General framework for signal processing in nonlinear mass-spring networks with application to keyword spotting

Finn Bohte, Alice Louvet, Vincent Maillou, and Marc Serra-Garcia*
AMOLF, Science Park 104, 1098 XG Amsterdam, Netherlands

(Received 9 April 2025; revised 26 July 2025; accepted 2 September 2025; published 14 October 2025)

Mechanical systems played a foundational role in computing history, and have regained interest due to their unique properties, such as low damping and the ability to process mechanical signals without transduction. However, recent efforts have focused primarily on elementary computations, implemented in systems based on predefined reservoirs, or in periodic systems such as arrays of buckling beams. Here we numerically demonstrate a passive mechanical system—in the form of a nonlinear mass-spring model—that tackles a real-world benchmark for keyword spotting in speech signals. The model is organized in a hierarchical architecture combining feature extraction and continuous-time convolution, with each individual stage tailored to the physics of the mass-spring systems considered. For each step in the computation, a subsystem is designed by combining a small set of low-order polynomial potentials. These potentials act as fundamental components that interconnect a network of masses. In analogy to electronic circuit design, where complex functional circuits are constructed by combining basic components into hierarchical designs, we refer to this framework as "springtronics." We introduce springtronic systems with hundreds of degrees of freedom, achieving speech classification accuracy comparable to that of existing submilliwatt electronic systems.

DOI: 10.1103/5hhw-sk9c

I. INTRODUCTION

Computing is not limited to conventional electronic systems [1], but has been demonstrated in a variety of unconventional platforms, such as spins [2], light [3], and DNA [4]. Among these, mechanical systems stand out for three key advantages. First, they allow direct processing of mechanical signals without transduction, as exemplified by passive sensors for spoken words [5] or steps [6]. Second, they can embody intelligent behavior within their structural dynamics, as demonstrated by agile soft robotic structures [7]. Third, they stand out for their low power dissipation, enabling experiments on the fundamental energetics of information processing [8] and promising efficient nanomechanical computation [9]—crucial for Internet of things devices. These advantages have sparked a renewed interest in mechanical computing, one of the earliest information processing platforms [10,11]. Recent results emanating from this interest include mechanical logic gates [12], an 8-bit processor [13], finite-state machines [14], and reservoir computers [15]. However, prior work focused primarily on elementary tasks such as counting [16], parity computation [17], and simple classification problems [5], where systems were predominantly designed with relatively simple architectures.

In this work, we numerically demonstrate a mechanical system—in the form of a nonlinear mass-spring model—that passively implements spoken keyword spotting (KWS), a real-world signal processing task. Owing to its hierarchical, multistage architecture, the model rivals the accuracy of low-power electronic systems [18] in a real-world, 12-class speech classification benchmark—representing a significant departure from the state of the art in mechanical computing, which has been focused on single-stage systems and toy problems.

The platform of mass-spring models consists of networks of discrete masses connected through (nonlinear) springs. These discrete models are established tools for studying complex mechanical phenomena, such as nonlinear effects in vibrations [19,20], allosteric responses in proteins [21], and locomotion in soft robots [22]. Importantly, discrete mass-spring models capture physical behavior independently of any specific realization. For example, the same mass-spring model captures frequency conversion in both magnetic systems [23] and geometrically nonlinear structures [24]. This abstraction is a fundamental principle in the design of electronic circuits, where idealized components—resistors, inductors, capacitors, etc.—are combined in discrete circuit models before consideration is given

*Contact author: m.serragarcia@amolf.nl

Published by the American Physical Society under the terms of the Creative Commons Attribution 4.0 International license. Further distribution of this work must maintain attribution to the author(s) and the published article's title, journal citation, and DOI.

to their physical implementation. Beyond electronics, this discrete-model-first approach recently found success in condensed matter physics. For instance, higher-order topological insulators were first discovered through discrete tight-binding models [25] and were later experimentally observed in physical systems [26]. Here we introduce a similar, discrete-model-first, approach for mechanical computing, constructing models from a small set of idealized mass-spring components. Inspired by circuit design, we refer to this approach as "springtronics." We demonstrate sprintronics by designing a mass-spring model that embeds a speech classification architecture with analog feature extraction and a continuous-time convolutional neural network (CNN).

This paper is organized as follows. In the remainder of the introduction, we establish the elements of the mass-spring models considered in this work. In Sec. II, we describe the keyword spotting architecture, which is composed of signal processing operations that are compatible with these mass-spring models. Section III focuses on the implementation of this architecture as a springtronic model. The model is organized into subsystems, each mapping to one signal processing step from the architecture introduced in Sec. II. In Sec. IV we examine the speech classification accuracy and energetic efficiency of our mass-spring model. Finally, the paper concludes with a discussion and outlook in Sec. V.

Throughout this paper, we use the term "mass-spring model" to refer to a network of masses interacting through force laws (Fig. 1). The mass-spring models are dynamic—with the masses subject to inertia and damping. The interactions considered here will all be conservative and reciprocal, with force laws derived from energy potentials acting on one or more degrees of freedom. The degrees of freedom consist of the scalar-valued positions $x_i(t)$ and velocities $\dot{x}_i(t)$ of the masses. While we depict masses graphically on a two-dimensional plane, as in Fig. 1, they should be thought of as having only one, out-of-plane displacement, as in Ref. [17]. This contrasts with studies on mass-spring networks that consider in-plane motion, i.e., where masses have two-dimensional displacements [21]. Although we visualize the models as consisting of masses vibrating out of the plane, in physical realizations our masses need not directly correspond to specific physical objects. Instead, they can be mapped to abstract degrees of freedom, such as structural modes or localized vibrations. Consequently, the model defines a topology rather than a geometry; the network dynamics are governed by the connections between masses rather than their spatial locations. In turn, the spatial layout of a physical realization is not fixed. For example, a model with linear topology could be realized compactly with cantilevers arranged in a winding or zigzag configuration. Finally, the mass-spring models are captured by equations

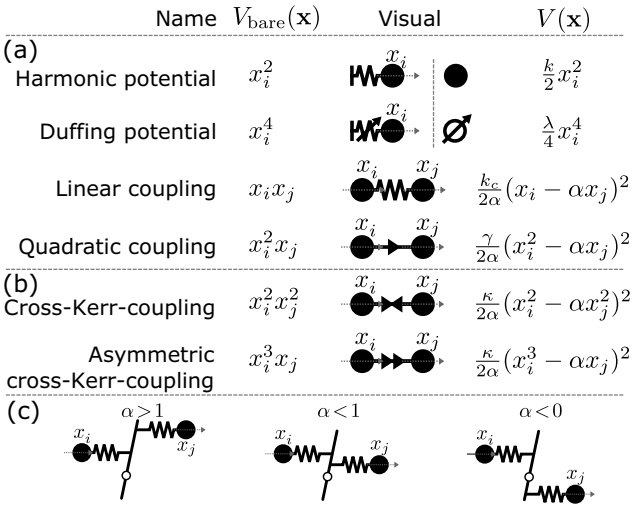


FIG. 1. Visual depiction of springtronics elements. Masses are represented by circles and are connected through potentials. Dampings are typically not drawn, but can be depicted as a dashpot. Potentials $V(\mathbf{x})$ are positive definite and act on one or multiple masses, and are illustrated in the following ways: (a) Local harmonic potentials are depicted as springs connected to a support. For Duffing potentials, the spring is crossed by an arrow. For visual clarity, springs for local potentials can be omitted, and the mass can be crossed by an arrow to indicate Duffing potentials. Linear coupling potentials are depicted as springs connecting two masses, and quadratic couplings are depicted as connecting lines overlaid with a triangle pointing toward x_j. Here "linearity" refers to the force-displacement relation derived from the potentials. (b) The cross-Kerr-coupling is depicted as a connection with two triangles pointing toward each other, and an asymmetric cross-Kerr-coupling is depited with two triangles pointing toward x_j. (c) Leverage parameter α. The coupling potentials are parametrized by a strength (k, γ, κ) and a leverage (α). The leverage can be understood as the arm of a lever inserted between two segments of a spring that connects two masses.

of motion

$$m_i\ddot{x}_i + b_i\dot{x}_i + \sum_j \frac{\partial}{\partial x_i} V_j = f_i(t), \tag{1}$$

where i indexes the masses, m_i is the mass, b_i is the local damping, $V_j(\mathbf{x})$ for $\mathbf{x} = (x_1, \ldots, x_n)$ are the potentials, and $f_i(t)$ is the external force.

Together with the masses and local dampings, interaction potentials form the elements of springtronics. We visually display them as shown in Fig. 1. The present work focuses on the four potentials listed in Fig. 1(a), consisting of the monomials $V_{\text{bare}}(\mathbf{x}) = x_i^2$, x_i^4, $x_i x_j$, and $x_i^2 x_j$. These monomials form an expressive basis; complex functions, from digital logic [13] to the speech recognition architecture implemented here, can be realized by combining these potentials. Moreover, they arise in a broad class of physical systems, because they constitute the low-order Taylor

approximation of a generic nonlinear potential [the Taylor approximation, when truncated to fourth order in the energy, may also contain additional terms, such as those in Fig. 1(b) which are not used in this work]. The first two potentials in Fig. 1(a) are local potentials, where x_i^2 yields a force linear to the displacement and x_i^4 yields a cubic force-displacement relation. We refer to the latter as a "Duffing term," but this same nonlinearity is known as the Kerr effect in optics. The last two are coupling potentials, where $x_i x_j$ yields a linear coupling and $x_i^2 x_j$ a nonlinear coupling. We refer to the linear coupling as a "linear spring" and the nonlinear coupling as the "quadratic coupling term." Physically, the latter interaction shows up in many systems in low-amplitude nonlinear regimes. In particular, it can be found in optomechanical cavities, for which rich intuition and experience has been accumulated [27]. In mechanics, the coupling appears in systems such as guitar strings, where pulling longitudinally on one end increases the vibrational frequency due to geometric nonlinearity [19,28]. In microelectromechanical system (MEMS) devices such as asymmetric nanobeam resonators, quadratic couplings can arise in the interactions between structural modes of vibration; in these systems, nonlinear parameters can be controlled through the geometric dimensions [29].

A note of caution is required: Not all systems composed of the above-mentioned elements have a lower bound on the potential energy. This can lead to divergent dynamics. For example, consider a system with two degrees of freedom and potential energy landscape $x_1^2 - x_1^2 x_2$. The system will experience divergence if the displacement x_2 exceeds 1. These divergences are unphysical—they do not occur in actual experimental realizations—and indicate that the bare term does not exist independently. To ensure that the springtronic models contain only physically meaningful interactions, we express our building blocks as combinations of terms $V_{\text{bare}}(\mathbf{x})$ that respect positive definiteness. We do so by constructing positive-definite polynomials $V(\mathbf{x})$, which can be factorized into the square of a difference, and thus enforce a lower bound on the potential energy. For the linear coupling $x_i x_j$, we take $x_i^2 - x_i x_j + x_j^2$ and write $V(x_i, x_j) = k_c/2\alpha(x_i - \alpha x_j)^2$, where k_c denotes the $x_i x_j$ coupling strength and α represents the leverage or arm of the coupling. α parametrizes the relative contribution of the local terms toward positive definiteness of V, and can be understood as the arm of a lever [Fig. 1(c)]. For typical springs, $\alpha = 1$, but in principle α can take any nonzero value. For the quadratic coupling $x_i^2 x_j$, we take $x_i^4 - x_i^2 x_j + x_j^2$ and write $V(x_i, x_j) = \gamma/2\alpha(x_i^2 - \alpha x_j)^2$, where γ denotes the $x_i^2 x_j$ coupling strength and α plays a similar role to that in the linear coupling. Remarkably, the requirement for positive-definite potentials has measurable real-world consequences. For example, in the case of the quadratic coupling nonlinearity in a guitar string, positive definiteness dictates that the boundary-induced frequency shift cannot exist independently of the longitudinal stiffness of the string, and of the Duffing nonlinearity of string vibrations.

In this section, we have introduced the components of the mass-spring models considered throughout the article [see Fig. 1(a)]. In the following sections, we explore how these components, i.e., the positive-definite potentials, can be combined to create an artificial neural network for speech classification. One challenge to overcome is that the building blocks of neural networks differ fundamentally from the potentials described above. Compared with neural networks, these mass-spring models exhibit reciprocal couplings, whereas neurons in a neural network break reciprocity. Additionally, the nonlinear interactions described above, and those typically arising in (micro)mechanical systems, are generally captured by lower-order polynomial terms, contrary to common activation functions in artificial neural networks such as sigmoidal functions and rectified linear units. We accommodate those differences by taking into account backaction when combining multiple stages, and by adapting the training process to work with the available nonlinearities.

II. KEYWORD SPOTTING ARCHITECTURE

In this work, we design a mass-spring model for KWS, a natural language processing problem where the task is to identify specific—sparsely occurring—keywords within speech signals. It forms the basis, for instance, of wake word detection to activate voice assistants [30]. Here we distinguish two stages in KWS systems: feature extraction and classification. Figure 2(a) illustrates these two stages in our system.

For feature extraction, we base our architecture on the log-mel spectrogram [31]. These features represent the time evolution of the signal energy at different frequency bands. In conventional digital implementations, they are obtained by applying a Fourier transform to windowed segments of speech, squaring the magnitudes to compute the power spectrum, and filtering it with a mel-scale filter bank—triangular bandpass filters that are logarithmically spaced to mimic human auditory perception. Finally, logarithmic compression is applied to reduce the dynamic range, yielding the log-mel spectrogram. The feature extraction stage of our system, shown in Fig. 2(b), resembles this standard method. However, we tailor the order of the operations and the type of nonlinearity to adapt the feature extraction to the physics of mass-spring models. These adaptations are described later in this section.

For classification, we base the model on CNN classifiers, which detect temporal and spectral correlations in the features. Specifically, we consider a CNN architecture with temporal convolutions [32]. The convolution combines the features $\mathbf{x}[t] \in \mathbb{R}^n$ computed for the last m time windows with a set weights $W \in \mathbb{R}^{n \times m \times c}$ and applies a

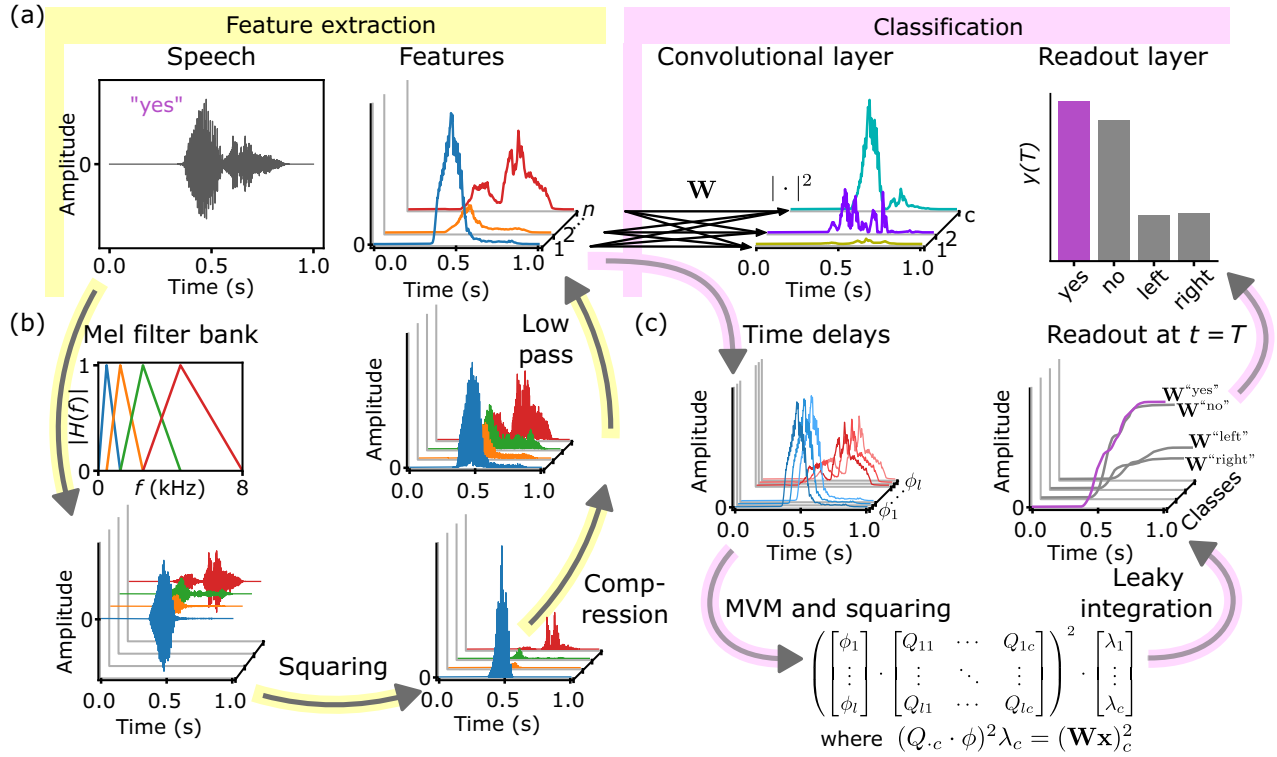


FIG. 2. Model architecture. (a) The model consists of a feature extraction stage and a classification stage. The n time-dependent features capture spectral and temporal patterns of the speech signal. The extraction method is based on log-mel spectrograms. Next, a convolutional classifier, composed of a CNN and a readout layer, predicts the likelihood of a keyword being present in the sound signal. (b) The features are extracted through signal filtering, signal squaring, and signal compression operations. First, a mel filter bank is applied, splitting the signal into multiple frequency bands. The resulting signals are squared, followed by a cubic root compression. Finally, a low-pass filter is applied to the signal. (c) The convolutional kernel weights $\mathbf{W}$ are realized via an instantaneous matrix-vector multiplication of time delayed copies of the features. We use a squaring activation function $x \mapsto x^2$ in the convolutional layer. The readout layer linearly combines the convolutional layer outputs, and performs a leaky integration over time. This integration yields the model readout. For each class, a different model is trained. The final prediction is given by the model producing the largest readout value.

nonlinear activation function σ, so that the output $\mathbf{h}[t] \in \mathbb{R}^c$ is given by

$$h_k[t] = \sigma\left(\sum_{i=1}^{n}\sum_{j=1}^{m} W_{ijk}x_i[t-(j-1)]\right). \qquad (2)$$

This operation can be understood as adding together several time-shifted copies of the signal $\mathbf{x}[t]$, and then applying a nonlinear transformation to the result. The weights W_{ijk} assigned to each time-shifted copy are the parameters of the CNN layer that are optimized during the training of the system—they determine what patterns the network will respond to. We structure the weights into a set of matrices $W_{ij}^{(k)}$ that we refer to as the convolutional kernels, each defining a channel of the CNN output. These weight matrices will be implemented as columns in a mechanical matrix-vector multiplication (MVM) [33] in the mass-spring model. The activation function in CNNs for speech classification typically is the rectified linear unit. However, this activation function does not have a springtronic equivalent. Thus, we will replace it by a squaring activation function.

Drawing from the established KWS architecture introduced above, we define our architecture in terms of analog, continuous-time signal processing operations, and discuss how the systems are trained, in this section. In Sec. III we discuss the mass-spring implementation.

A. Analog, continuous-time KWS architecture

This section introduces the analog, continuous-time KWS architecture that forms the basis of the mass-spring model. We preserve the operations from conventional digital KWS systems where feasible, replacing unsuitable blocks with alternatives that are more naturally adapted for mass-spring models.

The feature extraction stage, depicted in Fig. 2(b), comprises three operations: signal filtering, signal squaring, and cubic root compression. First, a mel filter bank splits

the input into n frequency-bin channels. Each filter output is then squared, rectifying the signal. We replace the logarithm in the log-mel spectrogram with cubic root compression, which is more amenable to mass-spring implementation, while maintaining similar noise-reduction benefits [34]. After compression, we apply a low-pass filter to remove high-frequency components introduced by the preceding squaring. Unlike digital KWS systems, which partition signals into time windows consisting of tens of milliseconds, and then compute features via fast Fourier transforms, analog systems process signals in continuous time. Still, we can achieve a similar temporal locality through the low-pass filter.

The classification stage of our model, illustrated in Fig. 2(c), is composed of a single temporal convolutional layer with a quadratic activation function, succeeded by a linear readout layer. This convolutional layer differs from Eq. (2), as conventional CNNs operate on discrete samples, while the systems considered here are continuous in time. Hence, the conventional convolutional kernels are defined as a discrete set of weights, while continuous-time convolutions would naturally imply the kernels are defined as continuous densities. However, we implement a discrete parametrization for the continuous kernels, allowing it to retain the structure of its digital counterpart. This is done by our defining the continuous kernel density as a weighted sum of Dirac δ functions $\delta_{t_j}(t)$ at uniformly spaced time points $t_j = (j-1)\Delta_t$. The coefficients multiplying the Dirac δ functions are the weights $W_{ij}^{(k)}$ from the discrete kernels. The convolution of this kernel and the analog features $\mathbf{x}(t)$ can now be expressed as

$$\int_0^t \left[\sum_{j=1}^{m} W_{ij}^{(k)} \delta_{t_j}(s)\right] x_i(t-s)ds.$$

Starting from this expression, we can now derive an expression similar to Eq. (2) for the output $\mathbf{h}(t) \in \mathbb{R}^c$ of the convolutional layer in our architecture. By interchanging integration and summation, making use of the shifting property $\int_0^t \delta_T(s) f(t-s)ds = f(t-T)$, and applying the squaring activation function $\sigma(x) = x^2$, we obtain

$$h_k(t) = \left(\sum_{i=1}^{n}\sum_{j=1}^{m} W_{ij}^{(k)} x_i(t-t_j)\right)^2. \tag{3}$$

Analogous to Eq. (2), the above operation can again be understood as combining time-shifted copies of the features $\mathbf{x}(t)$. Our systems implement this by introducing relative time delays of Δ_t increments between copies of the speech features, which are then combined through an instantenous matrix-vector multiplication that embeds the weights $W_{ij}^{(k)}$. Finally, the output channels $h_k(t)$ are combined in the readout layer, which also integrates the signal over time with exponential decay—an operation known as leaky integration. This leaky integration can be expressed as convolution with a continuous kernel given by $e^{-t/\tau}$, where τ sets the timescale of the decay, so the readout of the classifier is given by

$$y(t) = \int_0^t e^{-(t-s)/\tau} \sum_{k=1}^{c} h_k(s)ds. \tag{4}$$

Finally, the predicted class is determined from the output of the classification stage [Eq. (4)]. For binary classification problems, the architecture predicts one word or the other depending on whether the readout $y(t)$ is above or below a threshold at the final time $t = T$. For multiclass classification, the architecture predicts each class using a one-versus-the-rest approach, training distinct models per class, and selecting the final prediction via an argmax (or softmax) over all readouts [17].

B. Training

In this section, we describe the method for training the system to classify spoken words. Here the KWS system is trained by optimizing the weights of the convolutional kernels while keeping the rest of the architecture fixed. CNN kernels are commonly optimized using stochastic gradient descent, but the model from Eq. (4) forms a special case that allows more efficient training. Namely, the model can be cast in the form of a linear classifier.

First we derive the linear classifier from the expression of the model in Eq. (4). We begin by stacking the m time-delayed copies of the n speech feature channels into one vector $\phi(t) \in \mathbb{R}^{nm}$, i.e.,

$$\phi_l(t) = x_i(t-t_j), \quad \text{with } l = j + (i-1)m.$$

Next we rearrange the kernel weights into a matrix that acts on ϕ by taking

$$Q_{lk} = \lambda_k^{-2} W_{ij}^{(k)}, \quad \text{where } \lambda_k = |\sum_{i,j} W_{ij}^{(k)}|^{1/2}.$$

Taking $\Lambda = \mathrm{diag}(\lambda)$, we can rewrite Eq. (3) as the matrix identity

$$\begin{aligned} h_k(t) &= \lambda_k \left(\sum_{l=1}^{nm} Q_{lk}\phi_l(t)\right)^2 \\ &= (Q^\top \phi(t))^\top \Lambda (Q^\top \phi(t)) = \phi^\top(t) C \phi(t), \end{aligned} \tag{5}$$

where $C = Q\Lambda Q^\top$. Because C defines a quadratic form in Eq. (5), we can ensure C is symmetric and interpret Q as its eigenvectors and λ_k as its eigenvalues. Substituting

Eq. (5) in Eq. (4) and interchanging the summation and integration, we obtain

$$\sum_{k=1}^{c}\sum_{l=1}^{nm} C_{kl}\int_0^T e^{-(T-s)/\tau}\phi_k(s)\phi_l(s)dt = c^\top \Phi,$$

where we denote coefficients of the linear classifier by c and the classifier features by Φ. These features are obtained by computing $\int_0^T e^{-(T-s)/\tau}\phi_k(s)\phi_l(s)dt$ for all possible pairwise combinations (k,l), that are not equivalent due to time-translation symmetry. This yields a total of $N = nm + \binom{n}{2}(2m-1)$ remaining (k,l) pairs, so the dimension of the linear classifier is N.

To train the system, we can efficiently optimize the coefficients c using linear support vector machine (SVM) training algorithms [35]. For our systems, the classifier is of relatively low dimension, and the optimization problem is solved in the primal formulation of the L2-regularized L2-loss SVM (see Appendix D in Ref. [36]). The dimension of the classifier, as detailed above, follows from the—fixed—hyperparameters of the system: the number of frequency bins in the filter bank n, and the number of delays m corresponding to the width of the convolutional kernels. In this work, we focus on small models, with few mel filter bands $n = 4, 6, 8, 12, 16$ and time delays $m = 2, 3, 4$, and we fix the number of convolutional kernels to $c = nm$. The remaining hyperparameters are the delay time $\Delta_t = 32\,\text{ms}$, the readout time constant $\tau = 80\,\text{s}$, and the regularization strength for the SVM optimization. A more extensive exploration of these hyperparameters may offer further performance improvements in future work.

Finally, the convolutional kernels are determined from the optimized vector c through reconstruction of the quadratic form C. In the mass-spring model, the matrix of eigenvectors Q is embedded in the matrix-vector multiplication and the eigenvalues λ_k are embedded in the strength of the couplings connecting the matrix-vector multiplication network to the readout element. In a physical system based on the mechanical matrix-vector multiplication of Ref. [33], the matrix elements are realized through angles of the structure geometry. Similarly, the coupling strengths embedding the λ_k's are also a function of geometric parameters [as given by Eq. (8) in Ref. [29]].

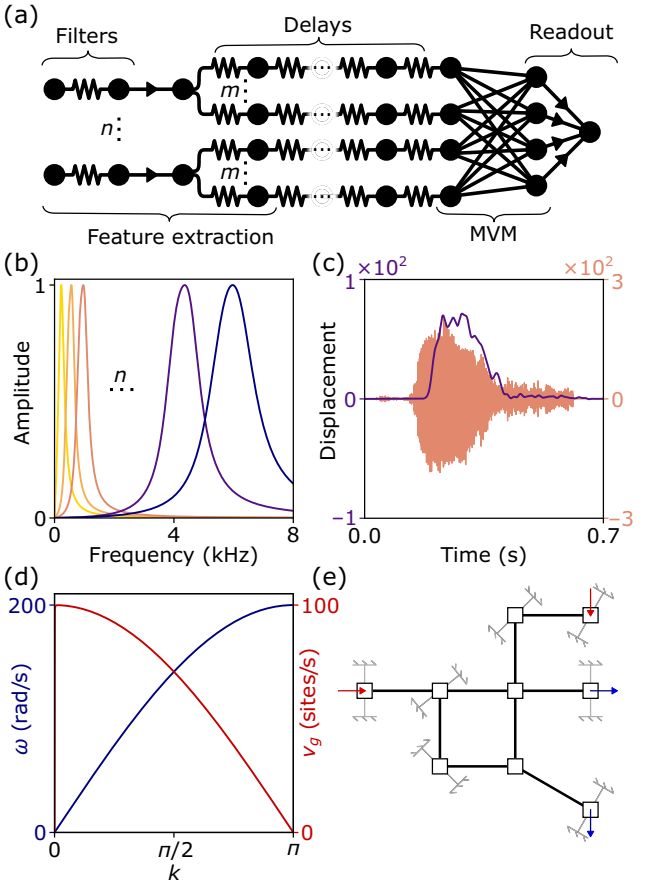


FIG. 3. Mass-spring model overview. (a) Speech classification mass-spring model. Subsystems are indicated with corresponding functionalities. (b) Amplitude response of the mass-spring approximation of the mel filter bank. The mechanical filter responses are formed by combining two Lorentzians [as shown in Fig. 4(b)]. (c) Example trajectory of a speech feature extracted by the mass-spring model (purple), with frequency band–filtered speech signals obtained through the approximate mel filter bank in (b) (orange). (d) Dispersion relation $\omega(k)$ for the delay line as per Eq. (7) (blue), with corresponding group velocity $v_g(k) = \partial\omega/\partial k(k)$ (red) for $m = 10^{-3}$, $k_c = 10.0$, and $k_l = 10^{-5}$. (e) Discretized representation of the geometry for the MVM unit cell from Ref. [33]. Note that the rectangles here represent masses that can move in the plane—corresponding to two springtronic degrees of freedom, and the straight lines represent geometric constraints. We construct an equivalent mass-spring model by assigning a large but finite stiffness to the geometric constraints, and then reducing the system to a set of input and output springtronic degrees of freedom.

III. MASS-SPRING MODEL IMPLEMENTATION

This section describes how we design the mass-spring model that realizes the KWS architecture from Sec. II. We implement the building blocks of the architecture as subsystems and interconnect them to construct the integrated system. An overview of the full mass-spring model is shown in Fig. 3. First, let us recall the building blocks of the speech classification architecture. The feature extraction stage consists of signal filtering, signal squaring, and signal compression. The convolutional layer of the classifier is composed of time delays, instantaneous matrix-vector multiplication, and squaring for the nonlinear activation function. Finally, the readout layer of the classifier combines the outputs of the convolutional layer and performs a leaky integration. We first discuss three building blocks that illustrate key design approaches for linear mass-spring models: signal filtering, time delays,

and matrix-vector multiplications. Each of these functionalities is achieved following distinct approaches: optimizing eigenmodes, tuning dispersion curves, and utilizing zero modes. We then discuss how the matrix-vector multiplication is integrated with the delay lines to complete the convolution. In the final subsection, we focus on the design of nonlinear building blocks by tuning the quadratic coupling. That subsection also covers the remaining steps required to complete and integrate the mass-spring speech classifier.

A. Design of linear mass-spring models

1. Mechanical mel filter bank: Designing frequency responses through eigenmode engineering

As discussed above, the first step in the feature extraction stage is signal filtering. Here we present the design of a mass-spring system that acts as a frequency band-pass filter. The amplitude response is tuned to match triangular mel filters by adjustment of the system's eigenmodes. The goal is to determine the parameters for a linear mass-spring model, with fixed topology, so that the system's frequency response $H(\omega)$ matches a predefined target response $T(\omega)$ in amplitude, while disregarding the phase of the response. The frequency response of the system is obtained through diagonalization, allowing the system's dynamics to be decomposed into a set of uncoupled eigenmodes. Combining the responses of the eigenmodes, the modal responses, we obtain the system response given by

$$H(\omega) = \sum_{k=0}^{n} \frac{F_k}{\omega_{0_k}^2 - \omega^2 + i\alpha\omega},$$

where n is the number of modes that compose the filter, ω_{0_k} denotes the modal frequency for eigenvalue $\lambda_k = \omega_{0_k}^2$, F_k represents the external force coupling to each mode, and α is the modal damping coefficient under the assumption of mass-proportional damping [i.e., $b_i = \alpha m_i$ for each mass in Eq. (1)]. The optimization problem is to minimize the misfit $\|T(\omega) - H(\omega)\|^2$ over ω_{0_k}, F_k, and α.

We construct a filter with two modes. While using more degrees of freedom will result in closer approximations, $n = 2$ achieves a good balance between accuracy and simplicity. Crucially, two modes can cancel out around $\omega = 0$ by coupling them with forces of opposing sign, as illustrated in Fig 4(b). This results in a response profile that matches the shape of the target filter much more closely than for a single mode, as shown in Fig 4(c), at the cost of a modest increase in complexity. The transfer function of this system is given by

$$H(\omega) = \frac{\alpha\omega_a^2}{\omega_a^2 - \omega^2 + i\alpha\omega} - \frac{\alpha\omega_b^2}{\omega_b^2 - \omega^2 + i\alpha\omega}, \qquad (6)$$

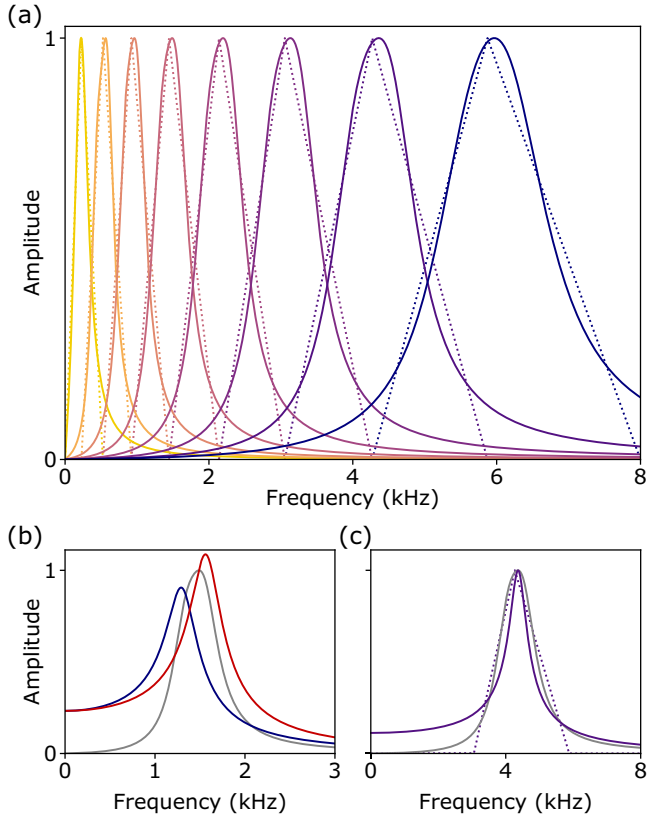


FIG. 4. Mechanical mel filters. (a) Amplitude responses of the optimized two-mode filter bank with eight filters (full line) and the target mel filter responses (dotted line). (b) Amplitude response of the fourth filter in (a) in gray, with the responses of the two individual modes it is composed of, $\alpha\omega_a^2/(\omega_a^2 - \omega^2 + i\alpha\omega)$ (blue) and $\alpha\omega_b^2/(\omega_b^2 - \omega^2 + i\alpha\omega)$ (red), the difference of which defines the filter as per Eq. (6). (c) Amplitude response of the seventh filter in (a) (gray) and an optimized single-mode—Lorentzian—filter (purple), with the corresponding mel filter (dotted line). The Lorentzian response is nonzero left of the mel filter band, and the width of the peak deviates more from the width of the triangular response compared with the two-mode filter case. These properties are related; the quality factor of peak-normalized Lorentizians determines both the response at $\omega = 0$ and the width of the peak. The response of the filter has an impact on the classification accuracy. For the classification problem described in Sec. IV, a two-mode filter achieves an accuracy of 81.4%, compared with 75.2% for a single-mode filter. As a reference, a digital mel filter bank achieves an accuracy of 82.0%.

and is parametrized by the frequency of the first mode ω_a, the frequency of the second mode $\omega_b = \Delta\omega_a$, defined via a relative difference Δ, and the damping parameter α. These parameters are optimized by means of gradient descent to minimize the misfit between Eq. (6) and the triangular mel filters, resulting in the filter responses depicted in Fig. 4(a). For the optimization, we initialize ω_a at the center frequency of the mel filter, set $\Delta = 1.1$, and choose $\alpha = 400$. We design a mass-spring system with this response using

two linearly coupled masses, x_a and x_b, each with the same local harmonic potential, as described by the following equations of motion:

$$\ddot{x}_a + \alpha\dot{x}_a + k_l x_a + k_c\,(x_a - x_b) = f_a(t),$$
$$\ddot{x}_b + \alpha\dot{x}_b + k_l x_b + k_c\,(x_b - x_a) = f_b(t).$$

By setting the masses to unity, we derive the mass-spring parameters from the modal parameters as follows: for the stiffnesses $k_l = \omega_a^2$, and $k_c = (\omega_b^2 - \omega_a^2)/2$, and for the force amplitudes $f_a \propto \alpha(\omega_a^2 + \omega_b^2)$, and $f_b \propto \alpha(\omega_a^2 - \omega_b^2)$ scaled to have peak response amplitude of 1. It is worth noting that, also in practice, mechanical bandpass filters have been experimentally realized with the use of two coupled resonators [37].

2. Mechanical delay lines: Designing time-dependent responses through dispersion engineering

As discussed in Sec. II, the convolutional layer is designed via a matrix-vector multiplication on time-delayed copies of the extracted speech features. Here we design the mass-spring model that induces these time delays. The model first splits each feature signal equally over m chains of linearly coupled oscillators, which function as waveguides. Specifically, these chains will be referred to as "delay lines," and the corresponding mass-spring model is described by

$$m\ddot{x}_i + k_c\,(2x_i - x_{i-1} - x_{i+1}) + k_l x_i = f_i(t),$$

where m is the mass, k_c is the coupling stiffness, k_l is the local stiffness, and i indexes the position along the chain. We tune the propagation speed of the delay lines to induce the relative time delays. The wave propagation speed is determined by the group velocity, given by $\partial\omega/\partial k$, where $\omega(k)$ is the dispersion relation that expresses the angular frequency ω as a function of the wave vector k. For the delay line, the dispersion relation and the group velocity (in sites per second) are respectively given by [38]

$$\omega(k) = \sqrt{\frac{k_l}{m} + \frac{4k_c}{m}\sin^2\left(\frac{k}{2}\right)}, \quad \frac{\partial\omega}{\partial k} = \frac{k_c}{m}\frac{\sin(k)}{\omega(k)}. \tag{7}$$

Notably, the group velocity is frequency dependent, so the delay line introduces signal distortion—different frequency components of the speech features propagate at different speeds. We mitigate this distortion by minimizing the variation in $\partial\omega/\partial k$ over the relevant frequency range. For a narrow frequency range, minimal variation is attained around the maximum of the group velocity, and this maximum can then be aligned with the frequency range of the signal. However, for our speech feature signals, we found the frequency range was too broad, and resorted to an alternative approach. Instead, we simply operate the delay line close to the limit of $k_l \to 0$, by taking $k_l = 10^{-6}k_c$. The result is a constant group velocity up to first order around zero frequency, with $\partial\omega/\partial k \approx \sqrt{m/k_c}$, as shown in Fig. 3(d). Subsequently, we obtain the desired time delay per site by tuning the mass m and coupling stiffness k_c. In the model, we first fix k_c and then set the mass according to $m = t_d^2 k_c$, for time delay t_d. For example, a 10-ms delay per site can be achieved by taking $k_c = 10$ and $m = 10^{-3}$. Recall that the architecture described in Sec. II requires m time-delayed copies per feature. To achieve this, we simply connect m delay lines to the features output with the same propagation speed to preserve the mechanical impedance, and multiply the lengths. In total, this leads to nm delay lines, as shown in Fig. 3(a), the outputs of which yield $\phi_l(t)$ from Eq. (5). We conclude this section by noting that the k_l/k_c ratio of 10^{-6} is not a strict requirement for practical implementation. For example, the system is still operable with an experimentally realizable stiffness ratio of 10^{-3} [39], at the cost of minimal performance hindrance [i.e., the system later considered in Fig. 8(e) experiences a 0.3 percentage point drop in accuracy]. Moreover, dispersion-free delay lines have been achieved with the use of acoustic metamaterials [40].

3. Matrix-vector multiplication: Designing instantaneous linear transformations through zero-mode engineering

Recall that we construct the convolutional layer with the time delays and an instantaneous matrix-vector multiplication. Here we discuss the mass-spring model for the matrix-vector multiplication that encodes the kernel weights of the convolutions. We adopt the mass-spring model from Ref. [33], which also includes an experimental demonstration. In our implementation, the mass-spring model uses idealized massless springs, eliminating dynamic effects. For physical realization, the masses need to be small enough to avoid any internal resonances when the structure is operated under dynamic conditions. Our MVM mass-spring model consists of a set of nm input masses and nm output masses, each of which is linearly coupled to all others, yielding a system with $2nm$ modes. Among these, nm are designed as zero modes—deformation patterns along which the internal forces in the system remain balanced—so that they encode the matrix elements, or the convolutional kernel weights in our case. Specifically, we encode the $nm \times nm$ unitary matrix Q from Eq. (5), which operates on the delay line outputs $\phi_l(t)$. The zero modes serve as the system's effective degrees of freedom and have a theoretical modal stiffness of zero, while the remaining modes ideally exhibit infinite stiffness. However, the stiffnesses of the nonzero modes in the mass-spring model inevitably remain finite. By tuning the lowest nonzero mode frequency sufficiently above the delay line's operating band, we ensure that

deformations along these modes remain negligible under the given input forces.

Finally, we complete the convolution by coupling the delay lines to the MVM to transmit the signals. Efficient signal transmission between linear subsystems relies on impedance matching, which minimizes reflections at the coupling interface. We achieve impedance matching by treating the MVM zero modes as an extension of the delay line and coupling them with the same stiffness as between the delay line masses. The zero modes, in principle, have zero stiffness, so the tuning focuses on modal mass and damping. The modal mass and damping are derived in the same way: Let us denote the *nm* zero modes of the system by $\mathbf{q}_z$. Then

$$\begin{bmatrix} \mathbf{x}_{\text{in}} \\ \mathbf{x}_{\text{out}} \end{bmatrix} = \begin{bmatrix} I \\ Q \end{bmatrix} \mathbf{q}_z,$$

since the zero modes encode Q. In our model, the degrees of freedom of the MVM all have the same mass m and damping b, leading to modal masses of $2m$ (and dampings of $2b$) because Q is unitary:

$$\begin{bmatrix} I \\ Q \end{bmatrix}^{\top} M \begin{bmatrix} I \\ Q \end{bmatrix} = I^{\top}(mI)I + Q^{\top}(mI)Q = 2mI.$$

Hence, we set the mass of the MVM input and output masses to half those in the delay line, and the damping to half of the impedance-matched damping of the delay line. The result is a mass-spring model that implements the sum in Eq. (5), corresponding to the convolution of the CNN layer.

B. Design of nonlinear mass-spring models

1. Nonlinear operations with quadratic coupling: The paradox of passive squaring

In the remainder of this section, we design the mass-spring models that realize the nonlinear operations in our KWS architecture. Recall that the nonlinear operations required for both the feature extraction and the activation function of the CNN rely on a signal squaring. Before discussing the nonlinear mass-spring models for speech classification, we first note that the squaring operation presents a paradox for passive systems: the operation increases signal energy for large-amplitude signals. Here we study the high-energy behavior of the quadratic coupling in a simple system to resolve this paradox. We embed the quadratic coupling in an infinite delay line and excite it with Gaussian pulse signals of increasing amplitude, as shown in Fig. 5(a). The infinite delay line is approximated by our truncating it and adding impedance-matched damping at the terminal sites. We define the input energy as the work done by the input force $F(t)$ on the input site x_1, $E_{\text{in}} = \int F(t)\dot{x}_1(t)dt$. The output energy is defined as the energy dissipated at the final site x_n,

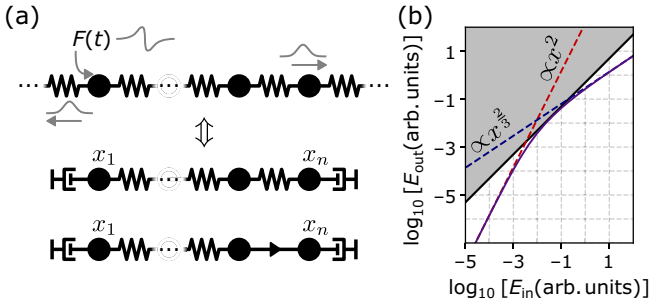


FIG. 5. Energetics of the quadratic coupling. (a) The system consists of a truncated chain of linearly coupled masses, modeling an infinite delay line, with impedance-matched damping on both ends to simulate an infinite chain length. The system is excited from the left terminal site x_1. We study how the energy dissipated at the right terminal site x_n changes when the last linear coupling is replaced with a quadratic coupling. (b) Log-log plot of energy dissipated at x_n (E_{out}) against energy input (E_{in}) for various input pulse amplitudes, comparing the quadratic coupling (purple) with linear coupling (black), with reference lines for squaring (red) and power 2/3 exponentiation (blue). The gray area is bounded by $E_{\text{out}} = \frac{1}{2}E_{\text{in}}$, corresponding to complete energy transfer where half of the input energy is dissipated at each terminal site, and indicates the area inaccessible due to energy conservation.

$E_{\text{out}} = \int b_n \dot{x}_n^2(t)dt$, where b_n is the impedance-matched damping value. The results, shown in Fig. 5(b), confirm that the quadratic coupling approximates squaring for low-energy input signals. For high-energy input signals, the coupling approximates a power 3/2 exponentiation. This effectively achieves a cubic root compression combined with a squaring—revealing the motivation for the feature extraction method from Sec. II.

2. Convolutional activation function and leaky integrator

As discussed in Sec. II, the classification stage is composed of a convolutional layer and a readout layer. For the mass-spring model design of the convolutional layer, two steps remain: scaling the MVM output with λ_k from Eq. (5), and applying the squaring activation function. We realize the squaring using a quadratic coupling, and encode the scaling in the strength of the coupling. The quadratic coupling connects the convolutional layer to the readout layer in the mass-spring model. The readout layer is implemented with the use of a single mass that is coupled to all the MVM outputs. First we design the response of the readout mass for leaky integration. This response is linear and parametrized by the frequency ω_r and quality factor Q_r. The quality factor determines the decay of the response. In the overdamped regime, characterized by high damping relative to inertia (i.e., low quality factor $Q < 0.5$), oscillatory systems exhibit exponential decay. The rate of decay is given by the slow timescale of the

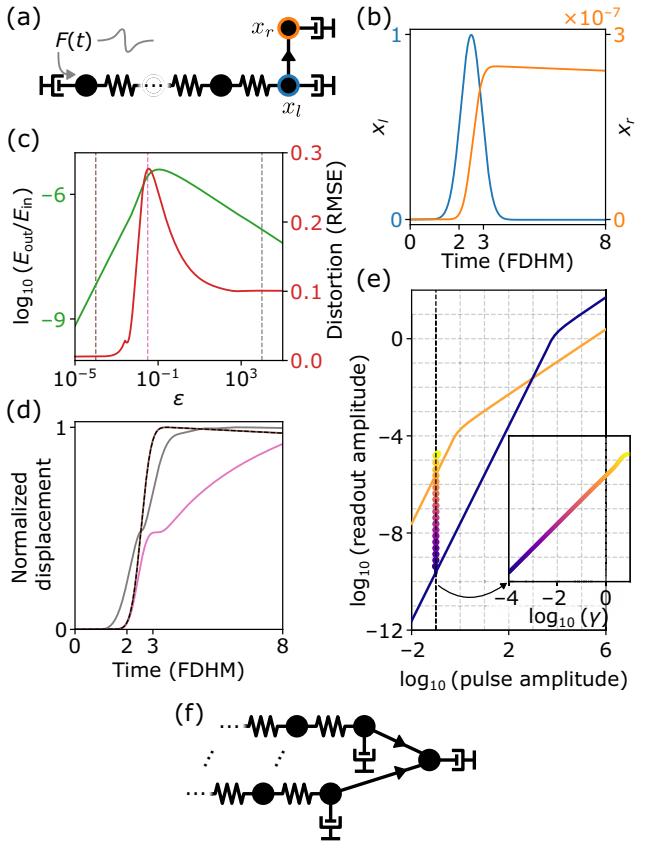


FIG. 6. Designing a readout layer based on a leaky integrator. (a) A single-input, single-output readout layer consists of a delay line excited by a pulse signal and terminated by impedance-matched damping, where the terminal site x_l (blue outline) is quadratically coupled to an overdamped mass x_r (orange outline) corresponding to the integrating readout. (b) Time-displacement curves of x_l (blue) and x_r (orange). The curve for x_l shows the pulse signal that propagates over the delay line, and the curve for x_r shows the leaky integrated pulse in the readout layer. (c) Energy transfer and distortion against the squaring error parameter ε. Energy transfer is computed as the energy dissipated by the readout divided by the work done by the input force, and is shown on a $\log_{10}$ scale. Distortion is computed as the root mean squared error (RMSE) between the readout trajectory and the leaky integrated square of the pulse signal [dashed black line in (d)] each normalized to their peak. (d) Peak-normalized readout trajectories for $\varepsilon = 10^{-4}$ (brown), $\varepsilon = 10^{4}$ (pink), and high distortion $\varepsilon = 3.16 \times 10^{-2}$ (gray). (e) Tuning γ to scale the readout. The readout displacement is plotted against input pulse amplitude for $\gamma = 1$ (orange) and $\gamma = 10^{-4}$ (blue). In the inset, the input pulse amplitude is fixed at 0.1 and γ is varied, and the readout displacement for varied γ is plotted (yellow to blue points) on a log-log scale. The results show a linear relationship for $\gamma \leq 1$, allowing scaling of the readout through γ. (f) Extension of the system in (a) to the multiple-input setting, which corresponds to the mass-spring model in Fig. 3(a). FDHM, full duration at half maximum.

system $\tau = 2Q_r/[\omega_r(1-\sqrt{1-4Q_r^2})]$. We fix $Q_r = 0.01$ and set $\omega_r \approx 1.25\,\text{rad/s}$ to match the readout timescale of $\tau = 80\,\text{s}$, as discussed in Sec. II.

Next we turn to the quadratic coupling for the squaring activation function of the convolutional layer. In the mass-spring model for speech classification, each of the MVM outputs is connected to a single readout. Here we consider one quadratic coupling that connects a terminated delay line to a leaky integrator, as shown in Fig. 6(a), and this setup straightforwardly extends to the multiple-input case illustrated in Fig. 6(f). This system is a variation of the previous setup shown in Fig. 5(a), but the quadratic coupling is now placed after the terminal site of the delay line that has impedance-matched damping. This placement reduces signal reflections from the quadratic coupling at the cost of a lower energy transfer—a trick we also use in the design of the classification mass-spring model. To design the activation function, we are interested in the parameter regime of the quadratic coupling, where the system performs leaky integration of the squared input signal. First we analyze the equations of motion to find a nondimensional parameter associated with the squaring error. Recall the quadratic coupling potential, and the resulting equations of motion as per Eq. (1):

$$m_l\ddot{x}_l + b_l\dot{x}_l + (k_l - 2\gamma x_r)x_l + 2\frac{\gamma}{\alpha}x_l^3 = F(t), \tag{8}$$

$$m_r\ddot{x}_r + b_r\dot{x}_r + (k_r + \gamma\alpha)x_r = \gamma x_l^2, \tag{9}$$

where $F(t)$ is the input signal and x_l corresponds to the delay line displacement, which is coupled to the readout mass with displacement x_r [see Fig. 6(a)]. Illustrative time trajectories of x_l and x_r are shown in Fig. 6(b). Notice that the stiffness of x_l has a cubic term and depends on the readout displacement. We refer to the effect of this dependency as "parametric backaction." Both the nonlinearity and the parametric backaction can cause errors in the squaring operation. Next we assume $k_r = 0$, so the stiffness of x_r stems only from the quadratic coupling via $\gamma\alpha$, which we tune through α. In turn, the resonance frequency of x_r becomes $\omega_r = \sqrt{\gamma\alpha/m_r}$, and the frequency response of x_r is given by

$$H_{x_r}(\omega) = \frac{\gamma F[x_l^2]}{\gamma\alpha - \omega^2 + i\omega b_r} = \frac{1}{\alpha}\frac{\omega_r F[x_l^2]}{\omega_r^2 - \omega^2 + i\omega(\omega_r/Q_r)},$$

which can be seen as α^{-1} times a linear operator acting on x_l^2. Reintroducing this into Eq. (8), we find that the parametric backaction becomes proportional to γ/α, just as the nonlinear stiffness, making it the only parameter associated with the squaring error at a given input pulse amplitude. The input pulse amplitude, which we denote by A, is determined by the amplitude of the force and the stiffness scale of the delay line k_l. Rescaling the displacement dimension for nondimensionalization of Eq. (8) reveals the error parameter depends on the square of the input pulse amplitude, yielding the squaring error parameter defined as $\varepsilon = A^2\gamma\alpha^{-1}$. Figure 6(c) shows the energy transfer and

the signal distortion against ε. Readout trajectories for different values of ε are shown in Fig. 6(d).

Now that we understand the parametric regime of the quadratic coupling for squaring, we move to the scaling of the output. Recall that this scaling implements multiplication of the λ_k's from the convolutional kernels. We continue with the assumption of $k_r = 0$, which leaves the coupling strength γ as the only remaining free parameter. To examine the effect of γ, we perform a sweep over the input pulse amplitudes for two values for γ, while keeping $\gamma\alpha$ fixed by adjusting α. The resulting readout amplitudes, shown in Fig. 6(e), indicate a shift of the squaring regime. We investigate how this shift scales with γ by fixing the pulse amplitude at 0.1 and varying γ, again for fixed $\gamma\alpha$. The inset in Fig. 6(e) shows a linear relationship between γ and the readout displacement. This relationship holds within the squaring regime, i.e., for small enough γ, and breaks down for large γ when the squaring regime shifts beyond the amplitude of the fixed input pulse. Bringing all of the above together, we tune the quadratic coupling by first fixing ε based on A, which is derived from the extracted features. Then γ is determined by λ_k of Eq. (5), and α is set by fixing $\gamma\alpha$ to tune the integration time of the readout. In practical nonlinear structures such as a MEMS, the quadratic coupling strength is a smooth function of geometric parameters that can be precisely controlled and calibrated [29].

3. Demodulation and compression for feature extraction

Recall from Sec. II that the feature extraction uses two nonlinear operations: signal squaring and cubic root compression. In the mass-spring model, both operations are achieved concurrently through a quadratic coupling operated in the high-energy regime. This coupling connects each filter bank output to a mass whose linear response implements a low-pass filter, completing our feature extraction stage. In total, we construct the feature extraction subsystem using our mass-spring mel filters, quadratically coupled to the low-pass filter, which then fans out to delay lines, as illustrated in Fig. 7(a). The features extracted with the use of this mass-spring model approximate the features from Sec. II well and qualitatively resemble the log-mel spectrogram, as shown in Figs. 7(b) and 7(c).

Finally, we design the remaining parameters for the feature extraction stage: the low-pass filter and the quadratic coupling. We set the parameters of the low-pass filter by first fixing the local stiffness k_{lp} and adjusting the mass m_{lp} to match the specified cutoff frequency $f_c = \sqrt{m_{lp}/k_{lp}}$. Recall that we fixed the low-pass frequency to 50 Hz in Sec. II. This value was motivated by visual inspection of the mass-spring features [see Fig. 7(d)] and assessment of the model on the validation set of the task discussed in Sec. IV. The last parameter, the damping, follows from the

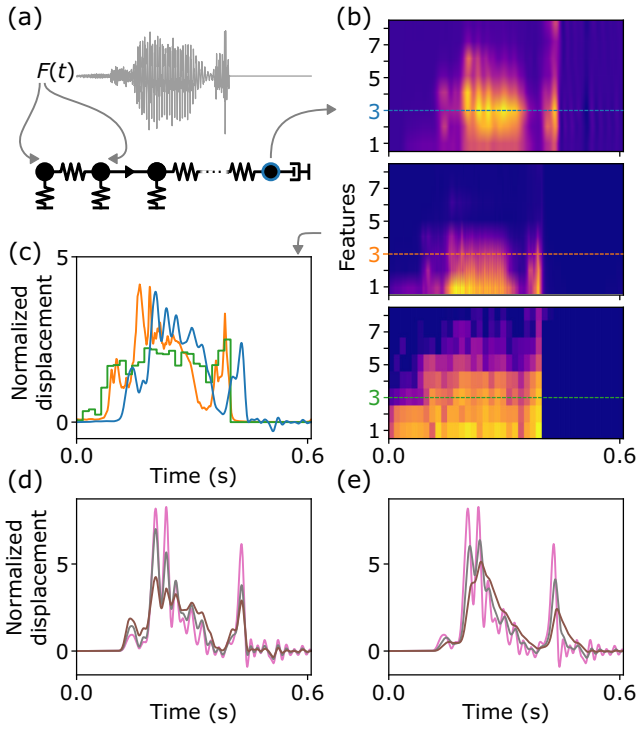


FIG. 7. Mass-spring feature extraction. (a) Feature extraction mass-spring model. The model consists of the two masses that act as the filter designed in Sec. III. These masses are quadratically coupled to the low-pass filter, which is connected to a delay line with terminal damping. Note that the low-pass mass has a local stiffness, in contrast to the system in Fig. 6(a), contradicting the assumption of $k_l = 0$ used in the analysis of the latter. The speech sample that we excite the system with is shown in gray. The displacements plotted in (b)–(e) correspond to the rightmost mass in the schematic (blue outline). (b),(c) Comparison of spectrograms [(b)] and their third channel [(c)]. Features are extracted via the mass-spring model (top, blue), direct evaluation of the architecture from Sec. II (middle, orange), and log-mel spectrograms (bottom, green). Amplitudes are normalized for the mean value. (d),(e) Time-displacement curve of the third mass-spring feature for different parameters: in (d) for different values of the compression parameter, $\alpha = 10^{-2}$ (pink), $\alpha = 10^{-4}$ (gray), and $\alpha = 10^{-6}$ (brown), and in (e) for different low-pass cutoff frequencies of 50 Hz (pink), 20 Hz (gray), and 10 Hz (brown). Displacement is normalized for the mean value.

quality factor of the filter via $b_{lp} = f_c/Q_{lp}$. In this work, we set $Q_{lp} = 0.5$ for critical damping, which prevents resonance with minimal energy loss. The damping of the mass has two origins: the local damping, and the impedance of the delay line to which it is coupled. We account for this by setting the local damping to b_{lp} minus the delay line impedance. For the quadratic coupling, we fix γ and tune α to control the compression. In this case, α determines in what regime of Fig. 5 we operate the coupling. In terms of the potential, recall that the positive-definite quadratic coupling introduces a nonlinear stiffness term $(\gamma/2\alpha)x_i^4$ on the filter output. Decreasing α lowers the compression

threshold, leading to a more compressed feature signal. The effects of compression strength, controlled through α, and the low-pass cutoff frequency f_c on the extracted speech features are displayed in Figs. 7(d) and 7(e), respectively. Analysis of the behavior of the quadratic coupling for compression is more complex than for the activation function discussed previously, because we cannot assume $k_r = 0$ for the system in Fig. 7(a), and we leave this for future work. At this point, we have completed the design of a mass-spring model that integrates a complete KWS system following the architecture from Sec. II.

IV. CLASSIFICATION PERFORMANCE AND EFFICIENCY

In this section, we test our model using the Google Speech Commands Dataset (GSCD), and explore how the performance relates to the energy transfer of the system. The GSCD provides a widely used benchmark for KWS systems, especially for low-power devices and small-footprint models. The dataset consists of 105 829 1-s recordings of 35 words by 2618 speakers. By default, 80% of the dataset is used for training, and the remaining is split evenly for validation and testing, both containing an equal number of samples for each class. Additionally, background noise with a random gain is added to the recordings in the training set with a probability of 0.8. Before exciting the mass-spring models with the speech signal, we perform additional preprocessing of the data that consists in centering the utterance in the time sequence, and subtracting signal components below the frequency range of speech. The latter is done by our identifying slowly varying signal background with a Savitzky-Golay filter ($f_c \approx 75$ Hz [41]) and subtracting it from the signal. The standard 12-class classification task is to distinguish a set of ten command words, "yes," "no," "up," "down," "left," "right," "on," "off," "stop," and "go," and two additional classes for "unknown word," comprising the remaining 25 words, and "silence" for recordings of background noise. By inclusion of a dedicated class for background noise, the task gauges the ability of the system to distinguish sparsely occurring speech events from background noise, which is crucial for KWS. In addition, the training data are overlaid with background noise to promote generalization and noise robustness. The performance metric—the accuracy—is simply the percentage of samples for which the top predicted class matches the ground truth label. In addition to this standard GSCD test, we also test the system on the four-digit binary classification task, also based on the GSCD, that was used in Ref. [5] on a mechanical system to compare it with prior work on mechanical speech recognition.

A remaining open problem is the detection of commands coming from multiple speakers. Mixed signals are a challenge for conventional KWS systems [42]. However, in mechanical speech recognition, an interesting alternative emerges by combining elastic event detection with acoustic speaker separation [43]. Assessing and improving the performance on a multispeaker setting is left for future work.

A. Mass-spring model classification performance

Here we determine the speech classification performance of the mass-spring model designed in Sec. III and discuss the results. We simulate the system using a fixed time step fourth-order Runge-Kutta algorithm [44] implemented on a graphics processing unit using CUDA [45]. First we excite the system with the speech files from the training set to compute the SVM features for training the model as discussed in Sec. II B. The training results in a model for each class, which we then excite with the files from the test set. The predicted class follows from the model with the highest readout value. The resulting test set accuracies are shown in Fig. 8(a) in parentheses for various model sizes. Our best-performing model achieved 80.7% accuracy, with the corresponding confusion matrix shown in Fig. 8(c) and the true positive rates for each class shown in Fig. 8(d).

The accuracy of our model depends both on the KWS architecture and the mass-spring implementation thereof. To understand whether the observed classification error originates in the mass-spring implementation, or if it is a limitation of the KWS architecture chosen, we compare the classification accuracy of the mass-spring models with a digital realization of the signal processing pipeline from Sec. II. The evaluation results for various model sizes are shown in Fig. 8(a). We found that the results are comparable to those for the mass-spring model, attaining an accuracy of 82.0%, compared with the 80.7% of the mass-spring model realization. This indicates the mass-spring model approximates the targeted signal processing operations to a significant degree, underlining the success of the springtronics approach. We suspect that a significant fraction of the mismatch can be attributed to the mass-spring realization of the mel filter bank, as the difference decreases for a higher number of filter bins. Moreover, the digital evaluation allowed us to explore the performance of the architecture for larger model sizes, up to 16 mel filter bins with four temporal delays per bin, which reached 88.1% accuracy. We hypothesize that significant room for improvement is left for the mass-spring model by both increasing the model size and tuning hyperparameters such as delay line time constants and filter parameters—potentially performing end-to-end backpropagation on the entire mass-spring model.

Next we compare our mass-spring model with electronic KWS systems. For classification, small and efficient models have been developed for deployment on edge devices, which reach accuracies of more than 95% (98.7% for

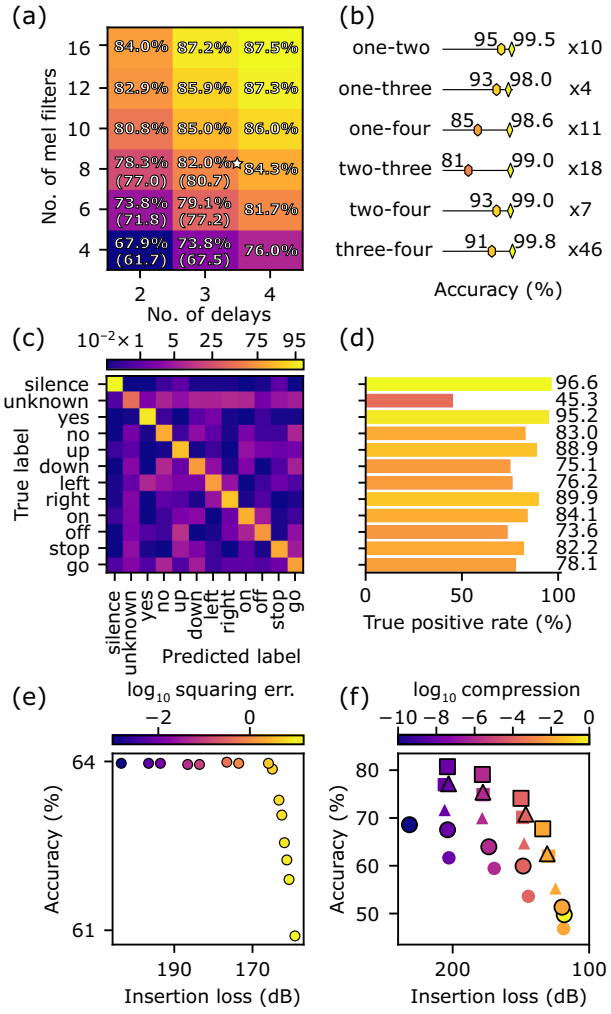


FIG. 8. Numerical characterization of the speech classification system. (a) Test set accuracy (in parentheses) of the KWS system for the GSCD 12-class task for various model sizes, based on digital implementation and simulation of mass-spring models. The best-performing mass-spring model is indicated with a star. (b) Binary classification accuracy for pairs of spoken digits "one," "two," "three," and "four" (diamonds) compared with that of the model from Ref. [5] (hexagons), and fold-error reduction (right). (c) Confusion matrix (true class normalized) for the best-performing mass-spring model. (d) True positive rate for each class, reflecting the confusion matrix diagonal. (e) Accuracy vs insertion loss for different values of the squaring error ε in the mass-spring CNN activation function. (f) Accuracy vs insertion loss for different model sizes and feature compression levels. The mass-spring filter bank size is indicated by a symbol (circle for four filters, triangle for six filters, and square for eight filters) and the number of delays is indicated by outline color (white for two delays and black for three delays). Different compression levels are shown in different fill colors. Squaring error is fixed at $\varepsilon = 1$ for all models.

BC-ResNet [46], 96.6% for TC-ResNet [32], and 98.8% for WaveFormer [47]). For these systems, the design commonly considers only the power consumption of the classification stage. However, this constitutes only part of the energetic cost of complete systems, with typical signal acquisition, transduction, and feature extraction taking 7.0 mW [18] (for microphone, analog-to-digital converter, and feature extraction). At a submilliwatt energy budget for the entire KWS system, Cerutti *et al.* [18] achieved 80% in experiment, compared with Hello Edge [18,48], which achieved 84.3% while consuming more than 10 mW. Since our mass-spring model integrates the entire KWS system and achieves 80% accuracy, we conclude that it performs competitively with low-power electronic systems.

We also directly compared our mass-spring model against previous mechanical systems for speech classification, and we found a substantial increase in performance. The passive mechanical system introduced in Ref. [5] was tested primarily on a binary classification task of pairs of spoken digits: "one," "two," "three," and "four." Specifically, the pair "two"-"three" posed a challenge for the linear model of Dubček *et al.* [5], plateauing at 59% accuracy. A simple nonlinear mass-spring model was able to reach 81% in this task. In contrast, the hierarchical nonlinear model proposed here achieves 99.0% accuracy for the pair "two"-"three," reducing the error by 2 orders of magnitude compared with prior linear work, and essentially saturating the benchmark. The classification accuracy for each pair compared with the model from Ref. [5] is shown in Fig. 8(b). Generally, we observe a severalfold reduction in classification error for all other word pairs—highlighting the information processing capabilities of hierarchical mass-spring models designed with the use of springtronics. In this section, we have demonstrated that mass-spring models can implement KWS architectures with much higher accuracies than prior studies—one of the main takeaway points of this article.

B. Energetic considerations

In electronic KWS systems there is a trade-off between power consumption and classification accuracy, as studied in depth in Ref. [18]. This subsection discusses the trade-off between the energy efficiency and accuracy in our model. The system is fully passive, starting from zero initial conditions, and relaxes back to the starting state through energy dissipation. The loss of signal power between input and output, quantified as insertion loss, is determined primarily by dissipative losses and signal reflections. These signal reflections can arise from impedance mismatches between linear systems or from nonlinearities such as the quadratic coupling. Three aspects of the model influence both accuracy and efficiency: the model size, the parameters of the squaring activation function, and the feature extraction parameters. Here we numerically investigate their effects by training the model and evaluating its classification accuracy and insertion loss on the test set.

First we fix the model size to a system with four frequency-bin filters and three time delays, and investigate the trade-off for the squaring and feature extraction. For the squaring activation function, our goal is to balance the energy transfer and squaring error. As shown in Fig. 6(c), reduction of the squaring error leads to a decrease in energy transfer. To extend this analysis to the classification model, we examine how the squaring error parameter affects both the test set accuracy and the insertion loss. The results are depicted in Fig. 8(e). We found the classification accuracy remains stable across a broad range for $\varepsilon < 1$, but starts to degrade beyond this point. This aligns with expectations from Fig. 6(c), where the accuracy of signal squaring undergoes a similar transition. To balance the trade-off for the squaring, we set $\varepsilon = 1$. The performance reduction at lower insertion losses is likely tied to our training approach, which assumes perfect squaring. A training strategy that accounts for the exact response of the quadratic coupling could extend its operational range, potentially reducing insertion loss further.

Next, for the feature extraction parameters, we explore the role of the design parameter α. Recall from Sec. III B that α controls the feature compression. We observe that greater compression increases accuracy, but at the cost of higher insertion loss, until a saturation is reached at around $\alpha < 10^{-10}$. Conversely, the reduction in insertion loss for lower compression saturated at around $\alpha > 1$. Regarding model size, Fig. 8(f) shows that increasing the model size increases accuracy with minimal impact on insertion loss, particularly at higher compression. Beyond the aspects of the model discussed here, we suspect that optimization of unexplored parameters—such as the integration timescale of the readout layer—could further reduce insertion loss. Another factor limiting efficiency is the reliance on the squaring activation function. Relaxing the assumption of perfect squaring and instead leveraging the exact dynamics of the quadratic coupling may lead to a more efficient convolutional layer. The key takeaway point from this study is that, within our current architecture and training protocol, there is a trade-off between the classification accuracy and the energy efficiency of the system.

V. CONCLUSION AND OUTLOOK

In this work, we have demonstrated a passive mass-spring model for speech classification, achieving accuracy comparable to that of low-power electronic systems, and is tolerant to manufacturing variations (see Appendix). While nonlinear mass-spring models have previously been explored for information processing, existing approaches have relied on reservoir computing [17]—where the system is parametrized randomly—or on repeating logic gates [13]. In contrast, we designed a modular, hierarchical mechanical system—consisting of filters, demodulators, delay lines, matrix-vector multiplications, nonlinear activation functions, and leaky integrators, each performing a relevant step of the computation. These subsystems are constructed from a small set of discrete, idealized components. Drawing an analogy to electronic circuit design, we refer to this design approach as "springtronics." By performing speech classification with competitive accuracy with low-power electronic solutions, our mass-spring models demonstrate that relatively simple systems—rooted in widely accessible classical mechanics—can perform complex information processing tasks. These systems are thus an elegant framework for theoretical studies on the physics of computation—where abstract information processing tasks must be embodied on a concrete physical realization to investigate their characteristics—and for potential applications on low-power MEMS devices, contributing to the realization of efficient, low-power mechanical computing across application domains.

This study also suggests several directions for future work. First, there is significant room for accuracy increases, for example, by constructing deep networks. This would require a different training approach, such as backpropagation with stochastic gradient descent. Designs can also be optimized to maximize the tolerance to fabrication imperfections and facilitate manufacturing. Second, a crucial next step involves realizing the device experimentally. This requires translating the mass-spring models into structural geometries, e.g., by leveraging methods such as those in Ref. [49,50]. Since springtronics is a generic information processing platform, future work should explore which tasks are best suited to such a mechanical computing approach. Springtronics is limited by the frequency and bandwidth of its constituent building blocks. Therefore, it is best suited to problems where signals are already mechanical, and where the frequency range and bandwidth match those of experimentally feasible mechanical resonators. Promising applications include structural health monitoring (detecting pipe leakages, crack formation, or harmful structural vibrations) and medical devices such as for step counting or heart rate monitoring. Within these applications, both digital and analog computing paradigms shall be explored.

ACKNOWLEDGMENTS

We thank Martin van Hecke for stimulating discussions. This work was funded by the European Union through ERC Grant No. 101040117 (INFOPASS).

The views and opinions expressed are those of the authors only and do not necessarily reflect those of the European Union or the European Research Council Executive Agency. Neither the European Union nor the granting authority can be held responsible for them.

DATA AVAILABILITY

The data that support the findings of this article are openly available [51], embargo periods may apply.

APPENDIX: ROBUSTNESS REGARDING IMPERFECTIONS

Physical implementations inevitably involve imperfections. Consequently, for realization of a device based on the mass-spring models designed in this work, it is crucial that the system still retains most of its performance under the finite precision of fabrication. Here we investigate how fabrication errors might affect system performance. In particular, we explore the effect of perturbations in parameters and removal of individual components.

First we consider random perturbation in the coefficients of the matrix-vector multiplication and the quadratic coupling strengths γ. The effect of small perturbations in these parameters is assumed to be interchangeable with the effect of small perturbations in the feature extraction stage, since both represent a mismatch between the system and the trained linear classifier. The results are shown in Figs. 9(a) and 9(b), respectively. We observe that for a 4% tolerance in γ, the accuracy remains above 80% in 52.1% of the systems—this means that one can retain an accuracy of more than 80% by rejecting half of the samples, simply doubling the fabrication cost. Similarly, for a 2% tolerance in the MVM coefficients, the accuracy remains above 80% in 45.0% of the systems. Such errors in MVM parameters correspond to imperfections in local and coupling stiffness of the physical realization. Stiffness parameters of a MEMS are well controlled and precisely tunable through postfabrication trimming using, for example, focused ion beams. For instance, an initial fabrication mismatch of 0.5% can be corrected up to less than 0.03% (in coarse, less than 50 Hz, and fine, more than 1 Hz, steps). [52] Errors in the nonlinear parameters, such as γ, are linear functions of geometric parameters and therefore are also tunable through postfabrication trimming [29].

Next we investigate the effect of the absence of components on the classification performance. Absent components break signal propagation, which can be represented by the removal of rows and columns of the MVM. Thus, we study the performance of the system when one to five randomly selected MVM rows or columns are removed. The results are shown in Fig. 9(c). In the worst case, the absence of one component can degrade the performance of the system close to the level of random guessing. However, 37.5% of the systems with three absent components still achieve more than 70% accuracy, and in the best case even five absent components do not cause a single percentage point drop in accuracy. These results suggest, on the one hand, there is some redundancy in the classifier and, on the other hand, that the classifier relies on a few critical components. To preserve performance, one can test fabricated devices and reject broken samples. However, it is also worth noting that micromechanical structures composed of on the order of hundreds to thousands of components have been successfully manufactured [53,54].

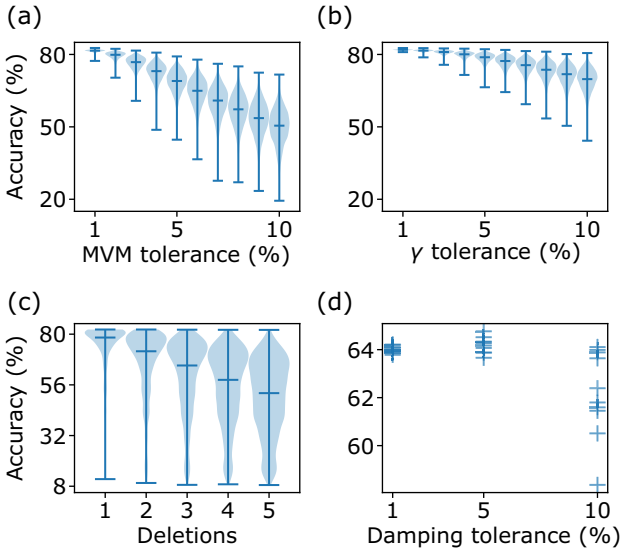


FIG. 9. Numerical investigation of the robustness of the speech classification system. Accuracies are shown for test set classification of the GSCD 12-class task: (a) Classification accuracy evaluated digitally for the system with eight mel filter bins and three delays, when noise is injected into the MVM coefficients. The coefficients are multiplied by a random variable uniformly distributed on $[1-\eta, 1+\eta]$, where $\eta \times 100\%$ is the tolerance. Distribution, extrema, and medians of 10^4 samples are shown for tolerances between 1% and 10%. (b) Classification accuracy for the same system as in (a), with noise applied to the coefficients γ of the nonlinear elements. (c) Classification accuracy for the same system as in (a), with up to five randomly selected rows or columns of the MVM deleted. (d) Classification accuracy for the same system as in Fig. 8(e), evaluated by simulation of the full mass-spring system, where noise of the same type as in (a) is injected into the damping coefficients [b_i in Eq. (1)] for ten samples each at 1%, 5%, and 10% tolerance.

Finally, we consider how accuracy is affected by perturbations in the damping parameters of the full mass-spring model, the b_i's in Eq. (1). The results, shown in Fig. 9(d), suggest accuracy can be robust regarding perturbations as large as 10%. The results reveal a 5% error in damping can increase the accuracy, indicating the current model parameters are not optimal.

Overall, the results suggest physical realization of our models does not require unreasonable precision, as the performance tolerates realistic imperfections. This robustness is somewhat expected given the system is designed to classify speech signals; utterances of the same word can differ greatly in pitch, tone, and speed between speakers. In the same vein, high sensitivity to parameter perturbations would be indicative of overfitting. However, we also found imperfections in some crucial parameters can

almost completely deteriorate the performance of the system. These parameters could be identified as part of future work, optimizing springtronic systems for robustness, similarly to considerations taken in designing digital computers [55]. We conclude that these results support practical feasibility of the theoretical mass-spring models proposed in the current work.

[1] H. Jaeger, B. Noheda, and W. G. V. D. Wiel, Toward a formal theory for computing machines made out of whatever physics offers, Nat. Commun. **14**, 4911 (2023).

[2] S. A. Wolf, A. Y. Chtchelkanova, and D. M. Treger, Spintronics—A retrospective and perspective, IBM J. Res. Dev. **50**, 101 (2006).

[3] P. L. McMahon, The physics of optical computing, Nat. Rev. Phys. **5**, 717 (2023).

[4] S. Yang, B. W. A. Bögels, F. Wang, C. Xu, H. Dou, S. Mann, C. Fan, and T. F. A. de Greef, DNA as a universal chemical substrate for computing and data storage, Nat. Rev. Chem. **8**, 179 (2024).

[5] T. Dubček, D. Moreno-Garcia, T. Haag, P. Omidvar, H. R. Thomsen, T. S. Becker, L. Gebraad, C. Bärlocher, F. Andersson, S. D. Huber, *et al.*, In-sensor passive speech classification with phononic metamaterials, Adv. Funct. Mater. **34**, 2311877 (2024).

[6] B. Barazani, G. Dion, J.-F. Morissette, L. Beaudoin, and J. Sylvestre, Microfabricated neuroaccelerometer: Integrating sensing and reservoir computing in MEMS, J. Microelectromech. Syst. **29**, 338 (2020).

[7] M. Sitti, Physical intelligence as a new paradigm, Extreme Mech. Lett. **46**, 101340 (2021).

[8] S. Dago, J. Pereda, N. Barros, S. Ciliberto, and L. Bellon, Information and thermodynamics: Fast and precise approach to Landauer's bound in an underdamped micromechanical oscillator, Phys. Rev. Lett. **126**, 170601 (2021).

[9] S. C. Masmanidis, R. B. Karabalin, I. D. Vlaminck, G. Borghs, M. R. Freeman, and M. L. Roukes, Multifunctional nanomechanical systems via tunably coupled piezoelectric actuation, Science **317**, 780 (2007).

[10] A. G. Bromley, Charles Babbage's analytical engine 1838, IEEE Ann. Hist. Comput. **4**, 196 (1982).

[11] T. Freeth, Y. Bitsakis, X. Moussas, J. H. Seiradakis, A. Tselikas, H. Mangou, M. Zafeiropoulou, R. Hadland, D. Bate, A. Ramsey, *et al.*, Decoding the ancient Greek astronomical calculator known as the Antikythera Mechanism, Nature **444**, 587 (2006).

[12] Z. Meng, W. Chen, T. Mei, Y. Lai, Y. Li, and C. Q. Chen, Bistability-based foldable origami mechanical logic gates, Extreme Mech. Lett. **43**, 101180 (2021).

[13] M. Serra-Garcia, Turing-complete mechanical processor via automated nonlinear system design, Phys. Rev. E **100**, 042202 (2019).

[14] J. Liu, M. Teunisse, G. Korovin, I. R. Vermaire, L. Jin, H. Bense, and M. van Hecke, Controlled pathways and sequential information processing in serially coupled mechanical hysterons, Proc. Natl. Acad. Sci. USA **121**, e2308414121 (2024).

[15] G. Dion, S. Mejaouri, and J. Sylvestre, Reservoir computing with a single delay-coupled non-linear mechanical oscillator, J. Appl. Phys. **124**, 154502 (2018).

[16] L. J. Kwakernaak and M. van Hecke, Counting and sequential information processing in mechanical metamaterials, Phys. Rev. Lett. **130**, 268204 (2023).

[17] J. C. Coulombe, M. C. A. York, and J. Sylvestre, Computing with networks of nonlinear mechanical oscillators, PLoS ONE **12**, e0178663 (2017).

[18] G. Cerutti, L. Cavigelli, R. Andri, M. Magno, E. Farella, and L. Benini, Sub-mw keyword spotting on an MCU: Analog binary feature extraction and binary neural networks, IEEE Trans. Circuits Syst. I: Regul. Pap. **69**, 2002 (2022).

[19] C. V. Raman, Experimental investigations on the maintenance of vibrations, Bull. Indian Assoc. Cult. Sci. **6**, 1 (1912).

[20] G. Duffing, *Erzwungene Schwingungen bei veränderlicher Eigenfrequenz und Ihre Technische Bedeutung* (Vieweg, Braunschweig, 1918).

[21] J. W. Rocks, N. Pashine, I. Bischofberger, C. P. Goodrich, A. J. Liu, and S. R. Nagel, Designing allostery-inspired response in mechanical networks, Proc. Natl. Acad. Sci. USA **114**, 2520 (2017).

[22] G. Urbain, J. Degrave, B. Carette, J. Dambre, and F. Wyffels, Morphological properties of mass–spring networks for optimal locomotion learning, Front. Neurorobot. **11**, 16 (2017).

[23] M. Serra-Garcia, M. Molerón, and C. Daraio, Tunable, synchronized frequency down-conversion in magnetic lattices with defects, Philos. Trans. R. Soc. A: Math. Phys. Eng. Sci. **376**, 20170137 (2018).

[24] M. Serra-Garcia, A. Foehr, M. Molerón, J. Lydon, C. Chong, and C. Daraio, Mechanical autonomous stochastic heat engine, Phys. Rev. Lett. **117**, 010602 (2016).

[25] W. A. Benalcazar, B. A. Bernevig, and T. L. Hughes, Quantized electric multipole insulators, Science **357**, 61 (2017).

[26] M. Serra-Garcia, V. Peri, R. Süsstrunk, O. R. Bilal, T. Larsen, L. G. Villanueva, and S. D. Huber, Observation of a phononic quadrupole topological insulator, Nature **555**, 342 (2018).

[27] M. Aspelmeyer, T. J. Kippenberg, and F. Marquardt, Cavity optomechanics, Rev. Mod. Phys. **86**, 1391 (2014).

[28] W. G. Conley, A. Raman, C. M. Krousgrill, and S. Mohammadi, Nonlinear and nonplanar dynamics of suspended nanotube and nanowire resonators, Nano Lett. **8**, 1590 (2008).

[29] K. Asadi, J. Yeom, and H. Cho, Strong internal resonance in a nonlinear, asymmetric microbeam resonator, Microsyst. Nanoeng. **7**, 9 (2021).

[30] I. López-Espejo, Z.-H. Tan, J. H. L. Hansen, and J. Jensen, Deep spoken keyword spotting: An overview, IEEE Access **10**, 4169 (2021).

[31] S. Davis and P. Mermelstein, Comparison of parametric representations for monosyllabic word recognition in continuously spoken sentences, IEEE Trans. Acoust. Speech Signal Process. **28**, 357 (1980).

[32] S. Choi, S. Seo, B. Shin, H. Byun, M. Kersner, B. Kim, D. Kim, and S. Ha, in *Interspeech 2019*

(International Speech Communication Association, Graz (Austria), 2019), p. 3372.
[33] T. Louvet, P. Omidvar, and M. Serra-Garcia, Reprogrammable, in-materia matrix-vector multiplication with floppy modes, Adv. Intell. Syst., 2500062 (2025).
[34] J. G. Lyons and K. K. Paliwal, in *Interspeech 2008* (International Speech Communication Association, Brisbane (Australia), 2008), p. 387.
[35] Y.-W. Chang, C.-J. Hsieh, K.-W. Chang, M. Ringgaard, and C.-J. Lin, Training and testing low-degree polynomial data mappings via linear SVM, J. Mach. Learn. Res. **11**, 1471 (2010).
[36] R.-E. Fan, K.-W. Chang, C.-J. Hsieh, X.-R. Wang, and C.-J. Lin, LIBLINEAR: A library for large linear classification, J. Mach. Learn. Res. **9**, 1871 (2008).
[37] M. S. Hajhashemi, A. Amini, and B. Bahreyni, A micromechanical bandpass filter with adjustable bandwidth and bidirectional control of centre frequency, Sens. Actuators A: Phys. **187**, 10 (2012).
[38] A. L. Fetter and J. D. Walecka, *Theoretical Mechanics of Particles and Continua* (Courier, Mineola, 2003).
[39] S. S. Verbridge, J. M. Parpia, R. B. Reichenbach, L. M. Bellan, and H. G. Craighead, High quality factor resonance at room temperature with nanostrings under high tensile stress, J. Appl. Phys. **99**, 124304 (2006).
[40] X. Zhu, K. Li, P. Zhang, J. Zhu, J. Zhang, C. Tian, and S. Liu, Implementation of dispersion-free slow acoustic wave propagation and phase engineering with helical-structured metamaterials, Nat. Commun. **7**, 11731 (2016).
[41] R. W. Schafer, What is a Savitzky–Golay filter? IEEE Signal Process. Mag. **28**, 111 (2011).
[42] Y. Shi, D. Wang, L. Li, J. Han, and S. Yin, in *Interspeech 2023* (International Speech Communication Association, Dublin (Ireland), 2023), p. 1488.
[43] K. Ma, H. Chen, Z. Wu, X. Hao, G. Yan, W. Li, L. Shao, G. Meng, and W. Zhang, A wave-confining metasphere beamforming acoustic sensor for superior human-machine voice interaction, Sci. Adv. **8**, eadc9230 (2022).
[44] W. H. Press, S. A. Teukolsky, W. T. Vetterling, and B. P. Flannery, *Numerical Recipes in C* (Cambridge Univ. Press, Cambridge, 1992).
[45] J. Sanders and E. Kandrot, *CUDA by Example: An Introduction to General-Purpose GPU Programming* (Addison-Wesley, Upper Saddle River, NJ, 2010).
[46] B. Kim, S. Chang, J. Lee, and D. Sung, in *Interspeech 2021* (International Speech Communication Association, Brno (Czech Republic), 2021), p. 4538.
[47] M. Scherer, C. Cioflan, M. Magno, and L. Benini, in *Design, Automation & Test in Europe Conference & Exhibition (DATE)* (IEEE, Valencia (Spain), 2024), p. 1.
[48] Y. Zhang, N. Suda, L. Lai, and V. Chandra, Hello edge: Keyword spotting on microcontrollers, ArXiv:1711.07128.
[49] A. L. Vanel, O. Schnitzer, and R. V. Craster, Asymptotic network models of subwavelength metamaterials formed by closely packed photonic and phononic crystals, Europhys. Lett. **119**, 64002 (2017).
[50] K. H. Matlack, M. Serra-Garcia, A. Palermo, S. D. Huber, and C. Daraio, Designing perturbative metamaterials from discrete models, Nat. Mater. **17**, 323 (2018).
[51] F. Bohte, T. Louvet, V. Maillou, and M. Serra-Garcia, Replication package for the article 'Mass-spring models for passive keyword spotting: A springtronics approach', https://doi.org/10.5281/zenodo.17056163 (2025).
[52] A. Efimovskaya, D. Wang, and A. M. Shkel, Mechanical trimming with focused ion beam for permanent tuning of MEMS dual-mass gyroscope, Sens. Actuators A: Phys. **313**, 112189 (2020).
[53] J. Doster, T. Shah, T. Fösel, P. Paulitschke, F. Marquardt, and E. M. Weig, Observing polarization patterns in the collective motion of nanomechanical arrays, Nat. Commun. **13**, 2478 (2022).
[54] C. Dorn, V. Kannan, U. Dreschler, and D. M. Kochmann, Graded phononic metamaterials: Scalable design meets scalable microfabrication, ArXiv:2507.01874.
[55] S. Borkar, Designing reliable systems from unreliable components: The challenges of transistor variability and degradation, IEEE Micro **25**, 10 (2006).