The solid diagonal line represents perfect measurement of RT. ChoiceKey gives a precise, but slightly biased, estimate of RT.

through external speakers. After the tone played, data capture within ChoiceKey was initialized. After an interval, the word *respond* appeared on screen, and the participant said "one" into a headset-mounted Sony DR-220 microphone. The speaking of the word "one" was intended to trigger recording, and for all 200 trials the triggering worked as required. The intervals between the tone and response were varied from 250 to 2,000 msec in intervals of 250 msec, with each interval occurring 25 times, yielding 200 RTs that spanned the range of RTs usually observed in simple psychological tasks.

Throughout the course of the experiment, a second laptop was set up nearby with its external microphone making a real-time recording of the entire proceedings under Adobe CS3 Soundbooth. This recording was later opened in Soundbooth as a waveform and the time between the tone and the "one" response was determined manually. This process gave us an accurate estimate of RT that should correlate highly with ChoiceKey's RT measurement.

Figure 2 shows the RTs recorded by ChoiceKey plotted against those derived from the waveform. The two measures of RT were highly correlated ($r = .999$). RTs from ChoiceKey were used as the response variable in a linear regression, with true RT used as the predictor variable. The slope of the regression line was 0.999 [$t(199) = 344.83$, $p < .001$], and the intercept was 90 msec [$t(199) = 18.875$, $p < .001$], suggesting that ChoiceKey gives a precise, but slightly biased, estimate of RT.

Rastle and Davis (2002) discussed biases in voice-key RT measurement as a function of the onset characteristics of different response waveforms. If experimenters are concerned about obtaining absolutely unbiased measurements of RT, the above procedure can be carried out for all responses separately. More likely to be of concern to users of ChoiceKey, however, are differences between biases in RT measurement for different responses. Unless this issue is addressed, differences in RT may be attributed to differences between stimuli, when the real cause is differences in the time taken for ChoiceKey to trigger the onset of recording.

To address this issue without the time and effort required by manual scoring of waveforms, ChoiceKey includes a function called *callib*. On each trial, this function presents participants with a "+" sign and asks them to make one response repeatedly throughout a block of trials of a duration determined by the experimenter. The process is then repeated for each response. The function returns the mean RT for each response. Any differences in RT due to onset biases for the different responses can then be identified and corrected.

Identification. Identification accuracy was investigated using data from 24 participants who read aloud the following three sets: {1,2,3,4}, {old,new}, and {high,low}. Participants were 12 male and 12 female first-year psychology students. Words in each set were spoken 50 times in a random order and the order of each set was counterbalanced across subjects. Responses were collected using the same hardware used in the previous experiment. As envisaged for a standard experiment, the first 10 responses spoken by participants were used to train ChoiceKey. The remaining 40 responses were used to test ChoiceKey's identification accuracy. Results from the response set consisting of numbers 1 to 4 were partitioned into six different response sets of size 2. These sets, along with {old,new} and {high,low}, were used to test ChoiceKey's two-choice identification accuracy. Table 1 shows the distributions of the number of errors out of the 80 identifications made by ChoiceKey.

Very few identification errors were observed for the majority of participants and responses. For most response sets, ChoiceKey made 0 errors, or 1, out of 80 for almost all participants. For the response set {old,new}, ChoiceKey made no errors in identifying responses for 16 out of 24 participants. The response sets {2,4} and {1,2} were also identified with very few errors, with either 0 errors or 1 error being made for 20 and 19 participants, respectively. The average accuracy of ChoiceKey's identification was highest for the response sets {old,new}, followed closely by {2,4} and {3,4}. Individual differences in identification far outweighed any differences observed as a function of age or gender.

The results of Table 1 indicate that, with only 10 training exemplars, ChoiceKey is able to perform well for some response sets, but that others are less discriminable. For example, in the response sets {high,low} and {2,3}, ChoiceKey was able to identify all responses correctly for only a quarter of the participants. Not only were some response sets less discriminable, but even for the responses sets in which ChoiceKey *was* almost perfectly accurate for the majority of participants, a small proportion of participants remained whose responses were difficult to discriminate. For example, the response set {1,2} leads to either one or no errors for 19 out of 24 participants, suggesting it as a good candidate for use with ChoiceKey. However, for 1 participant, 35 responses out of 80 were identified incorrectly. This suggests that ChoiceKey is not viable for some participants with minimal training.

One approach to solving this problem is to first screen participants based on a preexperimental test of ChoiceKey's accuracy. The function *traintest* provides an estimate of ChoiceKey's identification accuracy. Interestingly, for the participant with very low accuracy for the response set {1,2}, all other combinations of number response accuracy were also low. However, no errors in identification were observed for the response set {high,low} for this participant, suggesting that speaker identification accuracy varies substantially as a function of response set.

An alternative approach to dealing with low identification accuracy, for either particular participants or particular response sets, is

Table 1
Number of Errors (Out of 80 Identifications) Made by ChoiceKey for Individual Participants for Each Response Set

Response Set	0	1	2	3	>3	*M**
old,new	16	3	3	2	0	.99
2,4	15	5	1	1	2	.98
1,2	14	5	0	3	2	.96
1,3	11	6	1	4	2	.97
3,4	12	4	4	3	1	.98
1,4	9	2	2	9	2	.95
high,low	6	4	5	8	1	.96
2,3	6	2	4	7	5	.92

Note—The first cell indicates that for 16 out of 24 participants there were zero errors in identification for the word set {old,new}. *Mean proportion of correct identifications for each response set.

to use more than the ten training exemplars. To evaluate this strategy, 12 of the 24 participants completed an extra 50 responses for the word sets {old,new} and {high,low}. These extra responses were used to test the effect of varying the number of responses per word used in training ChoiceKey. We varied the number of responses, or exemplars, used to train ChoiceKey from 1 to 30. In each of these tests the final 70 responses made by participants were used to test ChoiceKey's identification accuracy.

Figure 3 shows the average percentage of accurate identifications made by ChoiceKey as a function of the number of training exemplars. For the response set {old,new}, increasing the number of training exemplars beyond 10 did not increase accuracy. This is likely due to a ceiling effect, since discrimination between the words was already close to perfect with 10 training exemplars. For the response set {high,low}, improvement was less rapid, but the same accuracy as for {old,new} was achieved with a set of 30 exemplars, suggesting that even difficult-to-discriminate word sets can be used with ChoiceKey, as long as sufficient training is provided.

Increasing the number of training exemplars improved accuracy for all participants, even those who had very low accuracy with fewer training exemplars. Figure 4A compares individual participant accuracy for the response set {high,low} with 10 and 30 training exemplars. Participants are ordered along the *x*-axis by accuracy for the case with 10 training exemplars, and the same order is used for the case with 30 exemplars to highlight individual improvement. For 23 out of 24 participants, accuracy either increased or remained perfect with the increase in training set size. Participants whose accuracies were lowest with 10 training exemplars showed the largest increase, bringing performance for almost all participants up to acceptable levels.

With only 10 training exemplars, accuracy was much worse for response sets of more than two words. Figure 4B shows accuracy for individual participants for the response sets {1,2,3} and {1,2,3,4} when ChoiceKey was trained with 10 exemplars. Participants are ordered by accuracy for the smaller response set on the *x*-axis. Average accuracy was roughly equivalent for both response sets, although some participants were noticeably less accurate for the larger response set. Only a quarter of the participants were more than 95% accurate, suggesting that a larger training set is required for the remaining participants.

These results suggest that experimenters who wish to use ChoiceKey for response sets larger than two should use the *traintest* function to screen participants and/or calibrate training set size to achieve the desired level of accuracy. The latter strategy should be implemented with caution, however, since we did not test whether larger set sizes display the same improvement in accuracy with training set size that we found with set size two.

A second limitation related to the use of larger set sizes is the associated computational cost, which increases as a polynomial function of both the response set size and the number of training exemplars. Sufficient time must be available between collecting the training data and commencing the experiment to process the training data. For example, when response set size was increased from two to four, the time taken to identify a single response, given 10 training exemplars, increased from 50 to 100 msec. When 40 training exemplars were used, however, identifying a single response took 100 and 300 msec for two and four responses, respectively.

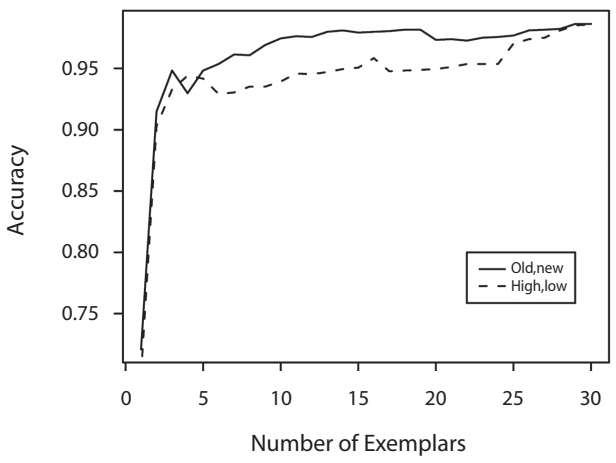

Figure 3. ChoiceKey identification accuracy averaged over participants and plotted as a function of number exemplars used in training. Accuracy was high for the response set {old,new} even when few training exemplars were used, but more exemplars were required for acceptable performance with the {high,low} response set.

RESULTS AND DISCUSSION

ChoiceKey is a measurement tool for the MATLAB environment that allows the collection of vocal choice responses. It is designed for use in experiments with a small number of possible responses. ChoiceKey provides precise estimates of vocal onset time, and can be easily calibrated to eliminate onset differences between responses (Rastle & Davis, 2002). For a variety of common response pairs, it can reliably identify most participants' responses with high accuracy after only minimal training, such as 10 training exemplars per response, which takes only around 2 min for a binary choice task.

However, we found that some response pairs are identified with lower accuracy than others (e.g., {high,low}). Although experimenters could simply use response pairs that are reliably identified with high accuracy (e.g., {old,new}), doing so removes the potential benefit of reduced response learning offered by spoken responses. An alternative strategy is to use a larger training set, which improves accuracy. For example, with 30 exemplars accuracy was equally good for {old,new} and {high,low}.

A second issue is that identification accuracy is low for some participants, suggesting that there may be a need for pretest screening of ChoiceKey's accuracy for each participant. This problem could be addressed through the use of more training exemplars, at least for the participants in our experiment. Increasing the number of responses used to train ChoiceKey from 10 to 30 not only increased identification accuracy for one of the response sets with the poorest performance, {high,low}, it also improved identification accuracy for those participants whose responses were most poorly identified when only 10 training exemplars were used.

Even after extended training, ChoiceKey did not perfectly identify all responses from all participants. These errors appeared to be asymptotic (i.e., they did not disappear with increased training). Such asymptotic errors are likely due to atypical responses, background noise (in the environment or in computer hardware), or both. We minimized the latter source of error by using a quiet testing environment and a high-quality sound card and microphone. However, it is likely that even when background noise and hardware errors are minimized, participants will sometimes say words in a way that was not encountered in training, caus-

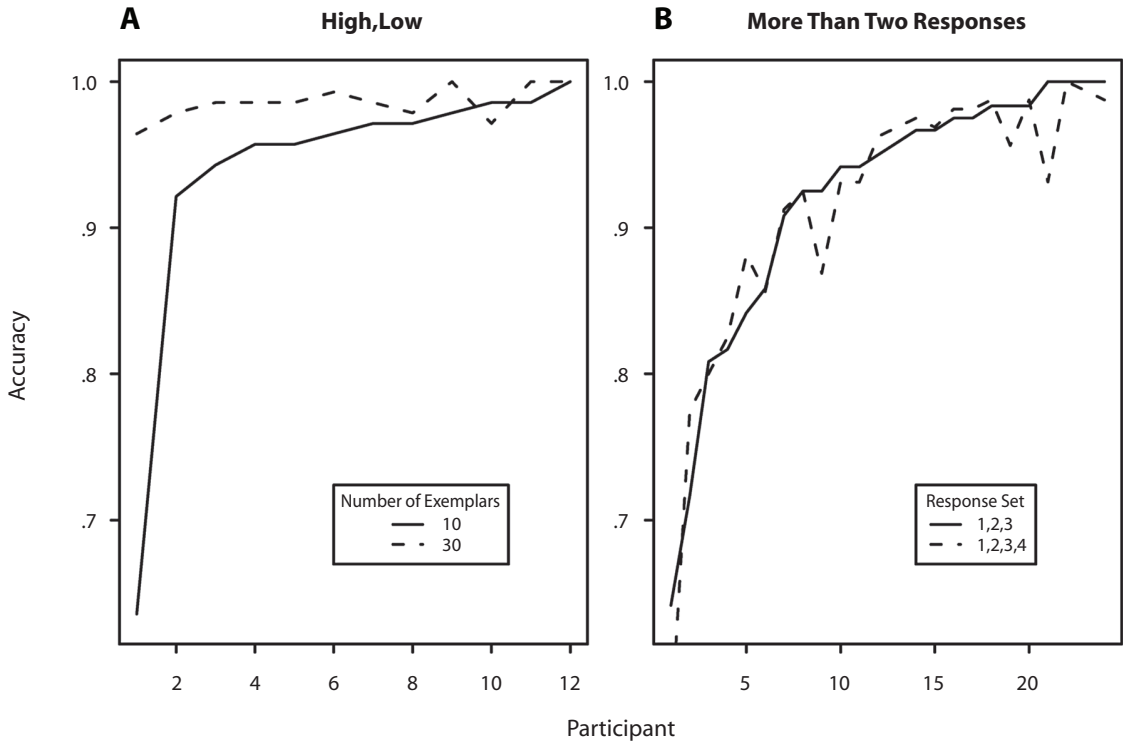

Figure 4. ChoiceKey recognition accuracy for individual participants for the response set {high,low} when 10 and 30 exemplars were used in training, and when the response sets {1,2,3} or {1,2,3,4} were used with 10 training exemplars. Participants are ordered according to accuracy in the 10 exemplar and {1,2,3} cases for (A) and (B), respectively.

ing misidentification. Fortunately, the proportion of such asymptotic errors in testing was low (around 1%).

It is arguable that this low error rate may not be too different from the rate of errors caused by participants pressing the wrong response button, and it may even be less than the buttonpress error rate when the response mapping is unfamiliar or insufficiently practiced. Similarly, ChoiceKey's low error rate may be comparable to errors made by the experimenter manually coding responses in real time. Where perfect vocal choice identification is required, we recommend that responses be recorded and scored offline. Even where an error rate of 1%–2% is acceptable, it may be prudent to record some responses and perform an offline check of ChoiceKey's scoring as a quality control measure.

Apart from lowering background noise and using high-quality hardware, we were not able to identify any other measures that reliably increased ChoiceKey's accuracy. For example, we were unable to identify an obvious reason why certain word pairs show lower discriminability than others. We therefore advise users that they choose the most natural response set for the task, and if necessary, increase the number of training exemplars until the desired level of identification accuracy is achieved. Similarly, there appears to be no clear pattern to the type of voice that ChoiceKey is able to identify with high accuracy (e.g., male vs. female voices). We suggest the same course of action: Pick the most natural methodology, and if pretest screening shows any individual participant with a high error rate, either use more training exemplars or exclude that participant's results from analysis.

Vocal responses are particularly advantageous with larger response sets, where button responding is naturally more error prone, due to the greater difficulty of learning a larger response mapping. Large response sets are also more likely to introduce differences in RT due to differences in response production time (e.g., differences between fingers). When more than 10 responses are required, individual fingers cannot be assigned to each response, requiring finger combinations (further increasing learning difficulty) or a movement response (e.g., moving a finger or mouse cursor from a "home" button to a response button). In both cases, RT variability and the potential for differences in production time go up.

Unfortunately, we found that creating a speech recognition system highly accurate for large response sets is very difficult. When the response set was extended beyond two alternatives, we observed a large drop in accuracy, to about 90% on average for three or four different responses. Individual differences existed, and we found that a majority of participants had low accuracy when ChoiceKey was trained with only 10 exemplars per response. These results indicate that (1) ChoiceKey should be used cautiously in the event of more than two response alternatives, and (2) extended training will likely be required to obtain high accuracy. Fortunately, because ChoiceKey is open source and implemented in the flexible MATLAB language, users may easily explore such extensions.

Some directions for these future extensions have already been discussed. For example, it may be possible to optimize certain parameters affecting identification accuracy individually for each participant (e.g., the number of Gaussian density mixtures to use in the GMM). Future improvement may also lie in an alternative form of statistical modeling of the features of the recorded speech segment. Our use of GMM as a model of these features was based on their success in the field of text-independent speaker recognition. Text-independent speaker identification involves the recognition of a voice regardless of the spoken utterance. Text-dependent speaker identification involves recognizing voices based on a particular set of spoken words. Hidden Markov models (HMMs) are often used to model the features of the spoken response in text-dependent speaker identification, since they incorporate temporal information from the sound segment and GMMs do not (Bimbot et al., 2004). It is possible that the lower accuracy observed for certain word pairs might be due to our use of time-independent modeling of spoken features. For example, the words *high* and *low* certainly sound dissimilar in real time, but collapsing their features to a single point in time, as in a GMM, may increase their similarity. Using HMMs instead of GMMs in ChoiceKey might lead to higher identification accuracy for word pairs whose features overlap significantly on a time-independent scale, or for larger sets of words.

To summarize: ChoiceKey can easily be used to collect spoken responses and precise RTs without the need for manual coding of responses associated with voice keys. We have shown that with 30 training exemplars, ChoiceKey is inaccurate on only around 1% of trials for only around a quarter of participants. We doubt whether this error rate is much different from errors made using other forms of response collection (i.e., pressing the wrong button when using a keyboard or mouse). Some may worry that the time taken to train ChoiceKey using 30 exemplars might be restrictive or offer no benefit over the time taken for participants to learn response-button mappings; however, with only two responses, this training would take only 2 min, assuming 2 sec per response. We also note that unlike a participant, ChoiceKey, once trained, will not forget its training. At present ChoiceKey's only major drawback is that it provides highly accurate identification of responses only in paradigms where two responses are used.

AUTHOR NOTE

This research was supported by Australian Research Council Discovery Project DP0881244 to S.D.B. and A.H. Correspondence concerning this article should be addressed to C. Donkin, School of Psychology, University of Newcastle, Callaghan, NSW 2308, Australia (e-mail: chris.donkin@newcastle.edu.au).

REFERENCES

BERINGER, J. (1992). Timing accuracy of mouse response registration on the IBM microcomputer family. *Behavior Research Methods, Instruments, & Computers*, **24**, 486-490.

BIMBOT, F., BONASTRE, J.-F., FREDOUILLE, C., GRAVIER, G., MAGRIN-CHAGNOLLEAU, I., MEIGNIER, S., ET AL. (2004). A tutorial on text-independent speaker verification. *EURASIP Journal of Applied Signal Processing*, **4**, 430-451.

CROSBIE, J. (1990). The Microsoft mouse as a multipurpose response device for the IBM PC/XT/AT. *Behavior Research Methods, Instruments, & Computers*, **11**, 305-316.

LACOUTURE, Y., & MARLEY, A. A. J. (2004). Choice and response time processes in the identification and categorization of unidimensional stimuli. *Perception & Psychophysics*, **66**, 1206-1226.

PLANT, R. R., HAMMOND, M., & TURNER, G. (2004). Self-validating presentation and response timing in cognitive paradigms: How and why? *Behavior Research Methods, Instruments, & Computers*, **36**, 291-303.

PLANT, R. R., HAMMOND, M., & WHITEHOUSE, T. (2003). How choice of mouse may affect response timing in psychological studies. *Behavior Research Methods, Instruments, & Computers*, **35**, 276-284.

RASTLE, K., & DAVIS, M. H. (2002). On the complexities of measuring naming. *Journal of Experimental Psychology: Human Perception & Performance*, **28**, 307-314.

RATCLIFF, R., & ROUDER, J. N. (1998). Modeling response times for two-choice decisions. *Psychological Science*, **9**, 347-356.

REYNOLDS, D. A. (1992). *A Gaussian mixture modeling approach to text-independent speaker identification*. Unpublished doctoral dissertation, Georgia Institute of Technology.

REYNOLDS, D. A., QUATIERI, T. F., & DUNN, R. B. (2000). Speaker verification using adapted Gaussian mixture models. *Digital Signal Processing*, **10**, 19-41.

RUBIN, D., HINTON, S., & WENZEL, A. (1999). The precise time course of retention. *Journal of Experimental Psychology: Learning, Memory, & Cognition*, **25**, 1161-1176.

SHIMIZU, H. (2002). Measuring keyboard response delays by comparing keyboard and joystick inputs. *Behavior Research Methods, Instruments, & Computers*, **34**, 250-256.

SIMON, J. R., & SUDALAIMUTHU, P. (1979). Effects of S–R mapping and response modality on performance in a Stroop task. *Journal of Experimental Psychology: Human Perception & Performance*, **5**, 176-187.

STEVENS, S. S., VOLKMANN, J., & NEWMAN, E. (1937). A scale for the measurement of the psychological magnitude of pitch. *Journal of the Acoustical Society of America*, **8**, 185-190.

STEWART, N. (2006). A PC parallel port button box provides millisecond response time accuracy under Linux. *Behavior Research Methods*, **38**, 170-173.

TREWIN, S., & PAIN, H. (1999). Keyboard and mouse errors due to motor disabilities. *International Journal of Human-Computer Studies*, **50**, 109-144.

VIDULICH, M. A., & WICKENS, C. D. (1985). Stimulus-central processing–response compatibility: Guidelines for the optimal use of speech technology. *Behavior Research Methods, Instruments, & Computers*, **17**, 243-249.

VOSS, A., LEONHART, R., & STAHL, C. (2007). How to make your own response boxes: A step-by-step guide for the construction of reliable and inexpensive parallel-port response pads from computer mice. *Behavior Research Methods*, **39**, 797-881.

WANG, H., & PROCTOR, R. W. (1996). Stimulus–response compatibility as a function of stimulus code and response modality. *Journal of Experimental Psychology: Human Perception & Performance*, **22**, 1201-1217.

SUPPLEMENTAL MATERIALS

The ChoiceKey source files and instructions may be downloaded as supplemental materials for this article from brm.psychonomic-journals.org/content/supplemental.

APPENDIX A
Example Experiment

We provide MATLAB code for a mock experiment (example.m), where participants are asked to determine on each trial which of two tones differing in loudness is presented. The purpose of this code is not only to help the user collect responses using ChoiceKey, but also to show how MATLAB can be used to control a simple experiment. To begin the experiment, "example" (the name of the .m file) should be entered into the MATLAB Command Window. The experiment starts by calling of the *train* function, which allows for the recording of responses and the subsequent training of ChoiceKey. After the training, a test of ChoiceKey's identification accuracy is performed using the *traintest* function. After the function reports accuracy and number of errors, the Enter key must be pressed to continue to the experiment.

When the experiment begins, participants are prompted to press any key to continue. The function *getkeywait* has been included, since it is a handy way to get MATLAB to wait to accept and then return keyboard responses. A fixation cross is then presented for 300 msec, followed by the presentation of the stimulus. In this experiment a tone is played; however, this can be easily adapted to any other stimuli, such as strings of characters, using code similar to that used to display the fixation cross. Similarly, images can be displayed using the *imread* and *image* functions in MATLAB.

The function *test* then allows the participant to speak their response, and will return the response which ChoiceKey calculates to be most probable given its training, as well as the RT. Feedback is displayed for 1 sec, as either the word "Correct" or the correct response, depending on whether the participant was correct or incorrect, respectively. At the end of each block the block number, trial number, stimulus presented, RT and given response are all recorded to a text file. If they have completed all blocks, participants are either given a break of fixed duration or thanked for their participation. Following is pseudocode for the example experiment:

```
#Train ChoiceKey using the train function
train()
#Test the trained version of ChoiceKey
traintest()
#The experiment
For (k in 1:number of trials)
    #Show the stimuli
    showstimuli()
    #Collect the response and use ChoiceKey to get RT and response
    test()
end
```

APPENDIX B

The file Donkin-BRM-2009.zip is available as supplemental materials from brm.psychonomic-journals.org/content/supplemental. The zip file contains the functions, and their respective .m files, required for ChoiceKey to run. The end user will normally only be concerned with the following four functions.

The *train* Function

Typically, the first function employed is *train*. Participants are first given the complete response set, followed by a series of presentations of each word individually. During each presentation, participants are asked to read aloud the presented word. These initial utterances form an exemplar set which ChoiceKey uses to make all future identifications. Only one parameter must be set for the *train* function, the response set: "words." Optional parameters are the number of responses per word to use for training, "ex," the duration of recording, "duration," the frequency of recording, "Fs," and "trigger," the input energy required before audio capture begins. The default number of exemplars that ChoiceKey uses is 10. After the initial exemplars are recorded there will be a short pause of around 30 sec to 1 min, depending on the size of the response set, while ChoiceKey extracts the features and builds a Gaussian mixture model for each exemplar (Reynolds et al., 2000). We experimented with different numbers of features (Gaussians) and found 5 (the default value) to be best with our response sets. Both a smaller and larger number of Gaussians decreased accuracy, and larger values increased computational time.

The *traintest* Function

The *testtrain* function provides a test of ChoiceKey's identification accuracy for each participant. An extra set of exemplars for each word in the response set is recorded, then used to provide an estimate of expected identification accuracy for the participant. The *testtrain* function requires the response set, "words," the number of responses per word to use in testing, "ntest," and the results of the training, "mu," "sigma," and "c," to be given. The optional parameters are the same as for the *train* function as well as an additional parameter, "silent," which defaults to "F" (false). The *traintest* function returns the proportion of correctly identified responses and displays it, and the number of errors in identification, in the MATLAB Command Window if "silent" is not set to "T" (true).

APPENDIX B (Continued)

The experimenter may choose to use this feedback to decide whether the expected identification accuracy is too low, and whether or not to use ChoiceKey with this participant, or possibly to use the extra responses just recorded as additional training exemplars. After the accuracy is displayed on-screen (if "silent" is set to "F"), the experimenter is asked to decide whether or not to use the additionally recorded responses as the exemplars in training ChoiceKey. If silent is set to "T," the participant's accuracy is written to a text document called acc.txt. If "yes" is chosen after the prompt, a pause will occur while ChoiceKey is trained on the new responses.

The *callib* Function

The *callib* function offers the experimenter a method of estimating the differences in identifying onset time for different responses. Participants are instructed to respond with one of the words from the response set for a block of *n* trials each time they are presented with a neutral stimulus (a "+" sign). The process is repeated for each word in the response set. The function writes to a text file, callib.txt, the average RT for each response. The *callib* function requires as input the response set, "words," and the number of recordings per response, "nrec." The optional parameters "duration," "Fs," and "trigger," default to 1.5 sec, 44100 Hz, and 0.05, respectively.

The *test* Function

After ChoiceKey has been trained for a participant's voice, the *test* function can be called whenever response collection is required. This function will record the participant's response and return the most likely spoken response, given the set of exemplars recorded in the training stage. The time taken to make the response is also returned. As input, the *test* function needs the three parameters "mu," "sigma," and "c" returned by the *train* function. Optional parameters also include "duration," if different from 1.5 sec; "Fs," if different from 44100 Hz; and "trigger," if different from 0.05.

Once the function is called, the audio capture device is activated and waits until audio input reaches the "trigger" threshold, after which it records audio signal for a set duration. The time taken from stimulus onset to the audio signal reaching threshold is recorded as the RT, and returned to the user. The trigger threshold value default of 0.05 worked well in our testing; however, this value can be changed by the user. For example, if background noise is present (e.g., from a computer fan), the threshold may be increased to reduce the false alarm rate due to triggering by background noise. However, setting this value too high may result in responses being missed by ChoiceKey.

(Manuscript received May 27, 2008;
revision accepted for publication August 26, 2008.)