

Fast and accurate annotation of acoustic signals with deep neural networks

Elsa Steinfath^{1,2}, Adrian Palacios^{1,2}, Julian R Rottschäfer^{1,2}, Deniz Yuezak^{1,2}, Jan Clemens^{1,3*}

*For correspondence:
clemensjan@gmail.com (JC)

¹European Neuroscience Institute - A Joint Initiative of the University Medical Center Göttingen and the Max-Planck-Society, Göttingen, Germany; ²International Max Planck Research School and Göttingen Graduate School for Neurosciences, Biophysics, and Molecular Biosciences (GGNB) at the University of Göttingen, Göttingen, Germany;
³Bernstein Center for Computational Neuroscience, Göttingen, Germany

Abstract Acoustic signals serve communication within and across species throughout the animal kingdom. Studying the genetics, evolution, and neurobiology of acoustic communication requires annotating acoustic signals: segmenting and identifying individual acoustic elements like syllables or sound pulses. To be useful, annotations need to be accurate, robust to noise, and fast.

We here introduce *DAS*, a method that annotates acoustic signals across species based on a deep-learning derived hierarchical presentation of sound. We demonstrate the accuracy, robustness, and speed of *DAS* using acoustic signals with diverse characteristics from insects, birds, and mammals. *DAS* comes with a graphical user interface for annotating song, training the network, and for generating and proofreading annotations. The method can be trained to annotate signals from new species with little manual annotation and can be combined with unsupervised methods to discover novel signal types. *DAS* annotates song with high throughput and low latency, allowing realtime annotations for closed-loop experimental interventions. Overall, *DAS* is a universal, versatile, and accessible tool for annotating acoustic communication signals.

Introduction

Animals produce sounds to foster group cohesion (*Haack et al., 1983; Janik and Slater, 1998; Chaverri et al., 2012*), to signal the presence of food, friend, or foe (*Cäesar et al., 2013; Clay et al., 2012*), and to find and evaluate mating partners (*Baker et al., 2019; Behr and von Helversen, 2004; Holy and Guo, 2005; Sangiamo et al., 2020*). Studying acoustic communication not only provides insight into social interactions within and across species; it can also reveal the mechanisms driving complex behaviors: The genetics and evolution of signal production and recognition (*Ding et al., 2016, 2019*), the genes and circuits driving song learning (*Kollmorgen et al., 2020*), or the fast and precise sensorimotor transformations involved in vocal interactions (*Coen et al., 2016; Cator et al., 2009; Fortune et al., 2011; Okobi et al., 2019*). The first step in many studies of acoustic communication is song annotation: the segmentation and labelling of individual elements in a recording. Acoustic signals are diverse and range from the repetitive long-distance calling songs of crickets, grasshoppers, and anurans (*Gerhardt and Huber, 2002*), the dynamic and context-specific

41 courtship songs of vinegar flies or rodents (*Coen et al., 2014; Clemens et al., 2018; Neunuebel et al.,*
42 *2015; Sangiamo et al., 2020*), to the complex vocalizations produced by some birds and primates
43 (*Lipkind et al., 2013; Weiss et al., 2014; Landman et al., 2020*).

44 This diversity in signal structure has spawned a zoo of annotation tools (*Arthur et al., 2013; Coffey et al., 2019; Tachibana et al., 2020; Goffinet et al., 2021; Kounura and Okanoya, 2016; Cohen et al., 2020*), but existing methods still face challenges: First, assessing vocal repertoires and their relation to behavioral and neural dynamics (*Clemens et al., 2018; Coffey et al., 2019; Neunuebel et al., 2015; Fortune et al., 2011; Okobi et al., 2019*) requires annotations to be complete and temporally precise even at low signal levels, but annotation can fail when signals are weak (*Coffey et al., 2019; Stern et al., 2017*). Second, analyses of large datasets and experimental interventions during behavior (*Fortune et al., 2011; Okobi et al., 2019; Bath et al., 2014; Tschida and Mooney, 2012; Stowers et al., 2017*) need annotations to be fast, but existing methods are often slow. Last, annotation methods should be flexible and adaptable (*Ding et al., 2016, 2019; Clemens et al., 2018; Clemens and Hennig, 2013*), but existing methods often work only for restricted types of signals or adapting them to new signals requires tedious manual tuning (*Clemens et al., 2018*).

56 In brief, an accurate, fast, and flexible framework for annotating song across species is missing.
57 A general framework would not only improve upon existing methods but would also facilitate
58 the study of species for which automated methods do not yet exist. Deep neural networks have
59 emerged as powerful and flexible tools for solving data annotation tasks relevant for neuroscience
60 such as object recognition, pose tracking, or speech recognition (*Krizhevsky et al., 2012; Graves and Jaitly, 2014; Mathis et al., 2018; Pereira et al., 2018; Graving et al., 2019*). These methods are
61 not only fast and accurate but also easily adapted to novel signals by non-experts since they only
62 require annotated examples for learning. Recently, deep neural networks have also been used
63 for annotating animal vocalizations (*Oikarinen et al., 2019; Coffey et al., 2019; Cohen et al., 2020; Sainburg et al., 2020; Arthur et al., 2021; Goffinet et al., 2021*).

66 We here present a new deep-learning based framework for annotating acoustic signals, called
67 *Deep Audio Segmenter (DAS)*. We test the framework on a diverse set of recordings from insects,
68 birds, and mammals, and show that *DAS* annotates song in single- and multi-channel recordings
69 with high accuracy. The framework produces annotations with low latency on standard PCs and is
70 therefore ideally suited for closed-loop applications. Small to moderate amounts of manual anno-
71 tations suffice for adapting the method to a new species and annotation work can be simplified by
72 combining *DAS* with unsupervised methods. We provide *DAS* as an open source software package
73 with a graphical user interface for manually annotating audio, training the network, and inferring
74 and proofreading annotations. Integration into existing frameworks for signal analysis or experi-
75 mental control is possible using a programmatic interface. The code and documentation for *DAS*
76 are available at <https://janclemenslab.org/das/>.

77 Results

78 Architecture and working principle of DAS

79 Acoustic signals are defined by features on multiple timescales—the fast harmonic oscillations of
80 the sound carrier (<10 ms), modulations of amplitude (AM) and frequency (FM) (10–100 ms), and
81 the sequencing of different AM and FM patterns into bouts, syllables, or phrases (10–1000 ms).
82 These patterns are typically made explicit using a hand-tuned pre-processing step based on time-
83 resolved Fourier or wavelet transforms (*Arthur et al., 2013; Van Segbroeck et al., 2017; Coffey et al., 2019; Oikarinen et al., 2019; Cohen et al., 2020*). Most deep-learning based methods then
85 treat this pre-defined spectrogram as an image and use methods derived from computer vision to
86 extract the AM and FM features relevant for annotation (*Oikarinen et al., 2019; Coffey et al., 2019; Cohen et al., 2020*). Recurrent units are sometimes used to track the sound features over time
88 (*Cohen et al., 2020*). This approach can produce accurate annotations but has drawbacks: First,
89 the spectrogram constitutes a strong and proven pre-processing step, but it is unsuitable for some

90 signal types, like short pulsatile signals. Second, the pre-processing transform is typically tuned by
91 hand and may therefore require expert knowledge for it to produce optimal results. Lastly, the
92 recurrent units used in some methods (*Cohen et al., 2020*) excel at combining information over
93 time to provide the context information necessary to annotate spectrally complex signals, but they
94 can be hard to train and slow to run (*Bai et al., 2018*).

95 *DAS* solves these limitations in three ways: First, the pre-processing step is optional. This makes
96 *DAS* more flexible, since signals for which a time-resolved Fourier transform is not appropriate—for
97 instance, short pulsatile signals—can now also be processed. Second, the optional preprocessing
98 step is integrated and optimized with the rest of the network. This removes the need to hand-
99 tune this step and allows the network to learn a preprocessing that deviates from a time-resolved
100 Fourier or wavelet transform if beneficial (*Choi et al., 2017*). Integrating the preprocessing into
101 the network also increases inference speed due to the efficient implementation and hardware
102 acceleration of deep-learning frameworks. Third and last, *DAS* learns a task-specific representation
103 of sound features using *temporal convolutional networks* (TCNs) (*Bai et al., 2018; van den Oord et al.,*
104 *2016; Guirguis et al., 2021*) (*Figure 1–Figure Supplement 1A-E*). At the core of TCNs are so-called
105 *dilated convolutions* (*Yu and Koltun, 2016*). In standard convolutions, short templates slide over
106 the signal and return the similarity with the signal at every time point. In *dilated* convolutions,
107 these templates have gaps, allowing to analyze features on longer timescales without requiring
108 more parameters to specify the template. Stacking dilated convolutions with growing gap sizes
109 results in a hierarchical, multi-scale representation of sound features, which is ideally suited for
110 the hierarchical and harmonic structure of animal vocalizations.

111 The output of the deep neural network in *DAS* is a set of confidence scores for each audio
112 sample, corresponding to the probability of each song type (*Figure 1C*). Annotation labels for the
113 different song types are mutually exclusive and are produced by comparing the confidence score
114 to a threshold or by choosing the most probable song type. Brief gaps in the annotations are closed
115 and short spurious detections are removed to smoothen the annotation. For song types that are
116 described as events, like the pulses in fly song (*Figure 1A*), the event times are extracted as local
117 maxima that exceed a confidence threshold.

118 ***DAS* accurately annotates song from a diverse range of species**

119 Fly courtship song

120 We first tested *DAS* on the courtship song of *Drosophila melanogaster*, which consists of two ma-
121 jor modes (*Figure 1A*): The sine song, which corresponds to sustained oscillations with a species-
122 specific carrier frequency (150 Hz), and two types of pulse song, which consists of trains of short
123 (5–10 ms) pulses with carrier frequencies between 180 and 500 Hz, produced with a species-specific
124 interval (35–45 ms in *D. melanogaster*). Males dynamically choose the song modes based on sen-
125 sory feedback from the female (*Coen et al., 2014, 2016; Clemens et al., 2018; Calhoun et al., 2019*).
126 Despite the relative simplicity of the individual song elements, an accurate annotation of fly song
127 is challenging because of low signal-to-noise ratio (SNR): The song attenuates rapidly with distance
128 (*Bennet-Clark, 1998*) and is highly directional (*Morley et al., 2018*), which can lead to weak signals
129 if the male is far from the microphone (*Figure 1A*). Moreover, the interactions between the flies
130 introduce pulsatile noise and complicate the accurate and complete annotation of the pulse song.

131 We first trained *DAS* to detect the pulse and the sine song recorded using a single microphone
132 (data from *Stern (2014)*) and compared the performance of *DAS* to that of the current state-of-the-
133 art in fly song segmentation, *FlySongSegmenter* (FSS) (*Arthur et al., 2013; Coen et al., 2014; Clemens*
134 *et al., 2018*). Annotation performance was quantified using *precision*, the fraction of correct de-
135 tections, and *recall*, the fraction of true song that is detected (*Figure 1E, J, Figure 1–Figure Supple-*
136 *ment 1F, G*). We counted detected pulses within 10 ms of a true pulse as correct detections. 10 ms
137 corresponds to 1/4th of the typical interval between pulses in a train and results are robust to the
138 choice of this value (*Figure 1–Figure Supplement 2A*). *DAS* detects pulses with a high precision of
139 97 % — only 3 % of all detected pulses are false detections — and a high recall of 96 % — it misses

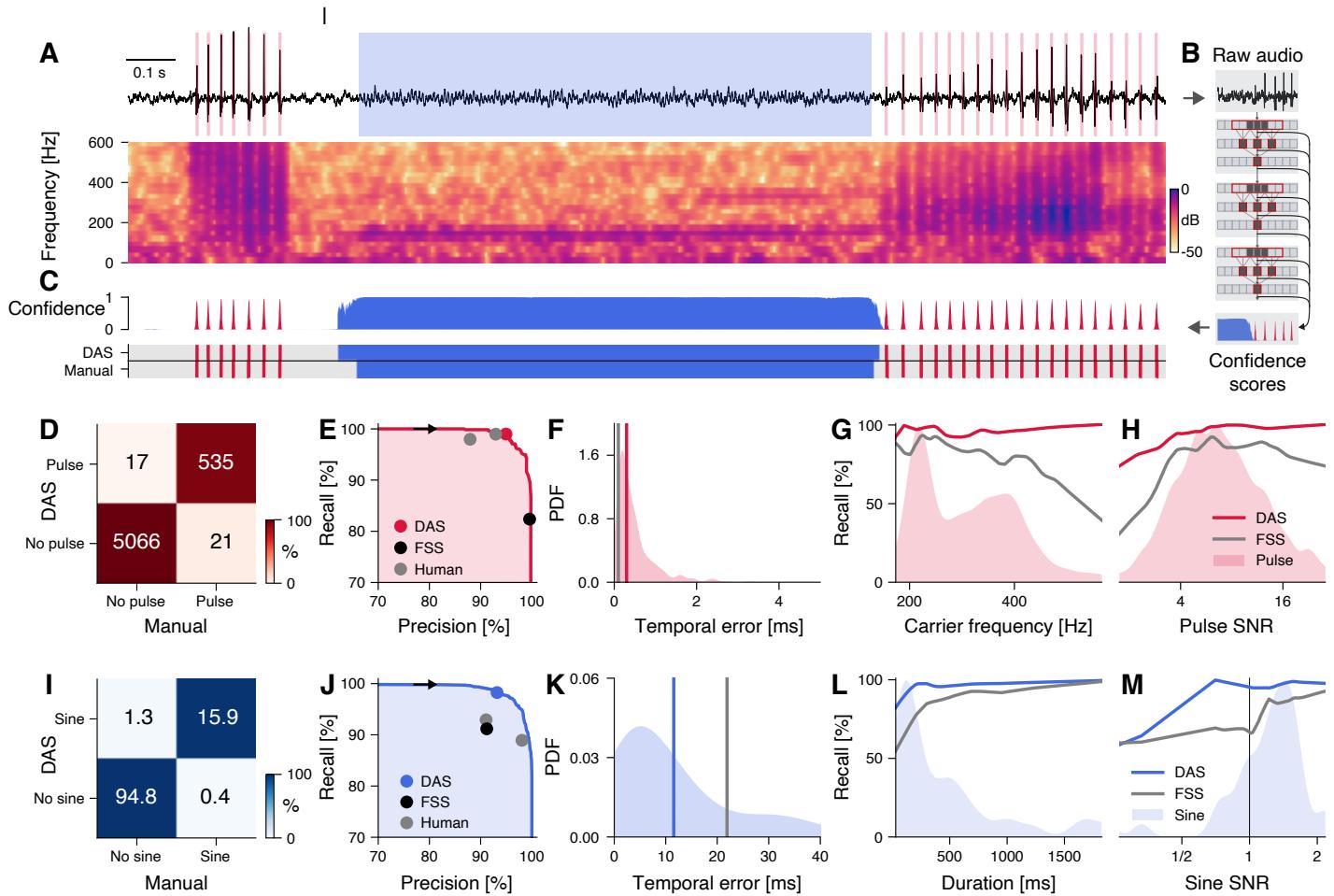


Figure 1-Figure supplement 1. DAS architecture and evaluation.

Figure 1-Figure supplement 2. Performance and the role of context for annotating fly pulse song.

Figure 1-Figure supplement 3. Performance for multi-channel recordings of fly courtship song.

140 only 4 % of all pulses. This is a substantial improvement in recall over *FSS*, which has slightly higher
141 precision (99 %) but misses 13 % of all pulses (87 % recall) (*Figure 1D, E*). In *DAS*, the balance be-
142 tween precision and recall can be controlled via the confidence threshold, which corresponds to
143 the minimal confidence required for labelling a pulse (*Figure 1C*): Lowering this threshold from
144 0.7 to 0.5 yields a recall of 99 % for pulse song with a modest reduction in precision to 95 %. The
145 performance gain of *DAS* over *FSS* for pulse stems from better recall at high frequencies (>400Hz)
146 and low SNR (*Figure 1G, H*). To assess *DAS* performance for sine song, we evaluated the sample-
147 wise precision and recall. *DAS* has similar precision to *FSS* (92 % vs 91 %) but higher recall (98 % vs.
148 91 %) (*Figure 1I, J*). Recall is higher in particular for short sine songs (<100 ms) and at low SNR (<1.0)
149 (*Figure 1L, M*). The performance boost for pulse and sine arises because *DAS* exploits context in-
150 formation, similar to how humans annotate song: For instance, *DAS* discriminates soft song pulses
151 from pulsatile noise based on the pulse shape but also because song pulses occur in regular trains
152 while noise pulses do not (*Figure 1–Figure Supplement 2C*). A comparison of *DAS*’ performance to
153 that of human annotators reveals that our methods exceeds human-level performance for pulse
154 and sine (*Figure 1E, J, Table 3*).

155 Temporally precise annotations are crucial, for instance when mapping sensorimotor trans-
156 formations based on the timing of behavioral or neuronal responses relative to individual song
157 elements (*Coen et al., 2014; Srivastava et al., 2017; Long and Fee, 2008; Benichov and Vallentin,
158 2020*). We therefore quantified the temporal error of the annotations produced by *DAS*. For pulse
159 song, the temporal error was taken as the distance of each pulse annotated by *DAS* to the nearest
160 true pulse. The median temporal error for pulse is 0.3 ms which is negligible compared to the av-
161 erage duration of a pulse (5-10 ms) or of a pulse interval (35-45 ms) (*Deutsch et al., 2019*). For sine
162 song, the median temporal error for on- and offsets was 12 ms, which is almost half of that of *FSS*
163 (22 ms). Sine song can have low SNR (*Figure 1M*) and fades in and out, making the precise identi-
164 fication of sine song boundaries difficult even for experienced manual annotators (see *Figure 1A,
165 C*).

166 Recording song during naturalistic interactions in large behavioral chambers often requires
167 multiple microphones (*Coen et al., 2014; Neunuebel et al., 2015*). To demonstrate that *DAS* can
168 process multi-channel audio, we trained *DAS* to annotate recordings from a chamber tiled with 9
169 microphones (*Coen et al., 2014*) (*Figure 1–Figure Supplement 3, Figure 1–Figure Supplement 1B*).
170 *DAS* processes multi-channel audio by using filters that take into account information from all chan-
171 nels simultaneously. As is the case for existing methods (*Arthur et al., 2013*), we achieved maximal
172 performance by training separate networks for the pulse and for the sine song (*Table 1*). In multi-
173 channel recordings, *DAS* annotates pulse song with 98 % precision and 94 % recall, and sine song
174 with 97 % precision and 93 % recall, and matches the performance of *FSS* (*FSS* pulse precision/recall
175 99/92, sine 95/93) (*Figure 1–Figure Supplement 3D-L*). Annotations of multi-channel audio have
176 high temporal precision for pulse (*DAS* 0.3 ms, *FSS* 0.1 ms) and sine (*DAS* 8 ms, *FSS* 15 ms) (*Figure 1–
177 Figure Supplement 3E, J*). Overall, *DAS* performs better or as well as the current state-of-the-art
178 method for annotating single and multi-channel recordings of fly song.

179 Mouse ultrasonic vocalizations

180 Mice produce ultrasonic vocalizations (USVs) in diverse social contexts ranging from courtship to
181 aggression (*Sangiamo et al., 2020; Warren et al., 2020; Neunuebel et al., 2015*). We tested *DAS* us-
182 ing audio from an intruder assay, in which an anesthetized female was put into the home cage and
183 the USVs produced by a resident female or male were recorded (*Ivanenko et al., 2020*). The female
184 USVs from this assay typically consist of pure tones with weak harmonics and smooth frequency
185 modulations that are often interrupted by frequency steps (*Figure 2A, B*). The male USVs are sim-
186 ilar but also contain complex frequency modulations not produced by the females in this assay
187 (*Figure 2C, D*). Recording noise from animal movement and interaction as well as the frequency
188 steps often challenge spectral threshold-based annotation methods and tend to produce false
189 positive syllables (*Tachibana et al., 2020; Coffey et al., 2019*). Moreover, weak signals often lead to

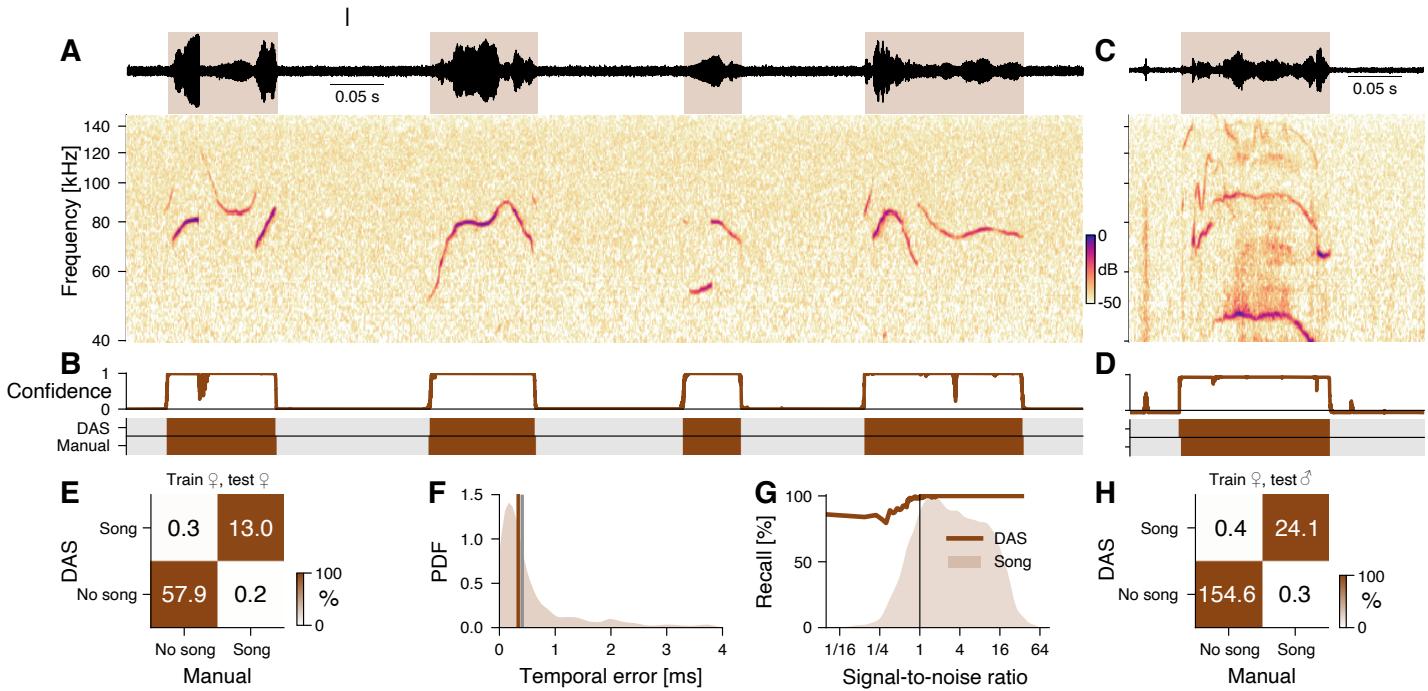


Figure 2. *DAS* performance for mouse ultrasonic vocalizations.

A Waveform (top) and spectrogram (bottom) of USVs produced by a female mouse in response to an anesthetized female intruder. Shaded areas (top) show manual annotations.

B Confidence scores (top) and *DAS* and manual annotations (bottom) for the female USVs in A. Brief gaps in confidence are filled to ensure smooth annotations.

C Example of male USVs with sex-specific characteristics produced in the same assay.

D Confidence scores (top) and *DAS* and manual annotations (bottom) for the male USVs in C from a *DAS* network trained to detect female USVs.

E Confusion matrix from a female-trained network for a test set of female USVs. Color indicates the percentage (see color bar) and text labels the seconds of song in each quadrant.

F Distribution of temporal errors for syllable on- and offsets in female USVs. The median temporal error is 0.3 ms for *DAS* (brown line) and 0.4 ms for *USVSEG* (*Tachibana et al., 2020*), a method developed to annotate mouse USVs (grey line).

G Recall of the female-trained network (brown line) as a function of SNR. The brown shaded area represents the distribution of SNRs for all samples containing USVs. Recall is high even at low SNR.

H Confusion matrix of the female-trained *DAS* for a test set of male USVs (see C, D for examples). Color indicates the percentage (see color bar) and text labels the seconds of song in each quadrant.

Figure 2–Figure supplement 1. Performance for marmoset vocalizations.

missed syllables or imprecisely delimited syllables. We first trained and tested *DAS* on recordings of a female mouse interacting with an anesthetized female intruder (Figure 2A). *DAS* annotates the female USVs with excellent precision (98 %) and recall (99 %) (Figure 2E) and low median temporal error (0.3 ms) (Figure 2F). *DAS* is robust to noise: Even for weak signals (SNR 1/16) the recall is 90 % (Figure 2G). These performance values are on par with that of methods specialized to annotate USVs (*Tachibana et al., 2020; Coffey et al., 2019; Van Segbroeck et al., 2017*) (see Table 2). USVs of female and male residents have similar characteristics (Figure 2A, B) and the female-trained *DAS* network also accurately annotated the male vocalizations (Figure 2H). Notably, even the male syllables with characteristics not seen in females in the paradigm were detected (Figure 2D). Overall, *DAS* accurately and robustly annotates mouse USVs and generalizes across sexes.

200 Marmoset vocalizations

We next examined the robustness of annotations produced by *DAS* to noisy recordings and variable vocalization types, by training a network to annotate vocalization from pairs of marmosets (*Landelman et al., 2020*). The recordings contain lots of background noises like faint calls from nearby animals, overdrive from very loud calls of the recorded animals, and large variability within syllables.

ble types (**Figure 2-Figure Supplement 1A-D**). Recently, a deep-learning based method was shown to produce good performance (recall 77 %, precision 85 %, 12.5 ms temporal error) when trained on 16000 syllables to recognize seven vocalization types (*Oikarinen et al., 2019*). We trained DAS on 1/9th of the data (1800 syllables) containing 4 of the 7 vocalization types. Despite the noisy and variable vocalizations, DAS achieves high syllable-wise precision and recall (96 %, 92 %, (**Figure 2-Figure Supplement 1E, F**)). Note that DAS obtains this higher performance at millisecond resolution (temporal error 4.4 ms, **Figure 2-Figure Supplement 1G**), while the method by *Oikarinen et al. (2019)* only produces annotations with a resolution of 50 ms (**Table 2**).

213 Bird song

214 Bird song is highly diverse and can consist of large, individual-specific repertoires. The spectral
215 complexity and large diversity of the song complicates the annotation of syllable types. Tradition-
216 ally, syllable types are annotated based on statistics derived from the segmented syllable spec-
217 trogram. Recently, good annotation performance has been achieved with unsupervised methods
218 (*Sainburg et al., 2020; Goffinet et al., 2021*) and deep neural networks (*Koumura and Okanoya,*
219 *2016; Cohen et al., 2020*). We first trained DAS to annotate the song from four male Bengalese
220 finches (data and annotations from *Nicholson et al. (2017)*). The network was then tested on a
221 random subset of the recordings from all four individuals which contained 37 of the 48 syllable
222 types from the training set (**Figure 3A, B, Figure 3-Figure Supplement 1A, B**). DAS annotates the
223 bird song with high accuracy: Sample-wise precision and recall are 97 % and syllable on- and off-
224 sets are detected with sub-millisecond precision (median temporal error 0.3 ms, **Figure 3C**). The
225 types of 98.5 % the syllables are correctly annotated, with only 0.3 % false positives (noise anno-
226 tated as a syllable), 0.2 % false negatives (syllables annotated as noise), and 1 % type confusions
227 (**Figure 3D, Figure 3-Figure Supplement 1C-D**). This results in a low sequence error (corresponding
228 to the minimal number of substitutions, deletions, or insertions required to transform the true
229 sequence of syllables into the inferred one) of 0.012. Overall, DAS performs as well as specialized
230 deep learning-based methods for annotating bird song (*Koumura and Okanoya, 2016; Cohen et al.,*
231 *2020*) (**Table 2**).

232 To further demonstrate the robustness of DAS, we trained a network to annotate song from
233 Zebra finches. In Zebra finch males, individual renditions of a given syllable type tend to be more
234 variable (*Fitch et al., 2002*). Moreover, the particular recordings used here (*Goffinet et al., 2021*)
235 contain background noise from the bird's movement. Despite the variability and noise, DAS anno-
236 tates the six syllables from a male's main motif with excellent precision and recall, and low tempo-
237 ral error, demonstrating that DAS is robust to song variability and recording noise (**Figure 3-Figure**
238 **Supplement 2**).

239 In summary, DAS accurately and robustly annotates a wide range of signals—from the pulsatile
240 song pulses of flies to the spectrally complex syllables of mammals and birds. DAS therefore consti-
241 tutes a universal method for annotating acoustic signals that is as good as or better than methods
242 specialized for particular types of signals (**Table 2**).

243 **DAS is fast**

244 To efficiently process large corpora of recordings and to be suitable for closed-loop applications,
245 DAS needs to infer annotations quickly. We therefore assessed the throughput and latency of DAS.
246 Throughput measures the rate at which DAS annotates song and high throughput means that large
247 datasets are processed quickly. Across the five species tested here, DAS has a throughput of 8-82x
248 realtime on a CPU and of 24-267x on a GPU (**Figure 4A**). This means that a 60-minute recording
249 is annotated in less than 5 minutes on a standard desktop PC and in less than 1.5 minutes using
250 a GPU, making the annotation of large datasets feasible (**Figure 4-Figure Supplement 1A-H**). The
251 differences in throughput arise from the different sample rates and the network architectures: The
252 marmoset network is fastest because of a relatively shallow architecture with only 2 TCN blocks
253 and a low sampling rate (44.1 kHz). By contrast, the multi-channel *Drosophila* networks have the

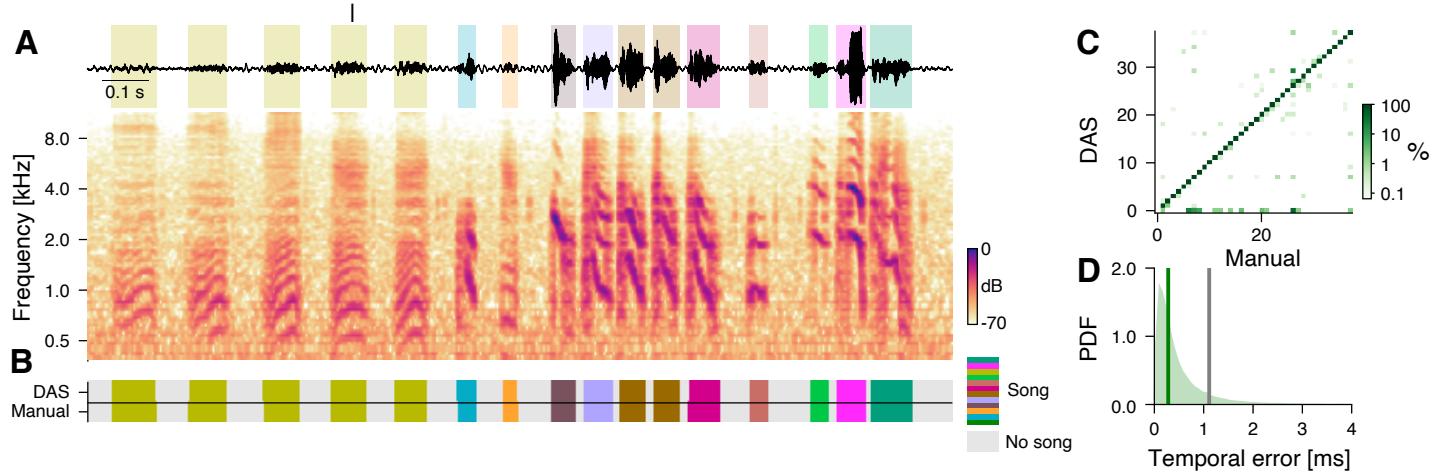


Figure 3. DAS performance for the song of Bengalese finches.

A Waveform (top) and spectrogram (bottom) of the song of a male Bengalese finch. Shaded areas (top) show manual annotations colored by syllable type.

B DAS and manual annotation labels for the different syllable types in the recording in A (see color bar). *DAS* accurately annotates the syllable boundaries and types.

C Confusion matrix for the different syllables in the test set. Color was log-scaled to make rare the annotation errors more apparent (see color bar). Rows depict the probability with which *DAS* annotated each syllable as any of the 37 types in the test dataset. The type of 98.5 % of the syllables were correctly annotated, resulting in the concentration of probability mass along the main diagonal.

D Distribution of temporal errors for the on- and offsets of all detected syllables (green shaded area). The median temporal error is 0.3 ms for *DAS* (green line) and 1.1 ms for TweetyNet (Cohen et al., 2020), a method developed to annotate bird song (grey line).

Figure 3-Figure supplement 1. Performance for the song of Bengalese finches.

Figure 3-Figure supplement 2. Performance for the song of a Zebra finch.

254 lowest throughput because of multi-channel inputs (9 channels at 10.0 kHz) and a comparatively
 255 deep architecture with four TCN blocks (**Table 4**).

256 A measure of speed crucial for closed-loop experiments is latency, which quantifies the time it
 257 takes to annotate a chunk of song and determines the delay for experimental feedback. Latencies
 258 are short, between 7 and 15 ms (**Figure 4B**, **Figure 4-Figure Supplement 1I-P**) on CPUs and GPUs.
 259 One network parameter impacting latency is the chunk size—the duration of audio processed at
 260 once—and we find that for fly song, latency can be optimized by reducing chunk size with a minimal
 261 impact on accuracy (**Figure 4-Figure Supplement 2**). The low latency of annotation makes *DAS*
 262 well suited for triggering realtime optogenetic or acoustic feedback upon the detection of specific
 263 vocalizations (Bath et al., 2014; Stowers et al., 2017).

264 We also compared the speed of *DAS* to that of other methods that were specifically developed
 265 to annotate the types of signals tested here. Since most existing methods are not suitable for esti-
 266 mating latency due to constraints in their design and interface, we only compared throughput. We
 267 find that *DAS* achieves 3x to 10x higher throughput than existing methods (**Table 2**). This has three
 268 main reasons: First, the relatively simple, purely convolutional architecture exploits the parallel
 269 processing capabilities of modern CPUs and GPUs. Second, Fourier or wavelet-like preproces-
 270 sing steps are integrated into the network and profit from a fast implementation and hardware accel-
 271 eration. Third, for multi-channel data, *DAS* combines information from all audio channels early,
 272 which increases throughput by reducing the data bandwidth.

273 Overall, *DAS* annotates audio with high throughput (>8x realtime) and low latency (<15 ms), and
 274 is faster than the alternative methods tested here. The high speed renders suitable for annotating
 275 large corpora and for realtime applications without requiring specialized hardware.

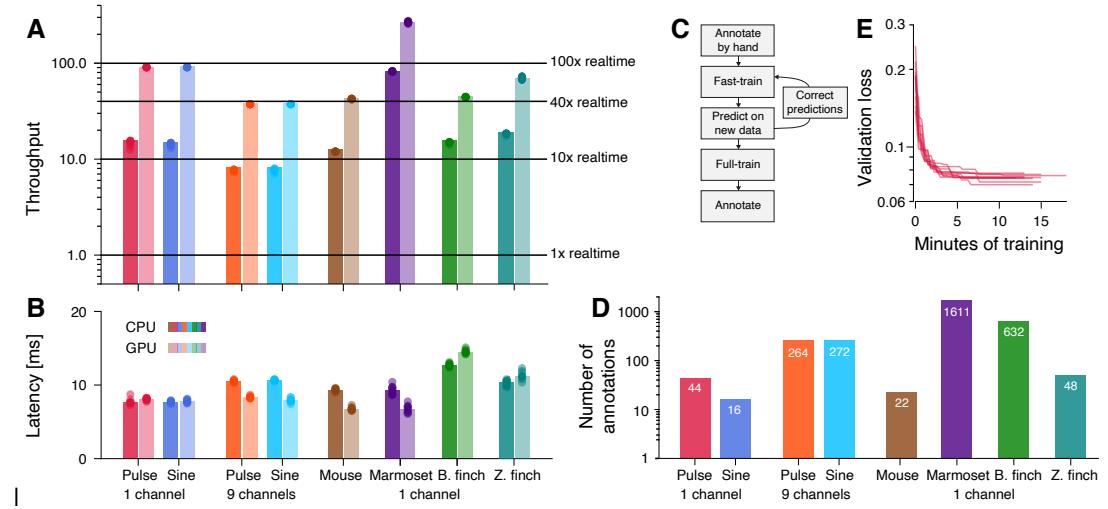


Figure 4. DAS annotates song with high throughput and low latency and requires little data.

A, B Throughput (A) and latency (B) of DAS (see also *Figure 4-Figure Supplement 1*). Throughput (A) was quantified as the amount of audio data in seconds annotated in one second of computation time. Horizontal lines in A indicate throughputs of 1, 10, and 40. Throughput is >8x realtime on a CPU (dark shades) and >24x or more on a GPU (light shades). Latency (B) corresponds to the time it takes to annotate a single chunk of audio and is similar on a CPU (dark shades) and a GPU (light shades). Multi-channel audio from flies was processed using separate networks for pulse and sine. For estimating latency of fly song annotations, we used networks with 25 ms chunks, not the 410 ms chunks used in the original network (see *Figure 1-Figure Supplement 2*).

C Flow diagram of the iterative protocol for fast training DAS.

D Number of manual annotations required to reach 90 % of the performance of DAS trained on the full data set shown in *Figure 1*, *Figure 1-Figure Supplement 3*, *Figure 2*, *Figure 2-Figure Supplement 1*, *Figure 3*, and *Figure 3-Figure Supplement 2* (see also *Figure 4-Figure Supplement 3*). Performance was calculated as the F1 score, the geometric mean of precision and recall. For most tasks, DAS requires small to modest amounts of manual annotations.

E Current best validation loss during training for fly pulse song recorded on a single channel for 10 different training runs (red lines, 18 minutes of training data). The network robustly converges to solutions with low loss after fewer than 15 minutes of training (40 epochs).

Figure 4-Figure supplement 1. Throughput and latency of inference.

Figure 4-Figure supplement 2. Reducing the chunk duration reduces latency and comes with minimal performance penalties.

Figure 4-Figure supplement 3. DAS requires small to moderate amounts of data for training.

Figure 4-Figure supplement 4. Example of fast training mouse USVs.

Figure 4-Figure supplement 5. DAS performance is robust to changes in the structural parameters of the network.

276 **DAS requires little manual annotation**

277 To be practical, *DAS* should achieve high performance with little manual annotation effort. We find
278 that *DAS* can be efficiently trained using an iterative protocol (*Pereira et al., 2018*) (*Figure 4C, Figure 4–Figure Supplement 4*): Annotate a small set of recordings and train the network for a few
280 epochs; then generate annotations on a larger set of recordings and correct these annotations.
281 Repeat the predict-correct-train cycle on ever larger datasets until performance is satisfactory. To
282 estimate the amount of manual annotations required to achieve high performance, we evaluated
283 *DAS* trained on subsets of the full training data sets used above (*Figure 4–Figure Supplement 3*).
284 We then took the number of manual annotations needed to reach 90 % of the performance of *DAS*
285 trained on the full data sets as an upper bound on the data requirements (*Figure 4D*). With a per-
286 formance threshold of 90 %, the resulting networks will produce sufficiently accurate annotations
287 for creating a larger body of training data with few corrections. Performance was taken as the F1
288 score, the geometric mean of precision and recall. For single-channel recordings of fly song, fewer
289 than 50 pulses and 20 sine songs are needed to reach 90 % of the performance achieved with the
290 full data set. For mouse vocalizations, *DAS* achieves 90 % of its peak performance with fewer than
291 25 manually annotated syllables. Even for the six syllables from a zebra finch, *DAS* reaches the
292 90 % threshold with only 48 manually annotated syllables (8 per type). Manually annotating such
293 small amounts of song for flies, mice, or zebra finches takes less than 5 minutes. Likewise, for the
294 song of Bengalese finches, 1-51 (median 8, mean 17) manual annotations are required per syllable
295 type, with one outlier requiring 200 syllables (*Figure 4–Figure Supplement 3C*). Closer inspection
296 reveals that the outlier results from an annotation error and consists of a mixture of three distinct
297 syllable types (*Figure 4–Figure Supplement 3D–F*). Even with this outlier, only 626 manually anno-
298 tated syllabes (424 without) are required in total to reach 90 % of the test performance of a network
299 trained on >3000 annotated syllables. Data requirements are higher for the multi-channel record-
300 ings of fly song (270 pulses and sine songs), and for the noisy and variable marmoset data (1610
301 annotations, 400 per type), but even in these cases, the iterative training protocol can reduce the
302 manual annotation work.

303 Overall, *DAS* requires small to moderate amounts of data for reaching high performance. High
304 throughput (*Figure 4A*) and small training data sets (*Figure 4D*) translate to short training times
305 (*Figure 4E*). The single-channel data sets typically achieve 90 % of the performance after less than
306 10 minutes of training on a GPU. Training on the full data sets typically finishes in fewer than 5
307 hours. Thus, *DAS* can be adapted to novel species in short time and with little manual annotation
308 work.

309 **DAS can be combined with unsupervised methods**

310 *DAS* is a supervised annotation method: It discriminates syllable types that have been manually
311 assigned different labels during training. By contrast, unsupervised methods can determine in un-
312 labelled data whether syllables fall into distinct types and if so, classify the syllables (*Tabler et al.,*
313 *2017; Coffey et al., 2019; Clemens et al., 2018; Goffinet et al., 2021; Sainburg et al., 2020; Sangiamo*
314 *et al., 2020; Arthur et al., 2021*). While *DAS* does not require large amounts of manual annotations
315 (*Figure 4D*), manual labeling of syllable types can be tedious when differences between syllable
316 types are subtle (*Clemens et al., 2018*) or when repertoires are large (*Sangiamo et al., 2020*). In
317 these cases, combining *DAS* with unsupervised methods facilitates annotation work. To demon-
318 strate the power of this approach, we use common procedures for unsupervised classification,
319 which consist of an initial preprocessing (e.g. into spectrograms) and normalization (e.g. of ampli-
320 tude) of the syllables, followed by dimensionality reduction and clustering (see Methods) (*Clemens*
321 *et al., 2018; Sainburg et al., 2020; Sangiamo et al., 2020*).

322 For fly song, *DAS* was trained to discriminate two major song modes, pulse and sine. However,
323 *Drosophila melanogaster* males produce two distinct pulse types, termed P_{slow} and P_{fast} (*Clemens*
324 *et al., 2018*), and unsupervised classification robustly discriminates the two pulse types as well
325 as the sine song in the *DAS* annotations (*Figure 5A–C*). Mouse USVs do not fall into distinct types

(*Tabler et al., 2017; Goffinet et al., 2021; Sainburg et al., 2020*). In this case, unsupervised clustering produces a low-dimensional representation that groups the syllables by the similarity of their spectrograms (*Tabler et al., 2017; Coffey et al., 2019; Sangiamo et al., 2020*) (*Figure 5D, E*). For marmosets, unsupervised classification recovers the four manually defined call types (*Figure 5F, G*). However, most call types are split into multiple clusters, and the clusters for trills and twitters tend to separate poorly (*Figure 4–Figure Supplement 3G*), which reflects the large variability of the marmoset vocalizations. This contrasts with the song of Zebra finches, for which the unsupervised method produces a one-to-one mapping between manually defined and unsupervised syllable types (*Goffinet et al., 2021; Sainburg et al., 2020*) (*Figure 5H, I*). For the song of Bengalese finches, the unsupervised classification recovers the manual labelling (*Goffinet et al., 2021; Sainburg et al., 2020*) (*Figure 5J, K*) and reveals manual annotation errors: For instance, the song syllable that required >200 manual annotations to be annotated correctly by *DAS* is a mixture of three distinct syllable types (*Figure 4–Figure Supplement 3C–F*).

Overall, unsupervised methods simplify annotation work: *DAS* can be trained using annotations that do not discriminate between syllables types and the types can be determined *post hoc*. If distinct types have been established, *DAS* can be retrained to directly annotate these types using the labels produced by the unsupervised method as training data.

Discussion

We here present *Deep Audio Segmenter* (*DAS*), a method for annotating acoustic signals. *DAS* annotates song in single- and multi-channel recordings from flies (*Figure 1, Figure 1–Figure Supplement 3*), mammals (Figs *Figure 2, Figure 2–Figure Supplement 1*), and birds (Figs *Figure 3, Figure 3–Figure Supplement 2*) accurately, robustly, and quickly (*Figure 4A, B*). *DAS* performs as well as or better than existing methods that were designed for specific types of vocalizations (*Koumura and Okanya, 2016; Cohen et al., 2020; Tachibana et al., 2020; Coffey et al., 2019*) (*Table 2*). *DAS* performs excellently for signals recorded on single and multiple channels (*Figure 1, Figure 1–Figure Supplement 3*), with different noise levels, and with diverse characteristics. This suggests that *DAS* is general method for accurately annotating signals from a wide range of recording setups and species.

Using a user-friendly graphical interface, our method can be optimized for new species without requiring expert knowledge and with little manual annotation work (*Figure 4C–E*). Network performance is robust to changes in the structural parameters of the network, like filter number and duration, or the network depth (*Figure 4–Figure Supplement 5*). Thus, the structural parameters do *not* need to be finely tuned to obtain a performant network for a new species. We have trained networks using a wide range of signal types (*Table 4*) and these networks constitute good starting points for adapting *DAS* to novel species. We provide additional advice for the design of novel networks in Methods. This makes the automatic annotation and analysis of large corpora of recordings from diverse species widely accessible.

We show that the annotation burden can be further reduced using unsupervised classification of syllable types, in particular for species with large or individual-specific repertoires (*Figure 5*) (*Clemens et al., 2018; Tabler et al., 2017; Coffey et al., 2019; Goffinet et al., 2021; Sainburg et al., 2020; Arthur et al., 2021*). In the future, incorporating recent advances in the self-supervised or semi-supervised training of neural networks will likely further reduce data requirements (*Mathis et al., 2021; Raghu et al., 2019; Devlin et al., 2019; Chen and He, 2020*). These approaches use unlabelled data to produce networks with a general and rich representation of sound features that can then be fine-tuned for particular species or individuals using few annotated samples. *DAS* currently does not work well with recordings in which the signals produced by multiple animals overlap. In the future, *DAS* will be extended with methods for multi-speaker speech recognition to robustly annotate vocalizations from animal groups.

Lastly, the high inference speed (*Figure 4A, B*) allows integration of *DAS* in closed-loop systems in which song is detected and stimulus playback or optogenetic manipulation is triggered with low

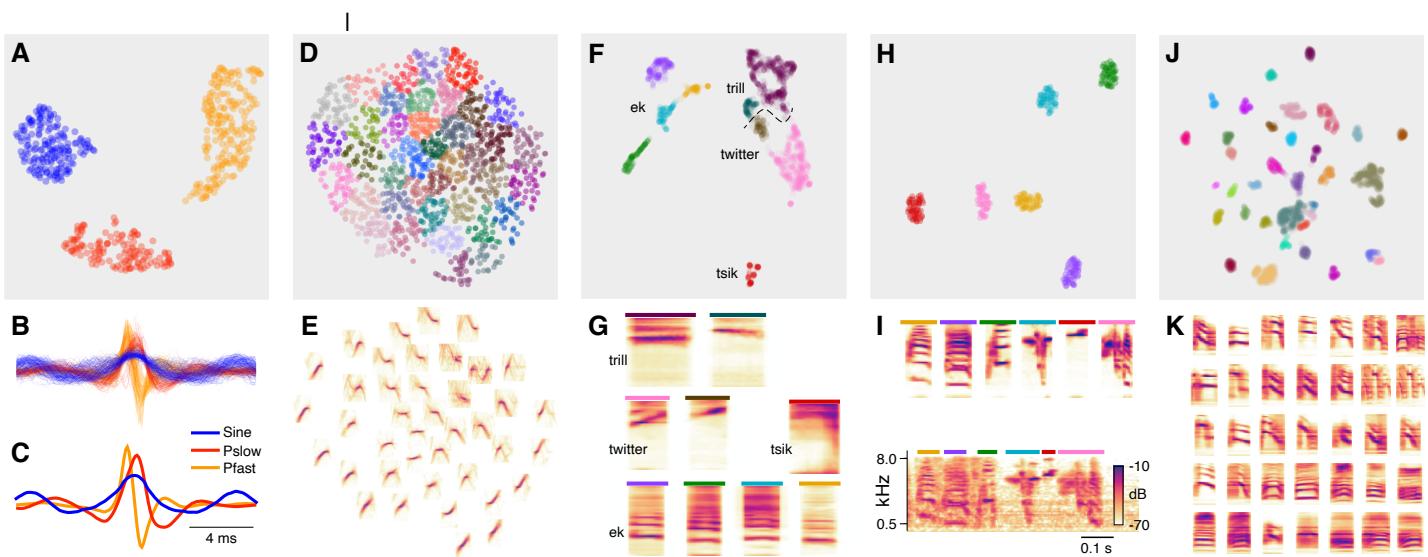


Figure 5. DAS can be combined with unsupervised methods for song classification.

A Low-dimensional representation obtained using the UMAP (*McInnes and Healy, 2018*) method of all pulse and sine song waveforms annotated by DAS in a test data set. Data points correspond to individual waveforms and were clustered into three distinct types (colors) using the density-based method HDBSCAN (*McInnes et al., 2017*).

B, C All waveforms (B) and cluster centroids (C) from A colored by the cluster assignment. Waveforms cluster into one sine (blue) and two pulse types with symmetrical (red, P_{slow}) and asymmetrical (orange, P_{fast}) shapes.

D, E Low-dimensional representation of the spectrograms of mouse USVs (D) and mean spectrogram for each cluster in D (E). Individual syllables (points) form a continuous space without distinct clusters. Song parameters vary continuously within this space, and syllables can be grouped by the similarity of their spectral contours using k-means clustering.

F, G Low-dimensional representation of the spectrograms of the calls from marmosets (F) and mean spectrogram for each cluster in F (G). Calls separate into distinct types and density-based clustering (colors) produces a classification of syllables that recovers the manual annotations (*Figure 4–Figure Supplement 3G*, homogeneity score 0.88, completeness score 0.57, v-score 0.69). Most types split into 2 to 4 clusters, reflecting the variability of the call types in marmosets. Colored bars on top of each spectrogram in G correspond to the colors for the individual clusters in F. The dashed line shows the boundary separating trills and twitters.

H, I Low-dimensional representation of the spectrograms of the syllables from one Zebra finch male, mean spectrogram for each cluster in H (I, top), and example of each clustered syllable within the motif (I, bottom). Density-based clustering (colors) recovers the 6 syllable types forming the male's motif. Colored bars on top of each spectrogram in I correspond to the colors for the individual clusters in H.

J, K Low-dimensional representation of the spectrograms of the syllables from 4 Bengalese finch males (J) and mean spectrogram for each cluster in J (K). Syllables separate into distinct types and density-based clustering (colors) produces a classification of syllables that closely matches the manual annotations (homogeneity score 0.96, completeness score 0.89, v-score 0.92).

X-axes of the average spectrograms for each cluster do not correspond to linear time, since the spectrograms of individual syllables were temporally log-rescaled and padded prior to clustering. This was done to reduce the impact of differences in duration between syllables.

376 latency (*Bath et al., 2014; Stowers et al., 2017*). In combination with realtime pose tracking (*Mathis*
377 *et al., 2018; Pereira et al., 2018; Graving et al., 2019*), *DAS* provides unique opportunities to tailor
378 optogenetic manipulations to specific behavioral contexts, for instance to dissect the neural circuits
379 underlying acoustic communication in interacting animals (*Coen et al., 2014; Fortune et al., 2011;*
380 *Okobi et al., 2019*).

381 Methods and Materials

382 Instructions for installing and using *DAS* can be found at <https://janclemenslab.org/das>. Code for the
383 *das* python module is available at <https://github.com/janclemenslab/das>, code for the unsupervised
384 methods is at https://github.com/janclemenslab/das_unsupervised. All fitted models (with example
385 data and code) can be found at <https://github.com/janclemenslab/das-menagerie>.

386 We also provide instructions for training *DAS* using google colab, which provides a GPU-accelerated
387 python environment. Colab removes the need to install GPU libraries: Annotations can be made
388 locally in the GUI without a GPU and training and prediction are done on GPU-accelerated nodes
389 in the cloud. See this notebook for details: <http://janclemenslab.org/das/tutorials/colab.html>.

390 Data sources

391 All data used for testing *DAS* were published previously. Sources for the original data sets, for
392 the data and annotations used for training and testing, and for the fitted models are listed in **Table 5**. Single-channel recordings of *Drosophila melanogaster* (strain OregonR) males courting fe-
393 males were taken from *Stern (2014)*. The multi-channel data from *Drosophila melanogaster* (strain
394 NM91) males courting females were recorded in a chamber tiled with nine microphones (*Coen*
395 *et al., 2014*) and was previously published in *Clemens et al. (2018)*. Annotations for fly song were
396 seeded with *FlySongSegmenter (Arthur et al., 2013; Coen et al., 2014)* and then manually corrected.
397 Recordings of mouse USVs were previously published in *Ivanenko et al. (2020)*. The USVs were
398 manually labelled using the *DAS* graphical interface. Marmoset recordings were taken from the
399 data published with *Landman et al. (2020)*. Since we required more precise delineation of the syl-
400 lable boundaries than was provided in the published annotations, we manually fixed annotations
401 for a subset of the data that then was used for training and testing. The network was trained
402 and tested on a subset of 4 vocalization types (eks/trills/tsiks/twitter, N=603/828/115/868). The
403 remaining vocalization types were excluded since they had 60 or fewer instances in our subset. To
404 test *DAS* on bird songs, we used a publicly available, hand-labeled collection of song from four male
405 Bengalese finches (*Nicholson et al., 2017*) and recordings of female-directed song from a male Ze-
406 bra finch from *Goffinet et al. (2021)*. For training and testing the Zebra finch network, we manually
407 labelled 473 syllables of 6 types (320 seconds of recordings).

409 DAS network

410 *DAS* is implemented in Keras (*Chollet et al., 2015*) and Tensorflow (*Abadi et al., 2016*). At its core,
411 *DAS* consists of a stack of temporal convolutional blocks, which transform an input sequence of
412 audio data into an output sequence of labels.

413 Inputs

414 *DAS* takes as input raw, single or multi-channel audio. Pre-processing of the audio using a Wavelet
415 or short-time Fourier transform is optional and integrated into the network. *DAS* processes audio
416 in overlapping chunks (**Figure 1-Figure Supplement 1A-D**). The chunking accelerates annotations
417 since multiple samples are annotated in a single computational step. Edge effects are avoided by
418 processing overlapping chunks and by discarding a number of samples at the chunk boundaries.
419 The overlap depends on the number of layers and the duration of filters in the network.

420 STFT frontend

The trainable short-term Fourier transform (STFT) frontend is an optional step and was implemented using kapre (*Choi et al., 2017*). Each frequency channel in the output of the frontend is the result of two, one-dimensional strided convolutions which are initialized with the real and the imaginary part of discrete Fourier transform kernels:

$$x(i, f) = \sum_{\tau=0}^{T-1} x(i * s + \tau)[\cos(2\pi f \tau / T) - i \sin(2\pi f \tau / T)]$$

$$y(i, f) = \log_{10}(\sqrt{\Re x(i, f)^2 + \Im x(i, f)^2})$$

421 where f is the frequency, s is the stride, and T is the filter duration. The stride results in down-
422 sampling of the input by a factor s .

423 The STFT kernels are optimized with all other parameters of the network during training. The
424 STFT frontend was used for mammal and bird signals, but not for fly song. In the mammal and
425 bird networks, we used 33 STFT filter pairs with a duration $T = 64$ samples and a stride $s = 16$ sam-
426 ples. For mouse and bird song, the STFT frontend sped up training and inference, and had a small
427 positive impact on performance. For fly song, the STFT frontend tended to reduce performance
428 and was omitted.

429 Temporal convolutional blocks

430 Temporal convolutional network (TCN) blocks are central to *DAS* and produce a task-optimized
431 hierarchical representation of sound features at high temporal resolution (*Bai et al., 2018*). Each
432 TCN block consists of a stack of so-called residual blocks (*Figure 1–Figure Supplement 1E*) (*He et al.,*
433 *2016*):

434 A *dilated convolutional layer* filters the input with a number of kernels of a given duration: $y_i(t) =$
435 $\sum_{\tau, \gamma} k_i(\tau, \gamma) * x(t - a\tau, \gamma)$, where $k_i(\tau, \gamma)$ is the i th kernel, $x(t, \gamma)$ is the input on channel γ at time t , $y_i(t)$
436 the output and a the gap or skip size (*Yu and Koltun, 2016*). In old-fashioned convolution $a = 1$.
437 Increasing a allows the kernel to span a larger range of inputs with the same number of parameters
438 and without a loss of output resolution. The number of parameters is further reduced for networks
439 processing multi-channel audio, by using *separable* dilated convolutions in the first two TCN blocks
440 (*Mamalet and Garcia, 2012*). In separable convolutions, the full two-dimensional $k(\tau, \gamma)$ convolution
441 over times and channels is decomposed into two one-dimensional convolutions. First, a temporal
442 convolution, $k^t(\tau, 1)$, is applied to each channel and then N channel convolutions, $k^y(1, \gamma)$, combine
443 information across channels. Instead of $\tau \times \gamma$ parameters, the separable convolution only requires
444 $\tau + N \times \gamma$ parameters. Note that each temporal convolution is applied to each channel, leading to a
445 sharing of filter parameters across channels. This makes explicit the intuition that some operations
446 should be applied to all channels equally. We also tested an alternative implementation, in which
447 individual channels were first processed separately by a single-channel TCN, the outputs of the TCN
448 blocks for each channel were concatenated, and then fed into a stack of standard TCNs with full
449 two-dimensional convolutions. While this architecture slightly increased performance it was also
450 much slower and we therefore chose the architecture with separable convolutions. Architecture
451 choice ultimately depends on speed and performance requirements of the annotation task.

452 A *rectifying linear unit* transmits only the positive inputs from the dilated convolutional layer by
453 setting all negative inputs to 0: $y_i = \max(0, y_i)$.

454 A *normalization layer* rescales the inputs to have a maximum absolute value close to 1: $y_i / (\max(|y_i|) +$
455 $10^{-5})$.

456 The output of the residual block, $z(t)$, is then routed to two targets: First, it is added to the input:
457 $o(t) = x(t) + z(t)$ and fed into subsequent layers. Second, via a so-called skip connection, the outputs
458 of all residual blocks are linearly combined to produce the network's final output (*van den Oord
et al., 2016*).

460 A single TCN block is composed of a stack of five residual blocks. Within a stack, the skip size
461 a doubles — from 1 in the first to $2^5 = 16$ in the final residual block of a stack. This exponential
462 increase in the span of the filter $\tau * a$ allows the TCN block to produce a hierarchical representation
463 of its inputs, from relatively low-level features on short timescales in early stacks to more derived
464 features on longer timescales in late stacks. Finally, a full network consists of a stack of 2 to 4 TCN
465 blocks, which extract ever more derived features (*Figure 1–Figure Supplement 1A-D*).

466 **Output**

467 The network returns a set confidence scores—one for each song type (and for no song)—for each
468 sample from a linear combination of the output of each residual block in the network, by using a
469 single dense layer and a softmax activation function. Re-using information from all blocks via so-
470 called skip connections ensures that downstream layers can discard information from upstream
471 layers and facilitates the generation of specialized higher-order presentations. If the input record-
472 ing got downsampled by a STFT frontend, a final upsampling layer restores the confidence scores
473 to the original audio rate by repeating values. The parameters of all used networks are listed in
474 *Table 4*.

475 **Choice of structural network parameters**

476 *DAS* performance is relatively robust to the choice of structural network parameters like filter du-
477 ration and number, or network depth (*Figure 4–Figure Supplement 5*). The networks tested here
478 are good starting points for adapting *DAS* to your own data (*Table 4*). In our experience, a network
479 with 32 filters, filter duration 32 samples, 3 TCN blocks, and a chunk duration of 2048 samples will
480 produce good results for most signals. A STFT-downsampling layer with 32 frequency band and
481 16x downsampling should be included for most signals except when the signals have a pulsatile
482 character. These parameters have been set as defaults when creating a new *DAS* network. Given
483 that *DAS* trains quickly (*Figure 4E*), network structure can be optimized by training *DAS* networks
484 over a grid of structural parameters, for instance to find the simplest network (in terms of the num-
485 ber of filters and TCN blocks) that saturates performance but has the shortest latency. We here
486 provide additional guidelines for choosing a network’s key structural parameters:

487 **The chunk duration** corresponds to the length of audio the network processes in one step and
488 constitutes an upper bound for the context available to the network. Choose chunks suffi-
489 ciently long so that the network has access to key features of your signal. For instance, for fly song, we
490 ensured that a single chunk encompasses several pulses in a train, so the network can learn to
491 detect song pulses based on their regular occurrence in trains. Longer chunks relative to this
492 timescale can reduce short false positive detections, for instance for fly sine song and for bird song.
493 Given that increasing chunk duration does not increase the number of parameters for training, we
494 recommend using long chunks unless low latency is of essence (see below).

495 **Downsampling/STFT** weakly affects performance but strongly accelerates convergence during
496 training. This is because A) the initialization with STFT filters is a good prior that reduces the number
497 of epochs it takes to learn the optimal filters, and B) the downsampling reduces the data bandwidth
498 and thereby the time it takes to finish one training epoch. The overall increase in performance
499 from adding the STFT layer is low because convolutional layers in the rest of the network can easily
500 replicate the computations of the STFT layer. For short pulsatile signals or signals with low sampling
501 rates, STFT and downsampling should be avoided since they can decrease performance due to the
502 loss of temporal resolution.

503 **The number of TCN blocks** controls the network’s depth. A deeper network can extract more
504 high-level features, though we found that even for the spectro-temporally complex song of Ben-
505 galese finches, deeper networks only weakly improved performance (*Figure 4–Figure Supplement 5*).

506 **Multi-channel audio** can be processed with multi-channel filters via full convolutions or with
507 shared channel-wise filters via time-channel separable convolutions. This can be set on a per-
508 TCN-block basis. We recommend to use separable convolutions in the first 1-2 layers, since basic

509 feature extraction is typically the same for each channel. Later layers can then have full multi-
510 channel filters to allow more complex combination of information across channels.

511 **Real-time performance** can be optimized by reducing networks complexity and chunk dura-
512 tion (*Figure 4–Figure Supplement 2*). We recommend starting with the default parameters sug-
513 gested above and then benchmarking latency. If required, latency can then be further reduced by
514 reducing chunk duration, the number and duration of filters, and the number of TCN blocks.

515 **Training**

516 Networks were trained using the categorical cross-entropy loss and the Adam optimizer (*Kingma*
517 and *Ba, 2015*) with a batch size of 32. Prediction targets were generated from fully annotated
518 recordings and one-hot-encoded: Segments were coded as binary vectors, with $y_i(t) = 1$ if a seg-
519 ment of type i occurred at time t , and $y_i(t) = 0$ otherwise. To encode uncertainty in the timing of fly
520 song pulses, the pulses were represented as Gaussian bumps with a standard deviation of 1.6 ms.
521 A “no song” type was set to $y_{\text{no song}}(t) = 1 - \sum_i y_i(t)$. That way, y corresponds to the probability
522 of finding any of the annotated song types or no song. For bird song, short gaps (6.25 ms, 200
523 samples at 32 kHz) were introduced between adjacent syllables to aid the detection of syllable on-
524 and offsets after inference. That way, syllable on- and offsets could be unequivocally detected as
525 changes from “no song” to any of the syllables. This reduced the amount of false positive on- and
526 offsets from switches in the label within a syllable.

527 Typically, multiple fully annotated recordings were combined in a data set. Each recording was
528 split 80:10:10 into a training, validation, and test set. The validation and test data were randomly
529 taken from the first, the middle or the last 10 % of each recording. Given the uneven temporal
530 distribution of call types in the marmoset recordings, we split the data 60:20:20 to ensure that
531 each call type was well represented in each split. For all networks, training was set to stop after
532 400 epochs or earlier if the validation loss was not reduced for at least 20 epochs. Training typically
533 stopped within 40-80 epochs depending on the dataset. The test set was only used after training,
534 for evaluating the model performance.

535 **Generation of annotations from the network output**

536 The confidence scores produced by the model correspond to the sample-wise probability for each
537 song type. To produce an annotation label for each sample, the confidence scores were further
538 processed to extract event times and syllable segments. In the resulting annotations, song types
539 are mutually exclusive, that is, each sample is labelled as containing a single song type even if song
540 types overlap.

541 Event times for event-like song types like fly pulse song were determined based on local maxima
542 in the confidence score, by setting a threshold value between 0 and 1 and a minimal distance
543 between subsequent peaks (using *peakutils* (*Negri and Vestri, 2017*)). For the pulse song of
544 flies, we set a minimal distance of 10 ms and a threshold of 0.7 for single channel data (*Figure 1*)
545 and 0.5 for multi-channel data (*Figure 1–Figure Supplement 3*).

546 For segment-like song types like fly sine song or the syllables of mouse, marmoset, and bird
547 song, we first transformed the sample-wise probability into a sequence of labels using *argmax_i y(i, t)*.
548 The resulting annotation of segments was then smoothed by filling short gaps (flies 20 ms, mice
549 10 ms, marmosets and birds 5 ms) and removing very short detections (flies 20 ms, mice 5 ms, mar-
550 mosets and birds 30 ms). These values were chosen based on the statistics of song found in the
551 training data. Syllable on- and offsets were detected as changes from no-song to song and song to
552 no-song, respectively. For bird and marmoset vocalizations, syllable labels were determined based
553 on a majority vote, by calculating the mode of the sample-wise labels for each detected syllable.

554 **Evaluation**

555 *DAS* was evaluated on segments of recordings that were not used during training.

556 Events

557 For events—fly pulse song, or the on- and offsets of segments—we matched each true event with
558 its nearest neighbor in the list of true events and counted as true positives only events within a
559 specified distance from a true event. For the pulse song of flies as well as for the onsets and off-
560 sets of mouse, marmoset, and bird syllables, this distance was set to 10 ms. Results were robust
561 to the specific choice of the distance threshold (*Figure 1–Figure Supplement 2A*). For the onsets
562 and offsets of fly sine song and of the marmoset vocalizations, we set this distance to 40 ms, since
563 these signals tended to fade in and out, making the delineation of exact boundaries difficult. False
564 positive events were counted if the distance from a detected event to the nearest true event ex-
565 ceeded the distance threshold or if another detected event was closer to each true event within
566 the distance threshold. If several detected pulses shared the same nearest true pulses, only the
567 nearest of those was taken as a true positive, while the remaining detections were matched with
568 other true pulses within the distance threshold or counted as false positives.

569 False negatives were counted as all true events without nearby detected events. For pulse,
570 pseudo true negative events were estimated as the number of tolerance distances (2x tolerance
571 distance) fitting into the recording, minus the number of pulses. These true negatives for pulse
572 do not influence precision, recall, and F1-scores and are only used to fill the confusion matrices in
573 *Figure 1D, I* and *Figure 1–Figure Supplement 3C, H*. Pulse and sine song were evaluated only up to
574 the time of copulation.

575 Matching segment labels

576 For songs with only one syllable type, we compared the predicted and true labels for each sample
577 to compute the confusion matrix (*Figure 1–Figure Supplement 1F, G*). In the case of multiple syllable
578 types, the mode of the true and predicted labels for the samples of each detected syllable were
579 compared. A true positive was counted if the mode of the true labels was the same for the samples
580 covered by the detected syllable. Using the true syllables as reference produces similar results
581 (*Figure 2–Figure Supplement 1E, F* and *Figure 3–Figure Supplement 1D, E*).

582 Performance scores

583 From the false negative (FN), false positive (FP), and true positive (TP) counts we extracted several
584 scores: Precision (P)—the fraction of true positive out of all detections $TP/(FP+TP)$ —and recall (R)—
585 the fraction of true positives out of all positives $TP/(TP+FN)$. The F1 score combines precision and
586 recall via their geometric mean: $2 \times P \times R / (P + R)$. For datasets with many different syllable types, we
587 also used as a summary measure of performance the accuracy—the fraction of correctly labelled
588 syllables: $(TP+TN)/(TP+TN+FP+FN)$. For comparison with other studies, we additionally provide the
589 error rate for the song of Bengalese finches, which is based on the Levenshtein edit distance and
590 corresponds to the minimal number of inserts, deletions, and substitutions required to transform
591 the sequence of true syllable labels into the sequence of predicted syllable labels normalized by
592 the length of the true sequence (*Koumura and Okanoya, 2016; Cohen et al., 2020*).

593 Temporal precision

594 The temporal precision for events (pulses, syllable onsets and offsets) was calculated as the median
595 absolute distance between all matched events.

596 Annotation errors for Bengalese finches

597 The network for annotating bird song was trained on all syllable types. We removed from the test
598 data one syllable type with only a single instance in the test set (which was correctly classified), be-
599 cause the performance could not be assessed reliably based on a single instance. We also excluded
600 as annotation error a syllable type that contained syllables of more than 6 distinct types.

601 **Estimation of signal-to-noise ratio from audio recordings**

602 To assess the robustness of annotation performance to noise, we assessed the recall of *DAS* for
603 epochs with different signal-to-noise ratios (SNRs) for the fly and the mouse networks. Because
604 of fundamental differences in the nature of the signals, SNR values were computed with different
605 methods and are therefore not directly comparable across species.

606 **Pulse**

607 Pulse waveforms were 20 ms long and centered on the peak of the pulse energy. The root-mean
608 square (RMS) amplitudes of the waveform margins (first and last 5 ms) and center (7.5-12.5 ms)
609 were taken as noise and signal, respectively. RMS is defined as $\sqrt{\sum_i x(i)^2}$. For multi-channel record-
610 ings, the pulse waveform from the channel with the highest center RMS was chosen to calculate
611 the SNR.

612 **Sine**

613 Signal was given by the RMS amplitudes of the recording during sine song. Noise is the RMS ampli-
614 tude in the 200 ms before and after each sine song, with a 10 ms buffer. For instance, if a sine song
615 ended at 1000 ms, the recording between 1010 and 1210 ms was taken as noise. From the 200 ms
616 noise, we excluded samples that were labelled as sine or pulse and included intervals between
617 pulses. For multi-channel recordings, the SNR was calculated for the channel with the largest sig-
618 nal amplitude.

619 **Mouse**

620 We assumed an additive noise model: $\sigma_{total}^2 = \sigma_{signal}^2 + \sigma_{noise}^2$. σ^2 is the squared signal averaged over a
621 window of 1 ms. Since noise variance changed little relative to the signal variance in our recordings,
622 we can assume constant noise over time to calculate the signal strength: $\sigma_{signal}^2 = \sigma_{total}^2 - \sum_t \sigma_{noise}^2$.
623 The sample-wise SNR is then given by $SNR(t) = \sigma_{signal}(t)^2 / \sum_t \sigma_{noise}^2$.

624 **Speed benchmarks**

625 Inference speed was assessed using throughput and latency. Throughput is the number of sam-
626 ples annotated per second and latency is the time it takes to annotate a single chunk. Throughput
627 and latency depend on the chunk duration—the duration of a recording snippet processed by the
628 network at once—and on the batch size—the number of chunks processed during one call. Larger
629 batches maximize throughput by more effectively exploiting parallel computation in modern CPUs
630 and GPUs and reducing overheads from data transfer to the GPU. This comes at the cost of higher
631 latency, since results are available only after all chunks in a batch have been processed. Using small
632 batch sizes and short chunks therefore reduces latency, since results are available earlier, but this
633 comes at the cost of reduced throughput because of overhead from data transfer or under-utilized
634 parallel compute resources. To assess throughput and latency, run times of `model.predict` were
635 assessed for batch sizes ranging from 1 to 1024 (log spaced) with 10 repetitions for each batch size
636 after an initial warmup run (*Figure 4–Figure Supplement 1A-L*). Results shown in the main text are
637 from a batch size corresponding to 1 s of recording for throughput (*Figure 4A*) and a batch size of 1
638 for latency (*Figure 4B*, see also *Figure 4–Figure Supplement 1*). For fly song, latency was optimized
639 by reducing the chunk size to 25.6 ms (*Figure 4–Figure Supplement 2*). Benchmarks were run on
640 Windows 10, Tensorflow 2.1, with the network either running on a CPU (Intel i7-7770, 3.6GHz) or
641 on a GPU (GTX1050 Ti 4GB RAM).

642 We also benchmarked the throughput of existing methods for comparison with *DAS* (*Table 2*).
643 Since neither of the methods considered are designed to be used in ways in which latency can
644 be fairly compared to that of *DAS*, we did not assess latency. The throughput values include all
645 pre-processing steps (like calculation of a spectrogram) and comparisons to *DAS* were done using
646 the same hardware (CPU for *FSS* and *USVSEG*, GPU for *TweetyNet* and *Oikarinen et al. (2019)*). The

647 throughput of *FSS* (*Arthur et al., 2013; Coen et al., 2014*) was tested using 400 seconds of single-
648 channel and 9-channel recordings in Matlab2019a. *USVSEG* (*Tachibana et al., 2020*) was tested on
649 a 72 second recording in Matlab2019a. *TweetyNet* (*Cohen et al., 2020*) was tested using a set of 4
650 recordings (total duration 35 seconds). Throughput for *TweetyNet* was given by the combined run-
651 times of the pre-processing steps (calculating of spectrograms from raw audio and saving them
652 as temporary files) and the inference steps (running the network on a GPU). For the previously
653 published network for annotating marmoset calls (*Oikarinen et al., 2019*), we relied on published
654 values for estimating throughput: A processing time of 8 minutes for a 60 minute recording corre-
655 sponds to a throughput of 7.5 s/s.

656 **Data economy**

657 For estimating the number of manual annotations required to train obtain accurate annotations,
658 we trained the networks using different fractions of the full training and validation sets (for instance,
659 0.001, 0.005, 0.01, 0.05, 0.1, 0.5, 1.0). Performance of all networks trained on the different subsets
660 was evaluated on the full test set. The number of manual annotations in each subset was deter-
661 mined after training from the training and validation sets. The number of annotations required
662 to exceed 90 % of the F1 score of a model trained on the full data sets was calculated based on a
663 lowess fit (*Cleveland, 1979*) to the data points (*Figure 4A, B*).

664 **Unsupervised classification**

665 Segmented signals were clustered using unsupervised methods described previously in *Clemens*
666 *et al. (2018)*, *Sainburg et al. (2020)*, and *Sangiamo et al. (2020)*. First, signals were pre-processed:
667 For fly song, pulse and sine waveforms of duration 15 ms were extracted from the recording,
668 aligned to their peak energy, normalized to unit norm, and adjusted for sign (see *Clemens et al.*
669 *(2018)* for details). For mouse, marmoset, and bird vocalizations, we adapted the procedures
670 described in *Sainburg et al. (2020)*: Noise was reduced in the bird song recordings using the
671 noisereduce package (<https://github.com/timsainb/noisereduce>). For mouse and marmoset vocal-
672 izations, noise reduction tended to blur the spectral contours and was omitted. Then, syllable
673 spectrograms were extracted from mel spectrograms of the recordings. The noise floor of the
674 spectrogram at each frequency was estimated as the median spectrogram over time and each
675 spectral band was then divided by the frequency-specific noise floor value. Finally, the spectro-
676 gram values were log-transformed and thresholded at 0 for mice and 2 for marmosets and birds
677 after visual inspection of the spectrograms to further remove background noise. To reduce differ-
678 ences in the duration of different syllables, all syllables were first log resized in time (scaling factor
679 8) and then padded with zeros to the duration of the longest syllable in the data set. Lastly, the fre-
680 quency axis of the spectrograms for mouse syllables were aligned to the peak frequency, to make
681 clustering robust to jitter in the frequency of the thin spectral contours (*Sangiamo et al., 2020*). The
682 peak frequency of each mouse syllable was calculated from its time-averaged spectrogram, and
683 only the 40 spectrogram frequencies around the peak frequency were retained.

684 The dimensionality of the pre-processed waveforms (fly) or spectrograms (mouse, marmoset,
685 birds) was then reduced to two using the UMAP method (*McInnes and Healy, 2018*) (*mindist*=0.5,
686 0.1 for marmosets to improve separation of clusters). Finally, signals were grouped using unsuper-
687 vised clustering. For the fly, marmoset, and bird signals, the UMAP distribution revealed distinct
688 groups of syllables and we used a density-based method to cluster the syllables (*Campello et al.*
689 *(2013)*, *min_samples*=10, *min_cluster_size*=20). For mouse USVs, no clusters were visible in
690 the UMAP distribution and density-based clustering failed to identify distinct groups of syllables.
691 Syllables were therefore split into 40 groups using k-means clustering.

692 **Open source software used**

- 693 • avgn https://github.com/timsainb/avgn_paper (*Sainburg et al., 2020*)
694 • hdbscan (*McInnes et al., 2017*)

- 695 • ipython (*Pérez and Granger, 2007*)
- 696 • jupyter (*Kluyver et al., 2016*)
- 697 • kapre (*Choi et al., 2017*)
- 698 • keras (*Chollet et al., 2015*)
- 699 • keras-tcn <https://github.com/philipperemy/keras-tcn>
- 700 • librosa (*McFee et al., 2015*)
- 701 • matplotlib (*Hunter, 2007*)
- 702 • noisereduce <https://github.com/timsainb/noisereduce>
- 703 • numpy (*Harris et al., 2020*)
- 704 • pandas (*McKinney, 2010*)
- 705 • peakutils (*Negri and Vestri, 2017*)
- 706 • scikit-learn (*Pedregosa et al., 2011*)
- 707 • scipy (*Virtanen et al., 2020*)
- 708 • seaborn (*Waskom et al., 2017*)
- 709 • snakemake (*Köster and Rahmann, 2018*)
- 710 • tensorflow (*Abadi et al., 2016*)
- 711 • UMAP (*McInnes and Healy, 2018*)
- 712 • zarr (*Miles et al., 2020*)
- 713 • xarray (*Hoyer and Hamman, 2017*)

714 Acknowledgments

715 We thank Kurt Hammerschmidt for providing mouse data prior to publication. We thank Mala
716 Murthy, David Stern, and all members of the Clemens lab for feedback on the manuscript.
717 This work was supported by the DFG through grants 329518246 (Emmy Noether) and 430158535
718 (SPP2205) and by the European Research Council (ERC) under the European Union’s Horizon 2020
719 research and innovation programme (Starting Grant agreement No. 851210 NeuSoSen).

Table 1. Precision, recall, and temporal error of DAS.

Precision and recall values are sample-wise for all except fly pulse song, for which it is event-wise. The number of classes includes the "no song" class. (p) Pulse, (s) Sine.

Species	Trained	Classes	Threshold	Precision [%]	Recall [%]	Temporal error [ms]
Fly single channel	Pulse (p) & sine (s)	3	0.7	97/92 (p/s)	96/98 (p/s)	0.3/12 (p/s)
Fly multi channel	Pulse (p)	2	0.5	98	94	0.3
Fly multi channel	Sine (s)	2	0.5	97	93	8
Mouse	Female	2	0.5	98	99	0.3
Marmoset	5 male-female pairs	5	0.5	85	91	4.4
Bengalese finch	4 males	49 (38 in test set)	0.5	97	97	0.3
Zebra finch	1 males	7	0.5	98	97	1.2

Table 2. Comparison to alternative methods.

Methods used for comparisons:

(1) *Arthur et al. (2013)*, (2) *Tachibana et al. (2020)*, (3) *Oikarinen et al. (2019)*, (4) *Cohen et al. (2020)*.

(A,B) DAS was trained on 1825/15970 syllables which contained 4/7 of the call types from *Oikarinen et al. (2019)*.

(B) The method by *Oikarinen et al. (2019)* produces an annotation every 50 ms of the recording - since the on/offset can occur anywhere within the 50 ms, the expected error of the method by *Oikarinen et al. (2019)* is at least 12.5 ms.

(C) The method by *Oikarinen et al. (2019)* annotates 60 mins of recordings in 8 minutes.

(D) Throughput assessed on the CPU, since the methods by *Arthur et al. (2013)* and *Tachibana et al. (2020)* do not run on a GPU.

(E) Throughput assessed on the GPU. The methods by *Cohen et al. (2020)* and *Oikarinen et al. (2019)* use a GPU.

Species	Precision [%]		Recall [%]		Jitter [ms]		Throughput [s/s]	
	DAS	other	DAS	other	DAS	other	DAS	other
Fly single (1)	97/92 (p/s)	99/91	96/98 (p/s)	87/91	0.3/12 (p/s)	0.1/22	15	4(D)
Fly multi (1)	98	99	94	92	0.3	0.1	8 (p)	0.4 (p+s)(D)
Fly multi (1)	97	95	93	93	8.0	15.0	8 (s)	0.4 (p+s)(D)
Mouse (2)	98	98	99	99	0.3	0.4	12	4(D)
Marmoset (3)	96	85(A)	92	77(A)	4.4	12.5(B)	82	7.5 (C,E)
Bengalese finch (4)	99	99	99	99	0.3	1.1	15	5 (E)
Zebra finch (4)	100	100	100	100	1.3	2.0	18	5 (E)

References

- 720
 721 Abadi M, Barham P, Chen J, Chen Z, Davis A, Dean J, Devin M, Ghemawat S, Irving G, Isard M, Kudlur M, Leven-
 722 berg J, Monga R, Moore S, Murray DG, Steiner B, Tucker P, Vasudevan V, Warden P, Wicke M, et al. TensorFlow:
 723 A System for Large-Scale Machine Learning. In: OSDI'16, USA: USENIX Association; 2016. p. 265–283.
- 724 Arthur BJ, Ding Y, Sosale M, Khalif F, Kim E, Waddell P, Turaga SC, Stern DL. SongExplorer: A deep learning
 725 workflow for discovery and segmentation of animal acoustic communication signals. bioRxiv. 2021; doi:
 726 [10.1101/2021.03.26.437280](https://doi.org/10.1101/2021.03.26.437280).
- 727 Arthur BJ, Sunayama-Morita T, Coen P, Murthy M, Stern DL. Multi-channel acoustic recording and automated
 728 analysis of *Drosophila* courtship songs. BMC Biology. 2013; 11(1):11.

Table 3. Comparison to human annotators for fly song.

See also *Figure 1E, J*.

Annotator	Sine recall [%]	Sine precision [%]	Pulse recall [%]	Pulse precision [%]
Human A	89	98	99	93
Human B	93	91	98	88
FSS	91	91	87	99
DAS	98	92	96	97

Table 4. Structural parameters of the tested networks.

Species	Rate [kHz]	Chunk [samples]	Channels	STFT downsample	Separable conv.	TCN stacks	Kernel size [samples]	Kernels
Fly single channel	10.0	4096	1	-	-	3	32	32
Fly multi channel (pulse)	10.0	2048	9	-	TCN stacks 1+2	4	32	32
Fly multi channel (sine)	10.0	2048	9	-	TCN stacks 1+2	4	32	32
Mouse	300.0	8192	1	16x	-	2	16	32
Marmoset	44.1	8192	1	16x	-	2	16	32
Bengale finch	32.0	1024	1	16x	-	4	32	64
Zebra finch	32.0	2048	1	16x	-	4	32	64

Table 5. Sources of all data used for testing DAS.

"Data" refers to the data used for DAS and to annotations that were either created from scratch or modified from the original annotations (deposited under <https://data.goettingen-research-online.de/dataverse/das>).

"Original data" refers to the recordings and annotations deposited by the authors of the original publication.

"Model" points to a directory with the model files as well as a small test data set and demo code for running the model (deposited under <https://github.com/janclemenslab/das-menagerie>).

Species	Reference	Data and model repositories
Fly single channel	<i>Stern (2014)</i>	data: https://doi.org/10.25625/TP4ODR
		original data: https://www.janelia.org/lab/stern-lab/tools-reagents-data
		model: https://github.com/janclemenslab/das-menagerie/dmel_single
Fly multi channel	<i>Clemens et al. (2018)</i>	data: https://doi.org/10.25625/8KAKHJ
		model: https://github.com/janclemenslab/das-menagerie/dmel_multi
Mouse	<i>Ivanenko et al. (2020)</i>	data: https://doi.org/10.25625/VVSKCH
		original data: https://data.donders.ru.nl/collections/di/dcn/DSC_620840_0003_891
		model: https://github.com/janclemenslab/das-menagerie/mouse
Marmoset	<i>Landman et al. (2020)</i>	data: https://doi.org/10.25625/DYG3KV
		original data: https://osf.io/q4bm3/
		model: https://github.com/janclemenslab/das-menagerie/marmoset
Bengalese finch	<i>Nicholson et al. (2017)</i>	data: https://doi.org/10.25625/ENKMJS
		original data: https://doi.org/10.6084/m9.figshare.4805749.v6
		model: https://github.com/janclemenslab/das-menagerie/bengalese_finch
Zebra finch	<i>Goffinet et al. (2021)</i>	data: https://doi.org/10.25625/ZXJJY
		original data: https://research.repository.duke.edu/concern/datasets/9k41zf38g
		model: https://github.com/janclemenslab/das-menagerie/zebra_finch

- 729 **Bai S**, Kolter JZ, Koltun V. An Empirical Evaluation of Generic Convolutional and Recurrent Networks for Se-
730 quence Modeling. ArXiv. 2018; abs/1803.01271.
- 731 **Baker C**, Clemens J, Murthy M. Acoustic Pattern Recognition and Courtship Songs: Insights from Insects. Annual
732 Review of Neuroscience. 2019; 42(1):129–147. doi: 10.1146/annurev-neuro-080317-061839.
- 733 **Bath DE**, Stowers JR, Hörmann D, Poehlmann A, Dickson BJ, Straw AD. FlyMAD: rapid thermogenetic control of
734 neuronal activity in freely walking *Drosophila*. Nature methods. 2014 May; .
- 735 **Behr O**, von Helversen O. Bat serenades—complex courtship songs of the sac-winged bat (*Saccopteryx bilin-*
736 *eata*). Behavioral Ecology and Sociobiology. 2004 Jun; 56(2):106–115.
- 737 **Benichov JI**, Vallentin D. Inhibition within a premotor circuit controls the timing of vocal turn-taking in zebra
738 finches. Nature communications. 2020 Jan; 11(1):1–10.
- 739 **Bennet-Clark HC**. Size and scale effects as constraints in insect sound communication. Philosophical Transac-
740 tions of the Royal Society B: Biological Sciences. 1998 Mar; 353(1367):407–419.
- 741 **Calhoun AJ**, Pillow JW, Murthy M. Unsupervised identification of the internal states that shape natural behavior.
742 Nature neuroscience. 2019 Nov; 16:1–10.
- 743 **Campello RJGB**, Moulavi D, Sander J. Density-Based Clustering Based on Hierarchical Density Estimates. In:
744 *Advances in Knowledge Discovery and Data Mining* Berlin, Heidelberg: Springer, Berlin, Heidelberg; 2013.p.
745 160–172.
- 746 **Cäsar C**, Zuberbühler K, Young RJ, Byrne RW. Titi monkey call sequences vary with predator location and type.
747 Biology Letters. 2013 Oct; 9(5):20130535.
- 748 **Cator LJ**, Arthur BJ, Harrington LC, Hoy R. Harmonic Convergence in the Love Songs of the Dengue Vector
749 Mosquito. Science. 2009; 323(5917):1166541+–1166541.
- 750 **Chaverri G**, Gillam EH, Kunz TH. A call-and-response system facilitates group cohesion among disc-winged
751 bats. Behavioral Ecology. 2012 Nov; 24(2):481–487.
- 752 **Chen X**, He K. Exploring Simple Siamese Representation Learning. arXiv:201110566 [cs]. 2020 Nov; .
- 753 **Choi K**, Joo D, Kim J. Kapre: On-GPU Audio Preprocessing Layers for a Quick Implementation of Deep Neural
754 Network Models with Keras. ArXiv. 2017; abs/1706.05781.
- 755 **Chollet F**, et al., Keras; 2015. <https://keras.io>.
- 756 **Clay Z**, Smith CL, Blumstein DT. Food-associated vocalizations in mammals and birds: what do these calls really
757 mean? Animal Behaviour. 2012 Feb; 83(2):323–330.
- 758 **Clemens J**, Coen P, Roemschied FA, Pereira TD, Mazumder D, Aldarondo DE, Pacheco DA, Murthy M. Discovery
759 of a New Song Mode in *Drosophila* Reveals Hidden Structure in the Sensory and Neural Drivers of Behavior.
760 Current biology. 2018 Aug; 28(15):2400–2412.e6.
- 761 **Clemens J**, Hennig RM. Computational principles underlying the recognition of acoustic signals in insects. Journal
762 of Computational Neuroscience. 2013 Aug; 35(1):75–85.
- 763 **Cleveland WS**. Robust Locally Weighted Regression and Smoothing Scatterplots. Journal of the American
764 Statistical Association. 1979; .
- 765 **Coen P**, Clemens J, Weinstein AJ, Pacheco DA, Deng Y, Murthy M. Dynamic sensory cues shape song structure
766 in *Drosophila*. Nature. 2014 Mar; 507(7491):233–237.
- 767 **Coen P**, Xie M, Clemens J, Murthy M. Sensorimotor Transformations Underlying Variability in Song Intensity
768 during *Drosophila* Courtship. Neuron. 2016 Feb; 89(3):629–644.
- 769 **Coffey KR**, Marx RG, Neumaier JF. DeepSqueak: a deep learning-based system for detection and analysis of
770 ultrasonic vocalizations. Neuropsychopharmacology. 2019 Jan; 231:1–10.
- 771 **Cohen Y**, Nicholson D, Gardner TJ. TweetyNet: A Neural Network That Enables High-Throughput, Automated
772 Annotation of Birdsong. bioRxiv. 2020 Aug; 14(8):2020.08.28.272088. doi: 10.1101/2020.08.28.272088.
- 773 **Deutsch D**, Clemens J, Thibierge SY, Guan G, Murthy M. Shared Song Detector Neurons in *Drosophila* Male and
774 Female Brains Drive Sex-Specific Behaviors. Current biology. 2019 Oct; 29(19):3200–3215.e5.

- 775 Devlin J, Chang MW, Lee K, Toutanova K. BERT: Pre-training of Deep Bidirectional Transformers for Language
776 Understanding. In: *NAACL*; 2019. .
- 777 Ding Y, Berrocal A, Morita T, Longden KD, Stern D. Natural courtship song variation caused by an intronic
778 retroelement in an ion channel gene. *Nature*. 2016; 536:329–332.
- 779 Ding Y, Lillvis JL, Cande J, Berman GJ, Arthur BJ, Long X, Xu M, Dickson BJ, Stern DL. Neural Evolution of Context-
780 Dependent Fly Song. *Current biology*. 2019 Mar; 0(0).
- 781 Fitch TW, Neubauer J, Herzel H. Calls out of chaos: the adaptive significance of nonlinear phenomena in
782 mammalian vocal production. *Animal Behaviour*. 2002; 63(3):407–418.
- 783 Fortune ES, Rodríguez C, Li D, Ball GF, Coleman MJ. Neural mechanisms for the coordination of duet singing
784 in wrens. *Science*. 2011 Nov; 334(6056):666–670.
- 785 Gerhardt CH, Huber F. Acoustic Communication in Insects and Anurans. University Of Chicago Press; 2002.
- 786 Goffinet J, Brudner S, Mooney R, Pearson J. Low-Dimensional Learned Feature Spaces Quantify Individual and
787 Group Differences in Vocal Repertoires. *eLife*. 2021 May; 10:e67855. doi: [10.7554/eLife.67855](https://doi.org/10.7554/eLife.67855).
- 788 Graves A, Jaitly N. Towards End-To-End Speech Recognition with Recurrent Neural Networks. In: *International
789 Conference on Machine Learning PMLR*; 2014.p. 1764–1772.
- 790 Graving JM, Chae D, Naik H, Li L, Koger B, Costelloe BR, Couzin ID. DeepPoseKit, a software toolkit for fast and
791 robust animal pose estimation using deep learning. *eLife*. 2019 Oct; 8:18.
- 792 Guirguis K, Schorn C, Guntoro A, Abdulatif S, Yang B. SELD-TCN: Sound Event Localization & Detection via
793 Temporal Convolutional Networks. 2020 28th European Signal Processing Conference (EUSIPCO). 2021; p.
794 16–20.
- 795 Haack B, Markl H, Ehret G. Sound communication between parents and offspring. In: Willott JF, editor. *The
796 auditory psychobiology of the mouse*. Springfield (Illinois): C. C. Thomas; 1983.p. 57–97.
- 797 Harris CR, Millman KJ, van der Walt SJ, Gommers R, Virtanen P, Cournapeau D, Wieser E, Taylor J, Berg S, Smith
798 NJ, Kern R, Picus M, Hoyer S, van Kerkwijk MH, Brett M, Haldane A, del Río JF, Wiebe M, Peterson P, Gérard-
799 Marchant P, et al. Array programming with NumPy. *Nature*. 2020; .
- 800 He K, Zhang X, Ren S, Sun J. Deep Residual Learning for Image Recognition. 2016 IEEE Conference on Computer
801 Vision and Pattern Recognition (CVPR). 2016; p. 770–778.
- 802 Holy TE, Guo Z. Ultrasonic Songs of Male Mice. *PLoS Biology*. 2005 Nov; 3(12):e386.
- 803 Hoyer S, Hamman J. xarray: N-D labeled arrays and datasets in Python. *Journal of Open Research Software*.
804 2017; 5(1). <http://doi.org/10.5334/jors.148>, doi: [10.5334/jors.148](https://doi.org/10.5334/jors.148).
- 805 Hunter JD. Matplotlib: A 2D graphics environment. *Computing in Science & Engineering*. 2007; 9(3):90–95. doi:
806 [10.1109/MCSE.2007.55](https://doi.org/10.1109/MCSE.2007.55).
- 807 Ivanenko A, Watkins P, van Gerven MAJ, Hammerschmidt K, Englitz B. Classifying Sex and Strain from Mouse
808 Ultrasonic Vocalizations Using Deep Learning. *PLOS Computational Biology*. 2020 Jun; 16(6):e1007918. doi:
809 [10.1371/journal.pcbi.1007918](https://doi.org/10.1371/journal.pcbi.1007918).
- 810 Janik VM, Slater PJB. Context-specific use suggests that bottlenose dolphin signature whistles are cohesion
811 calls. *Animal Behaviour*. 1998 Oct; 56(4):829–838.
- 812 Kingma DP, Ba J. Adam: A Method for Stochastic Optimization. *CoRR*. 2015; abs/1412.6980.
- 813 Kluyver T, Ragan-Kelley B, Pérez F, Granger B, Bussonnier M, Frederic J, Kelley K, Hamrick J, Grout J, Corlay
814 S, Ivanov P, Avila D, Abdalla S, Willing C, development team J. Jupyter Notebooks - a publishing format for
815 reproducible computational workflows. In: Loizides F, Scmidt B, editors. *Positioning and Power in Academic
816 Publishing: Players, Agents and Agendas* Netherlands: IOS Press; 2016. p. 87–90. <https://eprints.soton.ac.uk/403913/>.
- 818 Kollmorgen S, Hahnloser RHR, Mante V. Nearest neighbours reveal fast and slow components of motor learning.
819 *Nature*. 2020 Jan; 382:1–5.
- 820 Köster J, Rahmann S. Snakemake - a scalable bioinformatics workflow engine. *Bioinformatics*. 2018; 34 20:3600.

- 821 **Koumura T**, Okanoya K. Automatic Recognition of Element Classes and Boundaries in the Birdsong with Vari-
822 able Sequences. PLoS ONE. 2016 Jul; 11(7):e0159188.
- 823 **Krizhevsky A**, Sutskever I, Hinton GE. ImageNet Classification with Deep Convolutional Neural Networks. .
824 2012; p. 1097–1105.
- 825 **Landman R**, Sharma J, Hyman JB, Fanucci-Kiss A, Meisner O, Parmar S, Feng G, Desimone R. Close-Range
826 Vocal Interaction in the Common Marmoset (*Callithrix jacchus*). PLOS ONE. 2020 Apr; 15(4):e0227392. doi:
827 [10.1371/journal.pone.0227392](https://doi.org/10.1371/journal.pone.0227392).
- 828 **Lipkind D**, Marcus GF, Bemis DK, Sasahara K, Jacoby N, Takahasi M, Suzuki K, Fehér O, Ravbar P, Okanoya
829 K, Tchernichovski O. Stepwise acquisition of vocal combinatorial capacity in songbirds and human infants.
830 Nature. 2013 Jun; 498(7452):104–108.
- 831 **Long Ma**, Fee MS. Using temperature to analyse temporal dynamics in the songbird motor pathway. Nature.
832 2008 Nov; 456(7219):189–194.
- 833 **Mamalet F**, Garcia C. Simplifying ConvNets for Fast Learning. In: *Artificial Neural Networks and Machine Learning – ICANN 2012* Berlin, Heidelberg: Springer, Berlin, Heidelberg; 2012. p. 58–65.
- 835 **Mathis A**, Mamidanna P, Cury KM, Abe T, Murthy VN, Mathis MW, Bethge M. DeepLabCut: markerless pose
836 estimation of user-defined body parts with deep learning. Nature neuroscience. 2018 Aug; 20(Suppl. 2):1.
- 837 **Mathis A**, Yüksekgönül M, Rogers B, Bethge M, Mathis M. Pretraining boosts out-of-domain robustness for
838 pose estimation. 2021 IEEE Winter Conference on Applications of Computer Vision (WACV). 2021; p. 1858–
839 1867.
- 840 **McFee B**, Raffel C, Liang D, Ellis DP, McVicar M, Battenberg E, Nieto O. librosa: Audio and music signal analysis
841 in python. In: *Proceedings of the 14th python in science conference*, vol. 8; 2015. .
- 842 **McInnes L**, Healy J. UMAP: Uniform Manifold Approximation and Projection for Dimension Reduction. ArXiv.
843 2018; abs/1802.03426.
- 844 **McInnes L**, Healy J, Astels S. hdbscan: Hierarchical density based clustering. The Journal of Open Source
845 Software. 2017 mar; 2(11). <https://doi.org/10.21105%2Fjoss.00205>, doi: [10.21105/joss.00205](https://doi.org/10.21105/joss.00205).
- 846 **McKinney W**. Data Structures for Statistical Computing in Python. In: *Python in Science Conference*; 2010. .
- 847 **Miles A**, Kirkham J, Durant M, Bourbeau J, Onalan T, Hamman J, Patel Z, shikharsg, Rocklin M, raphael dussin,
848 Schut V, de Andrade ES, Abernathey R, Noyes C, sbalmer, pyup io bot, Tran T, Saalfeld S, Swaney J, Moore
849 J, et al., zarr-developers/zarr-python: v2.4.0. Zenodo; 2020. <https://doi.org/10.5281/zenodo.3773450>, doi:
850 [10.5281/zenodo.3773450](https://doi.org/10.5281/zenodo.3773450).
- 851 **Morley EL**, Jonsson T, Robert D. Auditory sensitivity, spatial dynamics, and amplitude of courtship song in
852 *Drosophila melanogaster*. The Journal of the Acoustical Society of America. 2018 Aug; 144(2):734–739.
- 853 **Negri LH**, Vestri C, lucashn/peakutils: v1.1.0. Zenodo; 2017. <https://doi.org/10.5281/zenodo.887917>, doi:
854 [10.5281/zenodo.887917](https://doi.org/10.5281/zenodo.887917).
- 855 **Neunuebel JP**, Taylor AL, Arthur BJ, Egnor SR. Female mice ultrasonically interact with males during courtship
856 displays. eLife. 2015 May; 4:e06203.
- 857 **Nicholson D**, Queen JE, Sober S. Bengalese Finch song repository. figshare; 2017. https://figshare.com/articles/dataset/Bengalese_Finch_song_repository/4805749/5, doi: [10.6084/m9.figshare.4805749.v5](https://doi.org/10.6084/m9.figshare.4805749.v5).
- 859 **Oikarinen T**, Srinivasan K, Meisner O, Hyman JB, Parmar S, Fanucci-Kiss A, Desimone R, Landman R, Feng
860 G. Deep Convolutional Network for Animal Sound Classification and Source Attribution Using Dual
861 Audio Recordings. The Journal of the Acoustical Society of America. 2019 Feb; 145(2):654–662. doi:
862 [10.1121/1.5087827](https://doi.org/10.1121/1.5087827).
- 863 **Okobi DE**, Banerjee A, Matheson AMM, Phelps SM, Long Ma. Motor cortical control of vocal interaction in
864 neotropical singing mice. Science. 2019 Mar; 363(6430):983–988.
- 865 **van den Oord A**, Dieleman S, Zen H, Simonyan K, Vinyals O, Graves A, Kalchbrenner N, Senior A, Kavukcuoglu
866 K. WaveNet: A Generative Model for Raw Audio. In: *SSW*; 2016. .

- 867 **Pedregosa F**, Varoquaux G, Gramfort A, Michel V, Thirion B, Grisel O, Blondel M, Prettenhofer P, Weiss R,
868 Dubourg V, Vanderplas J, Passos A, Cournapeau D, Brucher M, Perrot M, Duchesnay É. Scikit-learn: Machine
869 Learning in Python. *Journal of Machine Learning Research*. 2011; ;
- 870 **Pereira TD**, Aldarondo DE, Willmore L, Kislin M, Wang SSH, Murthy M, Shaevitz JW. Fast animal pose estimation
871 using deep neural networks. *Nature methods*. 2018 Dec; 16(1):1–125.
- 872 **Pérez F**, Granger B. IPython: A System for Interactive Scientific Computing. *Computing in Science & Engineering*.
873 2007; 9.
- 874 **Raghu M**, Zhang C, Kleinberg J, Bengio S. Transfusion: Understanding Transfer Learning for Medical Imaging.
875 In: *NeurIPS*; 2019. .
- 876 **Sainburg T**, Thielsk M, Gentner TQ. Finding, Visualizing, and Quantifying Latent Structure across Diverse
877 Animal Vocal Repertoires. *PLOS Computational Biology*. 2020 Oct; 16(10):e1008228. doi: [10.1371/journal.pcbi.1008228](https://doi.org/10.1371/journal.pcbi.1008228).
- 879 **Sangiamo DT**, Warren MR, Neunuebel JP. Ultrasonic signals associated with different types of social behavior
880 of mice. *Nature neuroscience*. 2020 Feb; 23(3):1–12.
- 881 **Srivastava KH**, Holmes CM, Vellema M, Pack AR, Elemans CPH, Nemenman I, Sober SJ. Motor control by pre-
882 cisely timed spike patterns. *Proceedings of the National Academy of Sciences*. 2017 Jan; 114(5):1171–1176.
- 883 **Stern DL**. Reported Drosophila courtship song rhythms are artifacts of data analysis. *BMC Biology*. 2014 Jun;
884 12(1):38.
- 885 **Stern DL**, Clemens J, Coen P, Calhoun AJ, Hogenesch JB, Arthur BJ, Murthy M. Experimental and statistical
886 reevaluation provides no evidence for Drosophila courtship song rhythms. *Proceedings of the National
887 Academy of Sciences*. 2017 Sep; 114(37):9978–9983.
- 888 **Stowers JR**, Hofbauer M, Bastien R, Griessner J, Higgins P, Farooqui S, Fischer RM, Nowikovsky K, Haubensak
889 W, Couzin ID, Tessmar-Raible K, Straw AD. Virtual reality for freely moving animals. *Nature methods*. 2017
890 Aug; 14(10):995–1002.
- 891 **Tabler JM**, Rigney MM, Berman GJ, Gopalakrishnan S, Heude E, Al-lami HA, Yannakoudakis BZ, Fitch RD, Carter
892 C, Vokes S, Liu KJ, Tajbakhsh S, Egnor SR, Wallingford JB. Cilia-mediated Hedgehog signaling controls form
893 and function in the mammalian larynx. *eLife*. 2017 Feb; 6:320.
- 894 **Tachibana RO**, Kanno K, Okabe S, Kobayashi KI, Okanoya K. USVSEG: A robust method for segmentation of
895 ultrasonic vocalizations in rodents. *PLoS ONE*. 2020 Feb; 15(2):e0228907.
- 896 **Tschida K**, Mooney R. The role of auditory feedback in vocal learning and maintenance. *Current Opinion in
897 Neurobiology*. 2012 Apr; 22(2):320–327.
- 898 **Van Segbroeck M**, Knoll AT, Levitt P, Narayanan S. MUPET—Mouse Ultrasonic Profile ExTraction: A Signal Pro-
899 cessing Tool for Rapid and Unsupervised Analysis of Ultrasonic Vocalizations. *Neuron*. 2017 May; 94(3):465–
900 485.e5.
- 901 **Virтанен P**, Gommers R, Oliphant TE, Haberland M, Reddy T, Cournapeau D, Burovski E, Peterson P, Weckesser
902 W, Bright J, van der Walt SJ, Brett M, Wilson J, Millman KJ, Mayorov N, Nelson ARJ, Jones E, Kern R, Larson E,
903 Carey CJ, et al. SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python. *Nature Methods*. 2020;
904 17:261–272. doi: 10.1038/s41592-019-0686-2.
- 905 **Warren MR**, Clein RS, Spurrier MS, Roth ED, Neunuebel JP. Ultrashort-range, high-frequency communication
906 by female mice shapes social interactions. *Scientific reports*. 2020 Feb; 10(1):1–14.
- 907 **Waskom M**, Botvinnik O, O’Kane D, Hobson P, Lukauskas S, Gemperline DC, Augspurger T, Halchenko Y, Cole JB,
908 Warmenhoven J, de Ruiter J, Pye C, Hoyer S, Vanderplas J, Villalba S, Kunter G, Quintero E, Bachant P, Martin
909 M, Meyer K, et al., mwaskom/seaborn: v0.8.1 (September 2017). Zenodo; 2017. <https://doi.org/10.5281/zenodo.883859>, doi: [10.5281/zenodo.883859](https://doi.org/10.5281/zenodo.883859).
- 911 **Weiss M**, Hultsch H, Adam I, Scharff C, Kipper S. The use of network analysis to study complex animal commu-
912 nication systems: a study on nightingale song. *Proceedings of the Royal Society B: Biological Sciences*. 2014
913 Jun; 281(1785):20140460.
- 914 **Yu F**, Koltun V. Multi-Scale Context Aggregation by Dilated Convolutions. *CoRR*. 2016; abs/1511.07122.

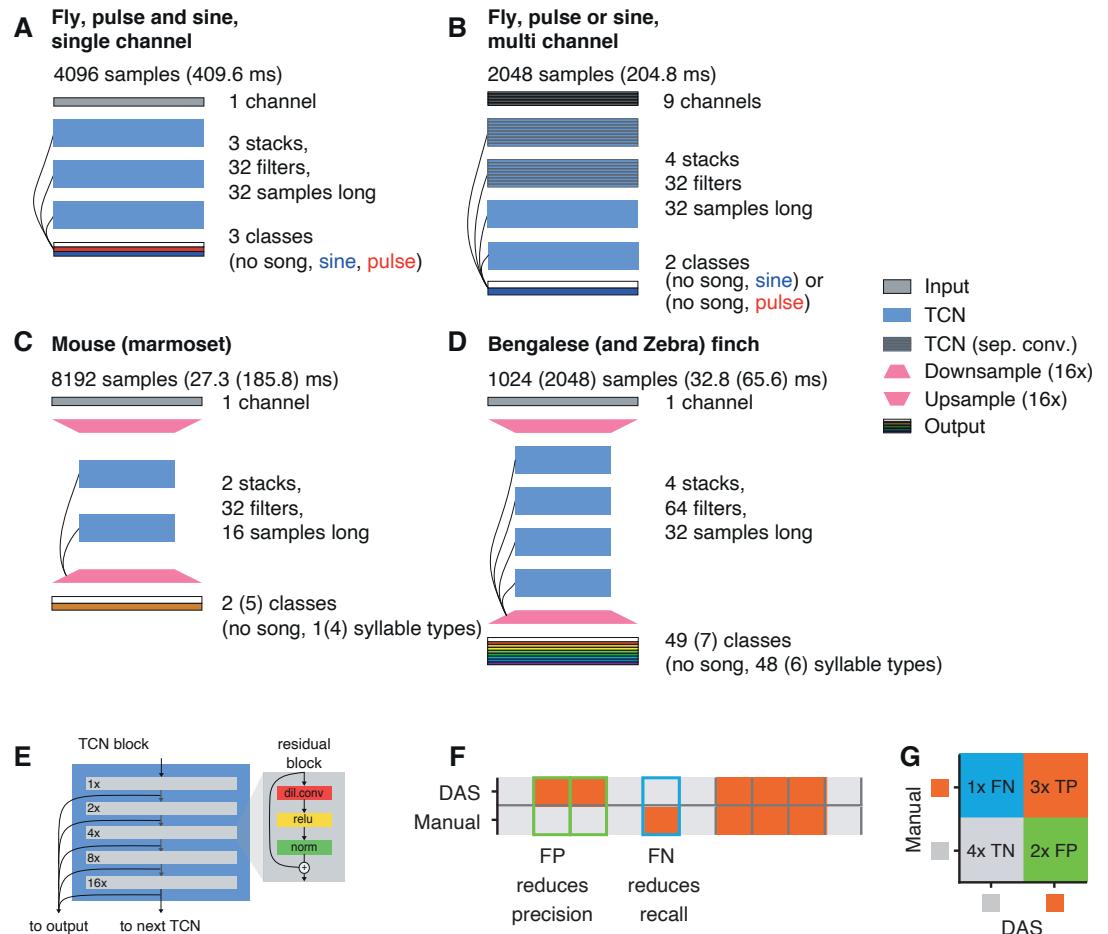


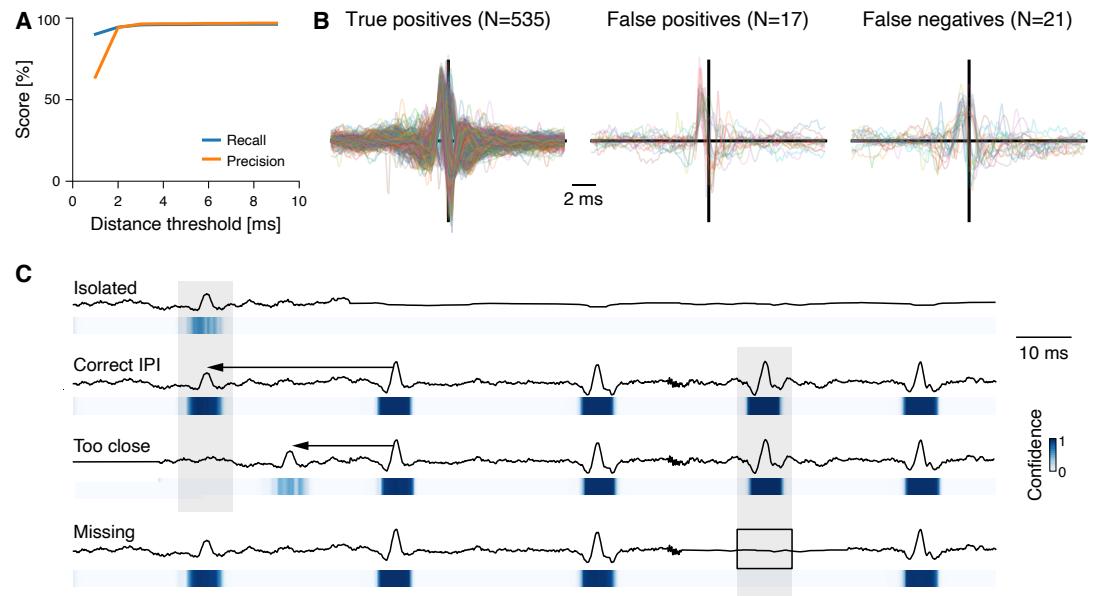
Figure 1-Figure supplement 1. DAS architecture and evaluation.

A-D Network architectures for annotating fly song from single (A) and multi-channel (B) recordings, mouse USVs and marmoset song (C), and bird song (Bengalese and Zebra finches) (D). See legend to the right. Each TCN stack consists of stacks of residual blocks shown in E. See **Table 4** for all network parameters.

E A TCN block (left) consists of a stack of 5 residual blocks (right). Residual blocks process the input with a sequence of dilated convolution, rectification (ReLU) and normalization. The output of this sequence of steps is then added to the input. In successive residual blocks, the dilation rate of the convolution filters doubles from 1x in the first to 16x in the last layer (see numbers to the left of each block). The output of the last residual block is passed as an input to the next TCN block in the network. In addition, the outputs of all residual blocks in a network are linearly combined to predict the song.

F Annotation performance is evaluated by comparing manual annotations (top) with labels produced by *DAS* (bottom). Grey indicates no song, orange song. True negatives (TN) and true positives (TP) are samples for which *DAS* matches the manual labels. False negatives (FNs) are samples for which the song was missed (blue frame) and reduce recall ($TP/(FN+TP)$). False positives (FP) correspond to samples that were falsely predicted as containing song (green frames) and reduce precision ($TP/(TP+FP)$).

G Precision and recall are calculated from a confusion matrix which tabulates TP (orange), TN (grey), FP (green), FN (blue). In the example, precision is $3/(3+2)$ and recall is $3/(1+3)$.



916 **Figure 1-Figure supplement 2. Performance and the role of context for annotating fly pulse song.**

A Recall (blue) and precision (orange) for fly pulse song for different distance thresholds. The distance threshold determines the maximal distance to a true pulse for a detected pulse to be a true positive.

B Waveforms of true positive (left), false positive (middle), and false negatives (right) pulses in fly song. Pulses were aligned to the peak, adjusted for sign, and their amplitude was normalized to have unit norm (see *Clemens et al. (2018)*).

C Waveforms (top) and confidence scores (bottom, see color bar) for pulses in different contexts. DAS exploits context effects to boost the detection of weak signals. An isolated ("Isolated", first row) weak pulse-like waveform is detected with low confidence, since similar waveforms often arise from noise. Manual annotators exploit context information – the fact that pulses often occur in trains at an interval of 40 ms – to annotate weak signals. DAS does the same – the same waveform is detected with much higher confidence due to the presence of a nearby pulse train ("Correct IPI", 2nd row). If the pulse is too close to another pulse ("Too close", 3rd row), it is likely noise and DAS detects it with lower confidence. Context effects do not affect strong signals. For instance, a missing pulse within a pulse train ("Missing", last row) does not reduce detection confidence of nearby pulses.

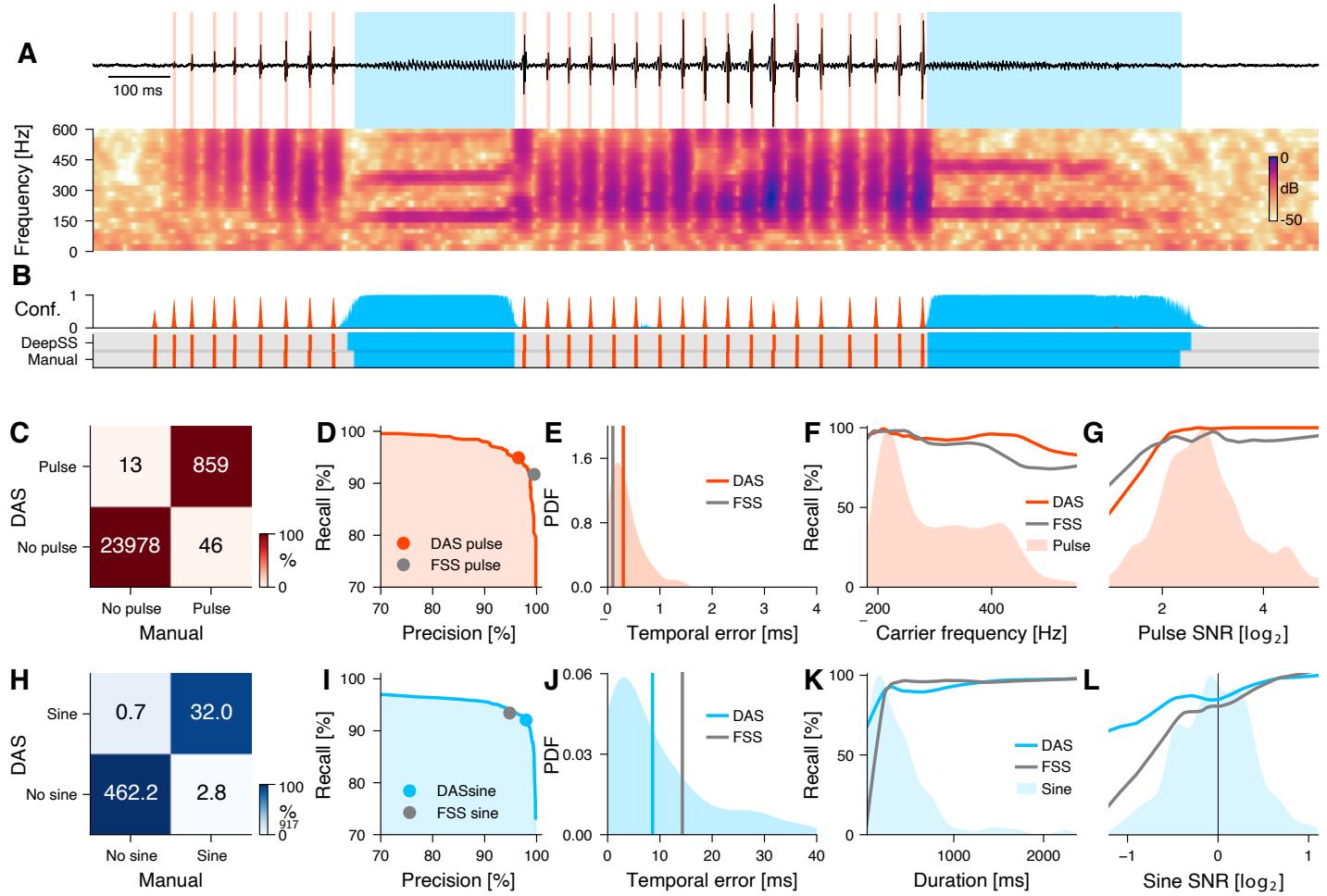


Figure 1–Figure supplement 3. @ plus2@ minus2@ @ plus1@ @ plus@ minus2@ **Performance for multi-channel recordings of fly courtship song.**

A Fly song (black) with manual annotation indicating sine (blue) and pulse (red). Traces (top) and spectrogram (bottom, see color bar) show data from the loudest of the nine audio channels.

B Confidence scores (top) for sine (blue) and pulse (red). The confidence is transformed into annotation labels (bottom) based on a confidence threshold (0.5 for sine and pulse). Ground truth (bottom) from manual annotations shown for comparison. *DAS* annotations were generated using separate networks for pulse or for sine.

C Confusion matrix for pulse from a test data set. Color indicates the percentage (see color bar) and text labels indicate number of pulses for each quadrant. All confusion matrices are normalized such that columns sum to 100 %. The concentration of values along the diagonal indicates high annotation performance.

D Precision-recall curve for pulse depicts the performance characteristics of *DAS* for different confidence threshold (from 0 to 1). Recall decreases and precision increases with the threshold. The closer the curve to the upper and right border, the better. The red circle corresponds to the performance of *DAS* for a threshold of 0.5. The grey circle depicts the performance of FlySongSegmenter (*FSS*).

E Probability density function (PDF) of temporal errors for all detected pulses (red shaded area), computed as the distance between each pulse annotated by *DAS* and its nearest manual pulse. Lines depict median temporal error for *DAS* (red line, 0.3 ms) and *FSS* (grey line, 0.1 ms).

F, G Recall of *DAS* (red line) and *FSS* (grey line) as a function of the pulse carrier frequency (F) and signal-to-noise ratio (SNR) (G). Red areas show the distributions of carrier frequencies (F) and SNRs (G) for all pulses.

H Same as in C but for sine. Color indicates the percentage (see color bar) and text labels indicate seconds of sine for each quadrant.

I Same as in D but for sine. The blue circle depicts the performance for the confidence threshold of 0.5 used in A.

J Distribution of temporal errors for all detected sine on- and offsets. Median temporal error is 9 ms for *DAS* (blue line) and 14 ms for *FSS* (grey line).

K, L Recall for *DAS* (blue line) and *FSS* (grey line) as a function of sine duration (K) and SNR (L). Blue shaded areas show the distributions of durations (K) and SNRs (L) for all sine songs. *DAS* outperforms *FSS* for sine songs with short durations and SNRs <1.0.

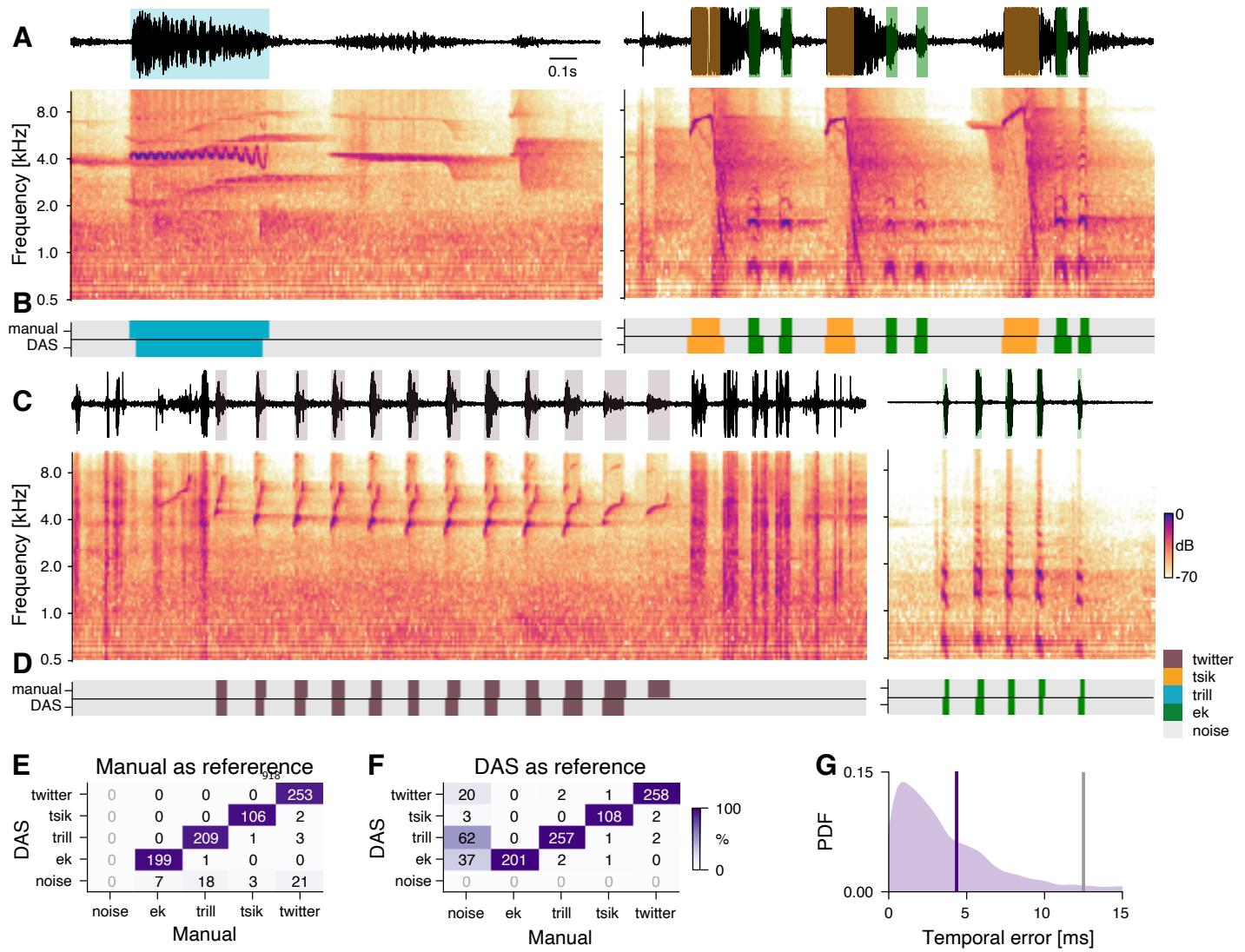


Figure 2-Figure supplement 1. Performance for marmoset vocalizations.

A, C Waveform (top) and spectrogram (bottom) of vocalizations from male and female marmosets. Shaded areas (top) show manual annotations of the different vocalization types, colored by type. Recordings are noisy (A, left), clipped (orange), and individual vocalization types are variable (C).

B, D DAS and manual annotation labels for the vocalizations types in the recordings in A and C (see color bar in C). DAS annotates the syllable boundaries and types with high accuracy. Note the false negative in D.

E, F Confusion matrices for the four vocalization types in the test set (see color bar), using the syllable boundaries from the manual annotations (E) or from DAS (F) as reference. Rows depict the probability with which DAS annotated each syllable as any of the four types in the test dataset. The type of most syllables were correctly annotated, resulting in the concentration of probability mass along the main diagonal. False positives and false negatives correspond to the first row in E and the first column in F, respectively. When using the true syllables for reference, there are no false positives (E, x="noise", grey labels) since all detections are positives. By contrast, when using the predicted syllables as reference, there are no true negatives (F, y="noise", grey labels), since all reference syllables are (true or false) positives.

G Distribution of temporal errors for the on- and offsets of all detected syllables (purple shaded area). The median temporal error is 4.4 ms for DAS (purple line) and 12.5 ms for the method by *Oikarinen et al. (2019)* developed to annotate marmoset calls (grey line).

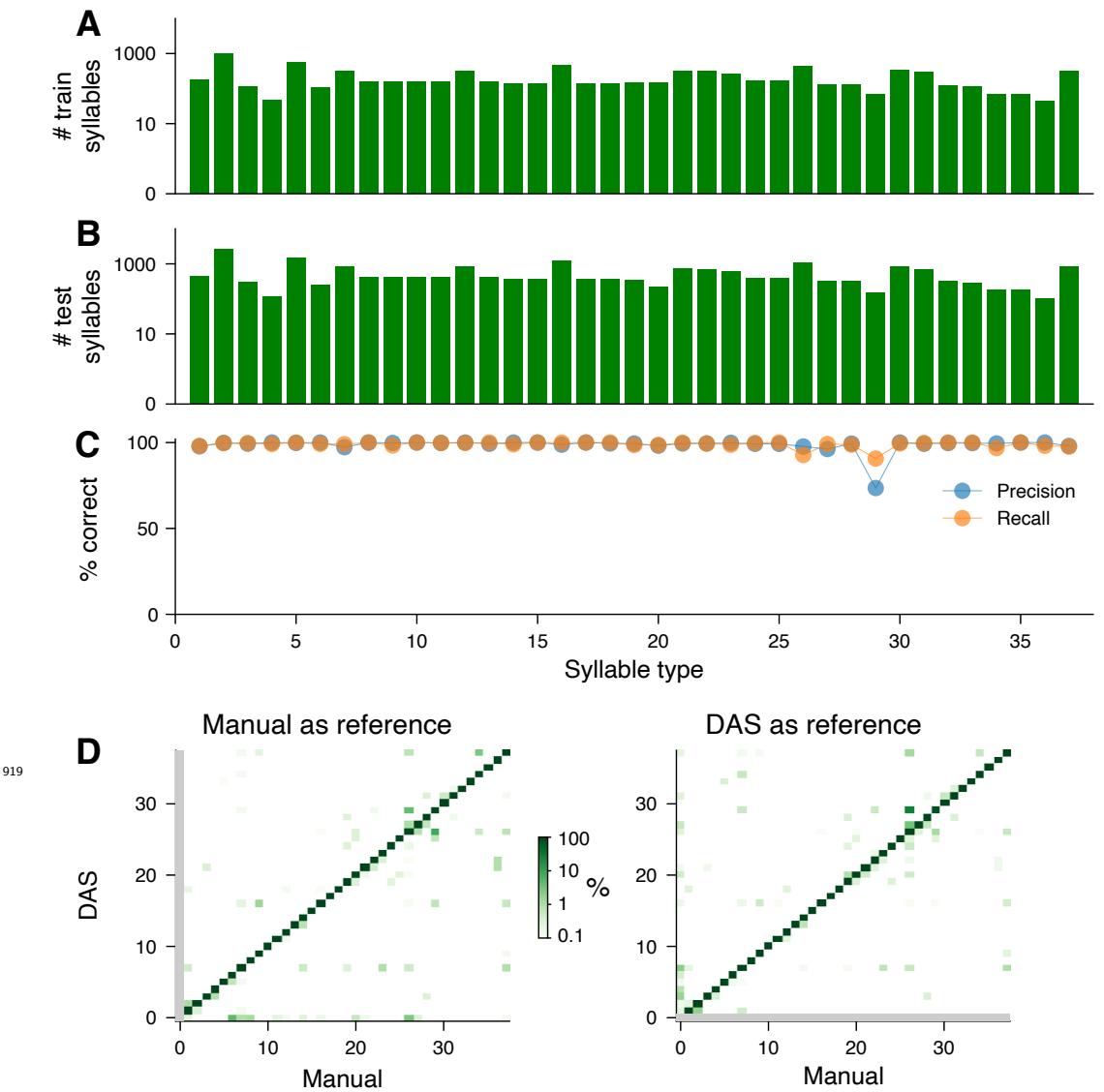
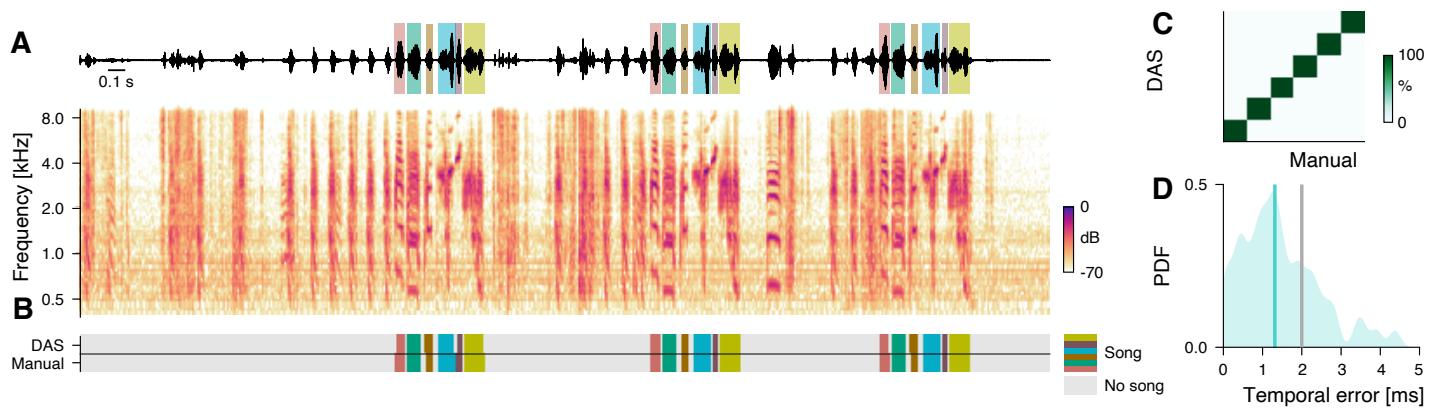


Figure 3-Figure supplement 1. Performance for the song of Bengalese finches.

A, B Number of syllables (log scaled) in the train (A) and the test (B) sets for each syllable type present in the test set.

C Precision (blue) and recall (orange) for each syllable type, computed from the confusion matrix in **Figure 3C**.

D Confusion matrices when using true (left, same as **Figure 3C**) and predicted (right) syllables as a reference. Colors were log-scaled to make the rare annotation errors more apparent (see color bar). The reference determines the syllable bounds and the syllable label is then given by the most frequent label found in the samples within the syllable bounds. When using the true syllables for reference, there are no false positives (left, $y=0$ (no song), grey line) since all detections are positives. By contrast, when using the predicted syllables as reference, there are no true negatives (right, $x=0$ (no song), grey line), since all reference syllables are (true or false) positives. The average false negative and positive rates are 0.3 % and 0.2 %, respectively.



920 **Figure 3-Figure supplement 2. Performance for the song of a Zebra finch.**

A Waveform (top) and spectrogram (bottom) of the song of a male Zebra finch. Shaded areas (top) show manual annotations of the six syllables of the male's motif, colored by syllable type.

B *DAS* and manual annotation labels for the six syllable types in the recording in A (see color bar). *DAS* accurately annotates the syllable boundaries and types.

C Confusion matrix for the six syllables in the test set (see color bar). Rows depict the probability with which *DAS* annotated each syllable as any of the six types in the test dataset. The type of 100 % (54/54) of the syllables were correctly annotated, resulting in the concentration of probability mass along the main diagonal.

D Distribution of temporal errors for the on- and offsets of all detected syllables in the test set (blue shaded area). The median temporal error is 1.3 ms for *DAS* (blue line) and 2 ms for *TweetyNet* (*Cohen et al., 2020*), a method developed to annotate bird song (grey line).

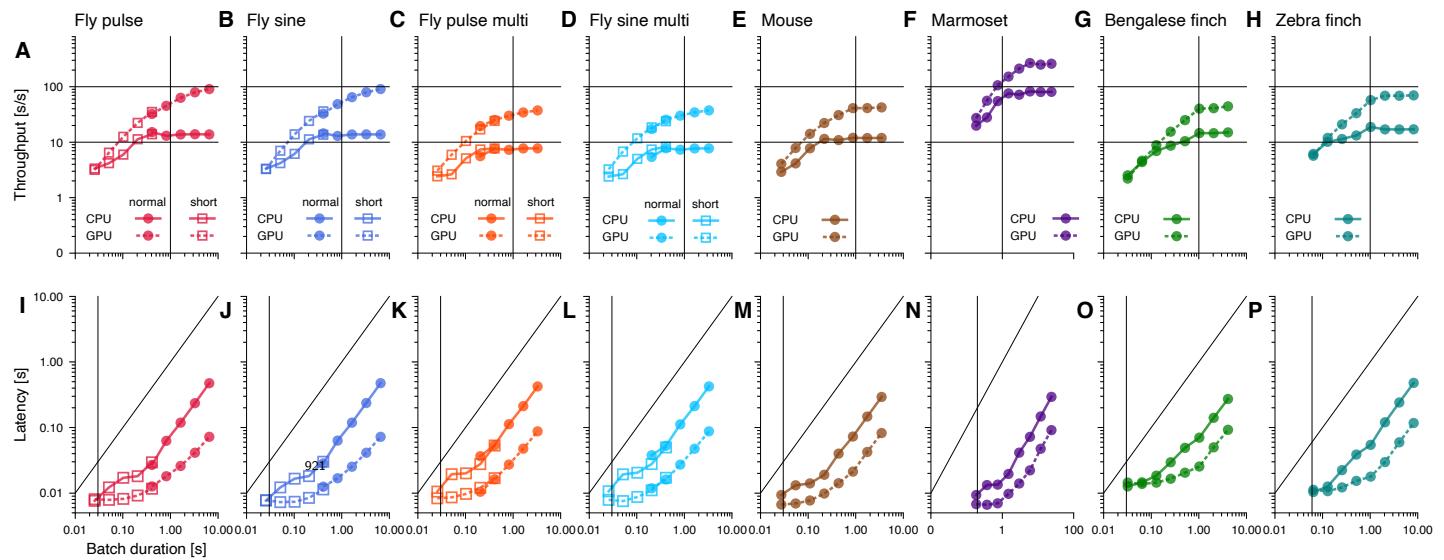


Figure 4-Figure supplement 1. Throughput and latency of inference.

Median throughput (A-H) and latency (I-P) across 10 inference runs for different song types. Batch duration is the product of chunk duration and batch size. Error bars are smaller than the marker size (see Figure 4A/B). Solid and dashed lines correspond to values when running *DAS* on a CPU and GPU, respectively. Squares and circles in the plots for fly song correspond to networks with short and long chunk durations, respectively (see Figure 4-Figure Supplement 2). In A-H, horizontal lines mark 10x and 100x realtime and vertical lines mark the batch duration used in Figure 4A. In I-P, the vertical line marks the batch duration used in Figure 4B. Differences in the throughput and latency between species arise from differences in the sample rate and the complexity of the network (number and duration of the filters, number of TCN blocks) (see Table 4).

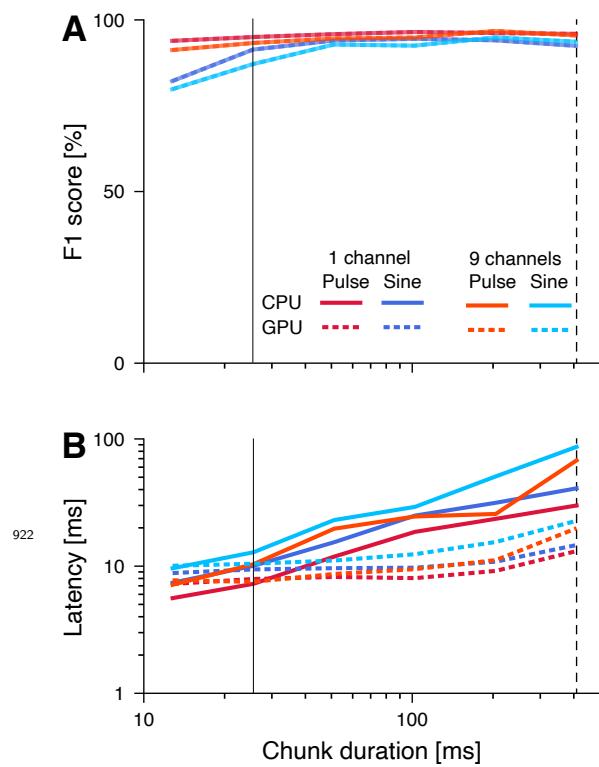


Figure 4-Figure supplement 2. Reducing the chunk duration reduces latency and comes with minimal performance penalties.

A F1 scores of networks with different chunk durations. Reduction of performance due to loss of context information with shorter chunks is minimal.

B Latency of networks with different chunk durations.

Dashed vertical lines indicate the chunk durations of the networks in **Figure 1**, **Figure 4B** and **Figure 4-Figure Supplement 1**. Solid vertical lines indicated the “short” chunk durations used in **Figure 4B** and **Figure 4-Figure Supplement 1**.

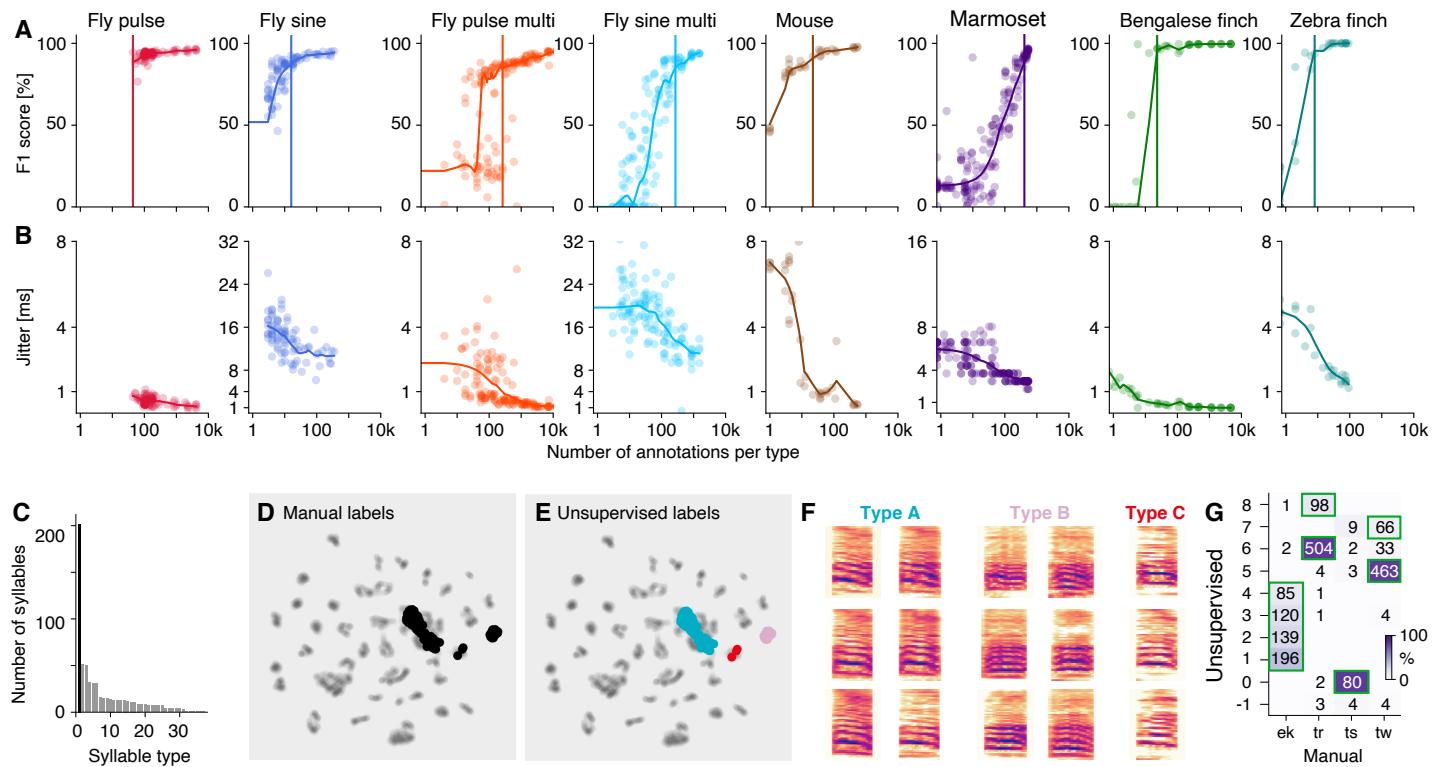


Figure 4-Figure supplement 3. DAS requires small to moderate amounts of data for training.

A Performance as a function of the number of manual annotations in the training set. Individual dots correspond to individual fits using different subsets of the data. For flies and mice, dots indicate the number of annotations for each type and performance is given by the F1 score – the geometric mean of precision and recall. For the marmosets and birds (rightmost panels), dots correspond to the median number of syllables per type in the training set and performance is given by the median accuracy over syllables per fit (see C for per-type statistics for Bengalese finches). Colored curves in A and B were obtained by locally weighted scatter plot smoothing (lowess) of the individual data points. Vertical lines correspond to the number of syllables required to surpass 90 % of the maximal performance.

B Temporal error given as the median temporal distance to syllable on- and offsets or to pulses.

C Number of annotations required per syllable type for Bengalese finches (range 2-64, median 8, mean 17). One outlier type requires 200 annotations and consists of a mixture of different syllable types (black bar, see D-F).

D, E UMAP embedding of the bird syllables with the outlier type from C labelled according to the manual annotations (D, black) and the unsupervised clustering (F). The unsupervised clustering reveals that the outlier type splits into two distinct clusters of syllables (cyan and pink) and three mislabelled syllables (red).

F Spectrograms of different types of syllable exemplars from the outlier type in C grouped based on the unsupervised clustering in E.

G Confusion matrix mapping manual to unsupervised cluster labels (see Figure 5F, G) for marmosets. Green boxes mark the most likely call type for each unsupervised cluster. While there is a good correspondence between manual and unsupervised call types, most call types are split into multiple clusters.

923

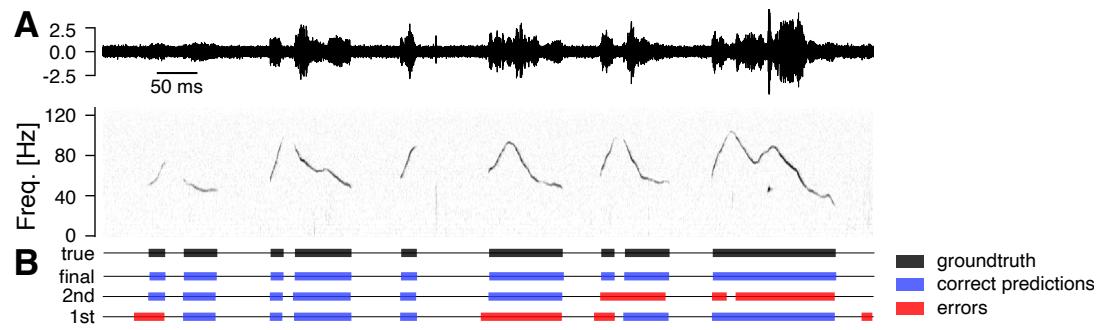


Figure 4-Figure supplement 4. Example of fast training mouse USVs.

924 DAS can be trained using an iterative procedure, in which the network is trained after annotating few syllables. This initial network is then used to create a larger dataset of annotations, by predicting and manually correcting annotation labels for a longer stretch of audio. This procedure is repeated to create ever larger training datasets until the performance is satisfactory. Shown is an example from mouse song.

A Example of the test recording for mouse song (top - waveform, bottom - spectrogram)

B Manual labels (true, black) and correct (blue) and erroneous (red) annotations from DAS for three different stages of training. Even for the first round of fast training, the majority of onsets is detected with low temporal error, requiring only few corrections. Number of syllables, seconds of song and F1-score for the different iterations (1st/2nd/final): 72/248/2706 syllables, 6/26/433 seconds of song, F1 score 96.6/97.9/98.9.

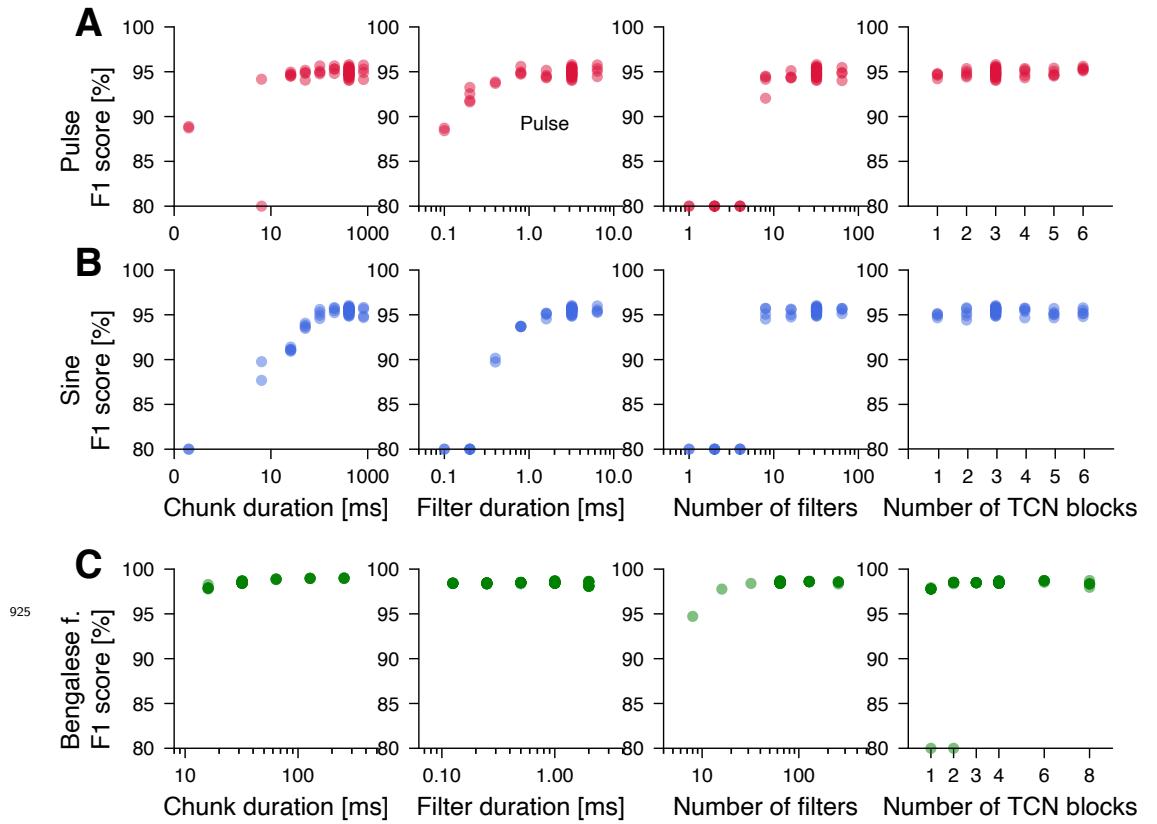


Figure 4-Figure supplement 5. DAS performance is robust to changes in the structural parameters of the network.

A, B Performance (F1 score) for networks with different structural parameters for fly pulse (A, red) and sine (B, blue). The starting point for each modified network was the network given in **Table 4**. We then trained network with individual modified parameters from scratch. As long as the network has chunks longer than 100 ms, filters longer than 2 ms, more than 8 filters, and at least one TCN block, the resulting model has an F1 score of 95 % for pulse and sine.

C Same in B but for Bengalese finches. Network performance is robust to changes in individual parameters. Convergence is inconsistent for shallow networks with 1-2 TCN blocks (rightmost panel).

F1 scores smaller than 80 % in A-C were set to 80 % to highlight small changes in performance.