Predicting broadband resonator-waveguide coupling for microresonator frequency combs through fully connected and recurrent neural networks and attention mechanism

Masoud Soroush,[†] Ergun Simsek,[*,†] Gregory Moille,[‡,¶] Kartik Srinivasan,[‡,¶] and Curtis R. Menyuk[†]

[†]*Department of Computer Science and Electrical Engineering, University of Maryland Baltimore County, Baltimore, MD 21250, USA.*

[‡]*Microsystems and Nanotechnology Division, National Institute of Standards and Technology (NIST), Gaithersburg, MD 20899, USA.*

[¶]*Joint Quantum Institute, NIST/University of Maryland, College Park, MD 20742, USA.*

E-mail: simsek@umbc.edu

Abstract

Broadband microresonator frequency combs are being intensely pursued for deployable technologies like optical atomic clocks. Spectral features such as the dispersion in their coupling to an access waveguide are critical for engineering these devices for application, but optimization can be computationally intensive given the number of different parameters involved and the broad (octave-spanning) spectral bandwidths. Machine learning algorithms can help address this challenge by providing estimates for the coupling response at wavelengths that are not used in the training data. In this work, we examine the accuracy of three neural network architectures: fully-connected

neural networks, recurrent neural networks, and attention-based neural networks. Our results show that when trained with datasets that are prepared by including upper and lower limits of each design feature, attention mechanisms can predict the coupling rate with over 90 % accuracy for spectral ranges 6× wider than the spectral ranges used in training data. Consequently, numerical optimization for the design of ring resonators can be carried out with a significantly reduced computational burden, potentially resulting in a six-fold reduction in compute time. Furthermore, for devices with particularly strong correlations between design features and performance metrics, even greater acceleration may be achievable.

Introduction

With their ability to unravel complex relationships between the inputs and outputs of nonlinear systems, machine learning (ML) methods have become a focal point of research and development during the past decade in almost all areas of science and engineering. The field of photonics also has seen a surge of interest in the use of artificial intelligence, particularly neural networks, to enhance the understanding and application of light-matter interactions. In particular, neural networks have already shown promise in a variety of photonics applications such as inverse photonic design,[1–12] material and device characterization,[1,3,13–19] optical sensing,[20] image processing and classification,[21] and optical communication.[22] In inverse photonic design,[1–12] the goal is to design optical components or devices with specific desired properties such as the desired transmission spectrum, scattering properties, bandwidth, or quantum efficiency. This goal can be achieved using numerical optimization algorithms that iteratively adjust the design parameters until the desired performance is achieved. However, there are also some other techniques that follow completely different strategies to do photonic inverse design. For example, in the adjoint method,[4–10] the gradient of an objective function with respect to the design parameters of a device is calculated first. Then, this gradient is used to optimize the device design through a process of iterative refinement. The

adjoint method has successfully been used to design waveguides,[7] beam splitters,[4–6] couplers,[10] and metamaterials.[6,7] Another successful strategy is using two neural networks. In generative adversarial networks (GANs),[11,12] for example, two neural networks, the generator and the discriminator, are trained together in an adversarial way. The generator is trained to create photonic structures that perform a specific function. The discriminator is then trained to identify whether the generated photonic structures perform the desired function or not. Through this adversarial process, the generator becomes better at designing photonic structures that perform the desired function, while the discriminator becomes better at identifying whether a given structure performs the desired function. Back-to-back (BtB) neural networks also employ two neural networks.[23,24] The first network is trained using a dataset of desired device performance metrics (such as transmission, reflection, quantum efficiency, dispersion, etc.) and corresponding design parameters (such as the dimensions and materials of the device). The second network is trained using a dataset of design parameters and corresponding device structures, typically generated using a physics-based simulator. Once the two networks are trained, they can be used together to perform photonic inverse design by specifying the desired device performance as input to the first network, which produces the corresponding design parameters as output. These design parameters can then be input to the second network, which generates the corresponding device structure.

For the forward problems in photonics, such as predicting the reflectance spectrum of nanoparticles,[13] optical properties of photonic crystals[14] and photonic crystal fibers,[15] or device performance metrics from device parameters,[16–19] fully connected and recurrent neural networks (FCNNs and RNNs, respectively) are two of the most common deep learning architectures. Due to the relative simplicity of forward problems compared to the inverse problems, neural networks can make predictions with very high accuracy, typically 98 % or even higher, e.g. in[13] and.[19] This high accuracy is beneficial for several types of problems in photonic design. As a concrete example, we may consider a small-scale device design problem that is tackled using numerical simulations in which there are only four parameters

to be optimized. If we try to implement a simple grid-based approach and try 20 unique values for each parameter, then we need to run $20^4 = 160,000$ simulations. Even if each simulation takes one second, the entire study would take almost two days. Through a machine learning-based approach, we can pick fewer unique values, let's say eight for each parameter, carry out $8^4 = 4096$ simulations, then train a neural network and make predictions for those 160,000 unique designs in only two hours (approximately). This scenario, of course, assumes a weak nonlinearity, i.e., there is a moderate correlation between the inputs and outputs. However, in photonics, we do not always deal with these kinds of weakly non-linear problems. For example, the evanescent coupling of light that propagates in a waveguide into an adjacent microring resonator requires a more sophisticated approach, which is the subject of this work.

Ring resonators are widely used in optics and photonics as critical elements of optical modulators, filters, switches, sensors, and lasers.[26,27] Light is typically coupled in and out of ring resonators via an adjacent waveguide through evanescent coupling. Resonator-waveguide coupling is quantified by introducing a coupling quality factor (Q_c), which is inversely proportional to the rate of coupling between the two elements, and depends on spatial mode overlap, phase mismatch, and coupling length. The coupling quality factor Q_c is wavelength-dependent, as both the spatial mode overlap and the phase mismatch vary with wavelength. Point coupling and pulley coupling are two common coupling configurations in use. A benefit of the pulley-type coupling is that the waveguide is wrapped around the resonator in order to increase the coupling length, as opposed to the traditional approach wherein straight waveguides provide a single-point coupling.[28,29] It has recently been shown that pulley-type coupling is particularly important in the context of broadband optical frequency combs based on microring resonators,[30] where it can be engineered to realize targeted values in different frequency bands of interest across an octave of spectral bandwidth. Such octave-spanning frequency combs that are based on integrated microring resonators are being intensely pursued for many applications, including the development of new generations

of optical atomic clocks and frequency synthesizers.[31–33]

After determining the electric field profiles inside the waveguide and ring resonator for a chosen frequency, Q_c can be calculated using the coupled mode theory (CMT) formalism of Ref.[30] This procedure can then be followed for a wide range of frequencies, e.g., over an octave, and for a variety of different waveguide geometries (parameterized by the waveguide width, resonator-waveguide gap, and pulley coupling length), in order to build a library of results for Q_c for a given ring resonator. As it is previously mentioned, this approach is laborious and computationally intensive. Thus, it is natural to seek more efficient methods, such as neural networks, to ease these challenges and yield a significant improvement. However, when running the CMT calculations, one can see that Q_c may change by ten orders of magnitude or even more across the octave of spectral bandwidth. This radical behavior of Q_c poses serious challenges for the task of predicting the coupling quality factor solely from geometric parameters and the wavelength. This extremely nonlinear behavior of the coupling quality factor cannot entirely be captured by even today's most advanced machine learning algorithms.

A more tractable approach is to use a portion of the Q_c values, e.g., the ones calculated at low frequencies, as the input of the model and then aim to predict the Q_c values at other frequencies (e.g., at high frequencies). If high accuracy is obtained, then this procedure can be used to reduce the computation time typically spent during device design and optimization studies. Considering the proven success of deep learning in forward applications,[13–19] high accuracy can be obtained with different types of neural networks such as FCNNs and RNNs. Noting that FCNNs require a fixed input size by design, if one wants to work with varying size inputs such as Q_c values calculated at varying frequency values and ranges, then RNNs would be the more appropriate choice. However, RNNs also suffer from their own disadvantages and limitations,[34] e.g., they get slower with increasing input size and their accuracy may drop significantly with decreasing input size. With their ability to pay attention to certain parts of the input, an attention mechanism is proposed as an alternative solution method to

address and resolve the issues of RNNs.

In this work, we compare the performances of these three NN implementations (FC-NNs, RNNs, and neural networks with an attention mechanism) in a brief yet systematic study focusing on two different scenarios. In the first scenario, we split the original dataset randomly into two to be used as training and testing datasets. In the second scenario, we split the original dataset in such a way that two unique values of the design features (ring width and waveguide width) are never used in the training, and more importantly, these values are beyond the original training range, e.g. $RW_{\text{test}} \notin [RW^{\text{min}}_{\text{training}}, RW^{\text{max}}_{\text{training}}]$ and $W_{\text{test}} \notin [W^{\text{min}}_{\text{training}}, W^{\text{max}}_{\text{training}}]$. Hence, we call these two scenarios interpolation and extrapolation problems, respectively. Once the neural networks are trained and a sufficiently high accuracy is achieved, the interpolation can be used for reducing the computation time and resources used for the design of microresonators meeting the required specifications within the predefined physical parameter space, while the extrapolation can be used to achieve the same for microresonators with physical parameters that are new to the neural network.

The organization of this manuscript is as follows. First, we provide a brief theoretical background for the physics of the resonator-waveguide coupling. Next, we quickly go over the fundamental concepts related to neural networks and present the detailed mechanics of the attention mechanism used in the context of deep learning. We then illustrate how we employ the additive attention mechanism as a tool to predict the Q_c values at higher frequencies based on the device design parameters and the Q_c values at low frequencies. We investigate how the accuracies of the designed networks change with the input length and the choice of optimizer, first for the interpolation problem and later for the extrapolation problem. Our numerical results indicate for the interpolation problem that even with quarter-octave data (i.e., Q_c values given for the frequencies ranging from ν to 1.25ν, with ν being the starting frequency of the low frequency band), we can predict Q_c at higher frequencies (e.g., 1.25ν to 2.5ν) with accuracies above 90 % using the attention mechanism, which means an 83% reduction in computation time. For extrapolation problems, high accuracy requires

larger training data compared to interpolation problems but still, 90 % accuracy and a 33 % reduction in computation time are possible. This implies that once the designed neural networks are trained with a sufficiently wide frequency range of data for Q_c values, the machine learning methods turn out to be a valuable source in reducing the consumed time and memory used to evaluate photonic device performance. Our study suggests that the same idea can be applied to estimate the characteristics of moderately non-linear devices when a large multi-dimensional dataset over a sufficiently broad frequency range is used for training.

Resonator-Waveguide Coupling Calculations

Figure 1 (a) illustrates the geometry studied in this work. The Si_3N_4 microring with an outer radius R and ring width RW is separated by a gap G from the coupling waveguide of width W. To increase the interaction length between the waveguide and the ring, the waveguide is wrapped around a portion of the microring, resulting in a pulley coupling design, where L_c represents the length of the pulley. To achieve an octave-spanning frequency comb for optical clock applications, the height of the ring is fixed to 410 nm and the ring width is chosen to be around 855 nm.[33] The ring radius is fixed to 80 μm to target a 275 GHz repetition rate, which would potentially be directly detectable through recent advances in high-speed photodetection.[35] 5750 unique devices are studied by changing the dimensions in a systematic fashion. Table 1 lists the minimum, maximum, and step size values for RW, W, G, and L_c. The substrate (lower optical cladding) is SiO_2, while the top and side claddings are air.

The resonator-waveguide coupling can be calculated using the spatial coupled mode theory (CMT) formalism in an integrated planar geometry by considering only the region over which their fields interact.[30] This approach essentially calculates a coupling rate per each

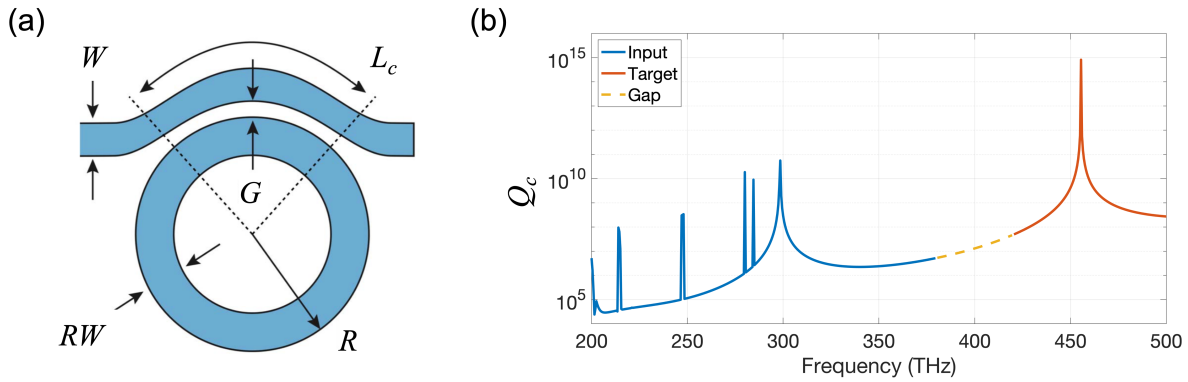

Figure 1: (a) Schematic illustration of the pulley-coupled microring resonator. (b) Q_c vs. frequency for the design where $RW = 865$ nm, $W = 650$ nm, $G = 300$ nm, and $L_c = 38$ μm. Note that along the region between the dashed black lines, the gap between the waveguide and ring is constant. Low frequency Q_c values (depicted with blue) are calculated via the coupled mode theory (CMT) and used as input along with design parameters. The objective of the neural network is to predict the high frequency Q_c values (depicted in red). The curve shown by the yellow dashed line represents the gap region that is not used during training or testing.

Table 1: Simulation parameter space includes 5 unique values of RW, W, and G, and 46 distinct L_c values, for a total of 5750 unique devices.

Parameter	Minimum	Step Size	Maximum
RW	855 nm	10 nm	895 nm
W	550 nm	25 nm	650 nm
G	100 nm	50 nm	300 nm
L_c	5 μm	1 μm	50 μm

round trip of the intracavity field, from which the coupling quality factor Q_c is given as:

$$Q_c = \omega \frac{n_g^R}{c_0} \frac{2\pi R}{|\kappa|^2}, \tag{1}$$

where ω is the angular frequency, n_g^R is the group index of the ring resonator, and κ is the coupling coefficient between the microring and waveguide given by:

$$\kappa = \int_L \Gamma(\omega, l) e^{i\phi} \mathrm{d}l, \tag{2}$$

with L being the optical path over which the ring and resonator fields spatially overlap,

$\phi = l\sqrt{(\delta\beta/2)^2 + \Gamma^2}$ the relative accumulated phase term between the light propagating in the waveguide and the microring, $\delta\beta$ the propagation constant difference between the ring and the waveguide modes, and Γ the overlap of the ring mode projected onto the waveguide mode described as:

$$\Gamma(\omega, l) = \frac{i\omega}{4} \int_S (\epsilon_{\mathrm{wg}} - \epsilon_{\mathrm{R}})\, \mathbf{E}^*_{\mathbf{R}} \cdot \mathbf{E}_{\mathbf{wg}}\, \mathrm{d}r\mathrm{d}z, \tag{3}$$

with $\epsilon_{\mathrm{wg/R}}$ and $\mathbf{E_{wg/R}}$ being the dielectric permittivity and electric field of the waveguide and ring modes (in absence of each other), respectively. We focus on fundamental transverse electric (TE) polarized modes of the waveguide and ring throughout this study, though our approach (both CMT calculations and NN predictions) can be readily appliedto other cases as well. An example output of the CMT-based Q_c calculation is plotted in Fig. 1(b) for the frequency range of 200 THz to 500 THz for the design with $RW = 865$ nm, $W = 650$ nm, $G = 300$ nm, and $L_c = 38$ μm. The main goal of this work is to construct an accurate deep model that receives the geometric features and Q_c values at lower frequencies (e.g., the blue part of the curve in Fig. 1(b)) as the input and predicts the values of Q_c at higher frequencies (e.g., the red part of the curve in Fig. 1(b)). When we use traditional machine learning methods such as linear regression or random forests to achieve this goal, we observe that the accuracy is low, $\approx$ 70 %. With fully-connected or recurrent neural networks, a much higher accuracy, e.g. $\approx$ 90 %, can be obtained as discussed in the Numerical Results section below. Before we dive into such analysis, first, let us provide a brief introduction to neural networks and go over the attention mechanism[1].

[1]The dataset and all the codes to build and test the neural networks studied in this work can be found at `https://github.com/simsekergun/QcPrediction`.

A Brief Introduction to Neural Networks

A neural network is a type of machine learning algorithm that is inspired by the structure and function of the human brain. Neural networks are composed of layers of artificial neurons that are interconnected in a specific way to enable the network to learn from data. Here, a neuron refers to a computational unit that is responsible for performing a computation on the input data and producing an output. A neuron typically receives input from multiple other neurons or external sources, which are multiplied by their corresponding weights and summed together. This weighted sum is then passed through an activation function, which determines the neuron's output. The activation function is a (nonlinear) function that is responsible for introducing nonlinearity in the neural network. The output of one neuron is typically connected to the input of another neuron, creating a network of interconnected neurons. The output of the final layer of neurons is the output of the neural network.

Activation functions are typically continuous functions, which are used to introduce nonlinearity into the network, so that the network can learn more complex and intricate patterns in the input data. There are several different types of activation functions that are commonly used in neural networks, including sigmoid, Tanh, ReLU (rectified linear unit), Leaky ReLU, and softmax. The choice of activation function depends on the problem being solved and the characteristics of the data.

Neural networks follow an optimization procedure to adjust the parameters of the model during the training in order to minimize the errors between the predicted outputs and the actual outputs. The parameters of the model are then updated based on the gradient of the error function with respect to those parameters. An optimizer can be used to control how fast the iterative steps of the gradient descent algorithm are traversed in the parameter space, and can have a significant impact on the convergence and accuracy

of the neural network. There are many different optimizers available, each with their own advantages and disadvantages. Some popular optimizers include stochastic gradient descent (SGD), which updates the parameters in small batches; Adam, which adapts the learning rate based on the past gradients; and Adagrad, which adapts the learning rate for each individual parameter based on its history.

The learning rate in neural networks is a hyperparameter that determines the strength of the iterative adjustments of the weights of the network throughout the training process. Roughly speaking, the learning rate controls the step size in the gradient descent optimization algorithm, which is used to update the weights of the network during training. A high learning rate means that the weights are updated more aggressively, leading to larger changes in the model's predictions with each iteration. However, if the learning rate is too large, the weights may overshoot the optimal values, causing the model to diverge and perform poorly. On the other hand, a low learning rate means that the weights are updated more slowly and the model takes longer to converge. This slower convergence can be useful for models with many parameters, as it helps prevent the model from getting stuck in local minima of the loss function.

In the context of neural networks, an epoch refers to one single step in the iterative procedure of the gradient descent algorithm during the training process. During an epoch, the neural network updates its parameters, such as weights and biases, based on the training data it has just processed. This process allows the network to gradually learn to recognize patterns and make predictions based on the input data. After each epoch, the neural network evaluates its performance on a separate validation dataset, which helps to prevent overfitting (i.e., the network becoming too specialized to the training data and performing poorly on new, unseen data). Neural networks typically undergo multiple epochs of training, with each epoch helping to refine the network's ability to make accurate predictions. The number of epochs required depends on the complexity of the problem and the size and quality of the training dataset. Typically,

when the training loss becomes sufficiently small and stable, one can declare that the training process is complete. Having very small training and testing losses that are very close to each other is always desirable because in that case, it indicates that the neural network sufficiently learns from the data without an under or overfitting problem.

Additive Attention Mechanism

Attention mechanisms have proven to be efficient tools to construct accurate deep models when the features are given in sequential forms. For concreteness, let us assume that features are given by the sequence $\langle \boldsymbol{x}_1, \boldsymbol{x}_2, \cdots, \boldsymbol{x}_n \rangle$, where $\boldsymbol{x}_i \in \mathbb{R}^d$ for all i. As shown in Fig. 2, every attention-based model consists of two separate neural networks. The first network, which we refer to as the backbone network $F_b(\cdot)$, receives the sequence $\langle \boldsymbol{x}_1, \boldsymbol{x}_2, \cdots, \boldsymbol{x}_n \rangle$ as the input and constructs a sufficiently rich representation of it. The output of the backbone network is a new sequence $\langle \boldsymbol{h}_1, \boldsymbol{h}_2, \cdots, \boldsymbol{h}_n \rangle$ where $\boldsymbol{h}_i = F_b(\boldsymbol{x}_i)$ for all i, and $\boldsymbol{h}_i \in \mathbb{R}^{\tilde{d}}$. The dimension $\tilde{d}$ of the new vector space is in general different from the original dimension d. Typically, $\tilde{d}$ is much larger than d.

The second neural network in an attention-based model is the score network, denoted by $F_s(\cdot)$, whose output is a scalar (*i.e.* $F_s(\boldsymbol{h}_i) \in \mathbb{R}$). The score network is trained to assign scores to elements of the input sequence. The basic idea is that the score network learns to dynamically assign higher scores to those elements of the input sequence that contribute to the final output of the model more significantly. Once the scores associated with the elements of the input sequence are determined, they are then passed to the softmax function to determine a discrete probability distribution. The standard softmax function is defined by the following continuous map from $\mathbb{R}^n$ to the standard $(n-1)$-simplex Δ^{n-1}

$$\mathsf{softmax} : \mathbb{R}^n \to \Delta^{n-1} = \Big\{ (z_1, \cdots, z_n) \in \mathbb{R}^n \,|\, z_i \geq 0, \forall i \,, \sum_{i=1}^{n} z_i = 1 \Big\}$$

$$\mathsf{softmax}(z_1, \cdots, z_n) = \left(\frac{e^{z_1}}{\sum_{i=1}^{n} e^{z_i}}, \cdots, \frac{e^{z_n}}{\sum_{i=1}^{n} e^{z_i}} \right) .$$

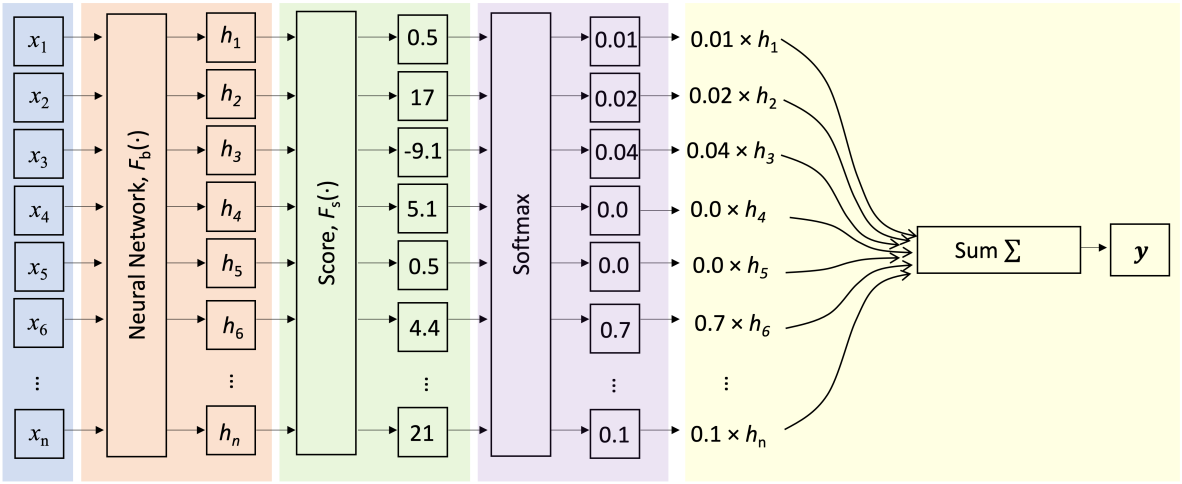

Figure 2: An illustration of the neural network implemented. In the beginning, each member of the input is equally important. A neural network is applied to each item independently, creating a new sequence representation h_i. A score function is applied to every input which gives each item a single value indicating its importance. The softmax function is applied to all the scores, creating a probability distribution. An attention mechanism multiplies each softmax score by the original item from the input and adds them all together.

Passing the attention scores to the softmax function, we can then define the weights of the discrete probability distribution by the following vector $\vec{\alpha}$

$$\vec{\alpha} = \mathsf{softmax}(F_s(\boldsymbol{h}_1), \cdots, F_s(\boldsymbol{h}_n)) = \mathsf{softmax}\big(F_s(F_b(\boldsymbol{x}_1)), \cdots, F_s(F_b(\boldsymbol{x}_n))\big) \ . \tag{4}$$

The final output of the attention-based model is then given by

$$\boldsymbol{y} = \vec{\alpha} \cdot \langle \boldsymbol{h}_1, \boldsymbol{h}_2, \cdots, \boldsymbol{h}_n \rangle = \sum_{i=1}^{n} \alpha_i \, F_b(\boldsymbol{x}_i) \ . \tag{5}$$

Score networks of attention-based models, $F_s(\cdot)$, fall into two general categories, namely the simple attention-based networks and attention-based networks with a context vector. For the former, the score, $F_s(\boldsymbol{h}_i)$, for the i-th element of the sequence solely depends on that element of the sequence. For the latter, $F_s(\boldsymbol{h}_i)$ is affected by all elements of the sequence. Since attention-based models with context incorporate all elements of the input sequence in assigning the attention scores to elements of the sequence, they perform stronger than

simple attention-based models. We define the context vector $\boldsymbol{h}_c$ to be the simple average of all elements of the sequence $\langle \boldsymbol{h}_1, \boldsymbol{h}_2, \cdots, \boldsymbol{h}_n \rangle$

$$\boldsymbol{h}_c = \frac{1}{n} \sum_{i=1}^{n} \boldsymbol{h}_i \ . \tag{6}$$

To show the dependence of the attention scores on the context vector, $\boldsymbol{h}_c$, we use the notation $F_s(\boldsymbol{h}_i; \boldsymbol{h}_c)$ to display attention scores in the presence of a context. One can design the score network in a variety of ways. Several prominent and common score networks have been enumerated in.[37] For our problem, we found that the additive attention score network performs the strongest among the existing score networks. The additive score network can be described by the following formula:[37]

$$F_s(\boldsymbol{h}_i; \boldsymbol{h}_c) = V^{\mathsf{T}}_{(1\times\tilde{d})}\, G\Big(W_{(\tilde{d}\times 2\tilde{d})}[\boldsymbol{h}_i; \boldsymbol{h}_c]\Big) \ . \tag{7}$$

In (7), $[\boldsymbol{h}_i; \boldsymbol{h}_c] \in \mathbb{R}^{2\tilde{d}}$ stands for the vector obtained by concatenating vectors $\boldsymbol{h}_i$ and $\boldsymbol{h}_c$, and $W_{(\tilde{d}\times 2\tilde{d})}$ is a fully-connected linear layer with the input dimension $2\tilde{d}$ and the output dimension $\tilde{d}$, followed by an activation function G. Finally, $V^{\mathsf{T}}_{(1\times\tilde{d})}$ in (7) represents a linear layer with the input dimension $\tilde{d}$ and the output dimension 1.

Implementation of the Attention Mechanism

In order to apply the attention mechanism to our problem, we need to organize the input of the model in a sequential form. First, we notice that for the frequency band 200 THz $\leq \nu \leq$ 500 THz (consisting of 601 distinct frequencies with step size 0.5 THz, which is small enough to catch the anti-resonances as discussed below), a finite value for Q_c has been calculated for these 601 distinct frequencies for all 5750 unique devices. Moreover, to ease the calculations, instead of working with Q_c directly (see Figure (1)-(b)), we consider

its base 10 logarithm[2]. The target variable of the model is set to be the vector $\boldsymbol{y}_i$ whose components are the values of $\log_{10} Q_c$ for the frequency band 420 THz $< \nu \leq$ 500 THz (*i.e.* high frequencies). Note that the subscript i indicates the device number, and $\boldsymbol{y}_i \in \mathbb{R}^{160}$ for all i. For the explanatory variables (*i.e.* features) of the model, we take a subset I_ν of the frequency band 200 THz $\leq \nu \leq$ 400 THz (*i.e.* lower frequencies). To organize the features of the model in the sequential form, we first divide the frequency interval I_ν into N_s uniform subintervals, $I_{\nu,j}$, where $j = 1, 2, \cdots, N_s$. Taking 20 samples over an octave-wide data is a common practice in computational electromagnetics to adequately grasp the main characteristics of spectral features.[36] Since we deal with a slightly wider band, we set N_s to 25. For each subinterval $I_{\nu,j}$, we collect the values of $\log_{10} Q_c$ in a vector $\boldsymbol{q}_{i,j} \in \mathbb{R}^T$, where the subscript i indicates the device number, subscript j refers to the subinterval $I_{\nu,j}$, and $T = N_\nu/N_s$, where N_ν is the number of frequencies at which we calculate the Q_c values to be used in the training data. We then concatenate the geometric features (in Table (1)) with each $\boldsymbol{q}_{i,j}$

$$\tilde{\boldsymbol{x}}_{i,j} = \left[\langle RW, W, G, L_c\rangle_i \,; \boldsymbol{q}_{i,j}\right] , \tag{8}$$

where $\tilde{\boldsymbol{x}}_{i,j} \in \mathbb{R}^{T+4}$. For each device i, we can then arrange a sequence of input variables of the form $\langle \tilde{\boldsymbol{x}}_{i,1}, \tilde{\boldsymbol{x}}_{i,2}, \cdots, \tilde{\boldsymbol{x}}_{i,N_s}\rangle$. All these data can be represented through a $(B, N_s, T+4)$-tensor $\mathbb{X}$, where the first index specifies the device (*i.e.* the batch number). Note that the last dimension is $T+4$, which corresponds to the n in Fig. 2 such that $n = T+4$, because in each case, we input 4 parameters (RW, W, G, and L_c) and T values of Q_c calculated at T different frequencies. As is customary in deep learning, the input data can be fed in two different ways, namely through $\mathbb{X}$ or $\mathbb{X}^\mathsf{T}$. We note that $\mathbb{X}^\mathsf{T}$ is the transpose of $\mathbb{X}$ along its last two dimensions, and hence is a $(B, T+4, N_s)$-tensor. When $\mathbb{X}$ is fed to the attention-based model, the attention is inclined towards the frequency subintervals, whereas for $\mathbb{X}^\mathsf{T}$, the attention is inclined towards the feature types. We found that for our problem, the

[2]There are some missing data points in the original datasets. Those are replaced with zeros after taking logarithms of Q_c values

attention-based models perform much stronger when $\mathbb{X}^{\mathsf{T}}$ is fed to the model as the input.

The last ingredient to specify is the exact structure of the backbone network $F_b(\cdot)$ for the problem at hand. In our study, the backbone network consists of 4 hidden layers with 256, 256, 512, and 512 neurons. The input of each layer passes a batch normalization before reaching the fully-connected linear layer. The output of each hidden layer passes a standard ReLU (rectified linear unit) activation function.

Numerical Results

Interpolation

First, we consider the following scenario: the four geometric parameters $\langle RW, W, G, L_c \rangle$ and Q_c values for the frequency range 200 THz to 400 THz are given as the input of the model, and our goal is to predict the Q_c values for 160 distinct frequencies ranging from 420 THz to 500 THz, i.e., $I_\nu : 200 \leq \nu \leq 400$ THz and $\boldsymbol{y} : 420$ THz $< \nu \leq 500$ THz. We randomly split the dataset into two disjoint subsets so that 4750 of the 5750 distinct designs are used for training and the remaining 1000 designs are used for testing.

To calculate the accuracy of the attention-based neural network, we use the standard mean squared error (MSE) as the loss function. The learning rate is set to 0.001. The standard ReLU is taken as the choice of the activation function G in (7) under the AdamW optimizer.[39] At the end of 100 epochs, the test R^2-score and the test loss reach (96.7 ± 0.4) % and (3.6 ± 0.4) %, respectively. Figure 3 shows comparisons of the true (solid curves) vs. the predicted (dashed curves) behavior of Q_c for four randomly chosen devices. There are a few comments in order. In the upper plot of Figure (3), neither design has a peak in the true Q_c values in the selected frequency range. In this case, the prediction of the attention-based network agrees with the ground truth values of Q_c for the selected designs with a remarkably high level of precision. However, this is not true for the devices chosen for the second plot in Figure (3), where we observe an anti-resonance in the high-frequency range.

This anti-resonance is a location where Q_c exhibits a (sharp) peak and the coupling coefficient (κ) between the resonator and waveguide tends towards zero, due to a perfect anti-phase condition in Eq. 2. In such cases, the attention-based model accurately identifies the position of the peak, but the model typically predicts a shorter height for the peak. This behavior of the model can be explained by noting that a (sharp) peak in the behavior of Q_c represents an extreme nonlinear behavior. Neural networks, on the other hand, incorporate nonlinearity through well-behaved activation functions which all fall into the class of regularly varying functions. In order to capture the height of the peaks accurately, the neural network must be capable of incorporating extreme nonlinear behaviors of the target variable but this is not necessary from a photonic design point of view because the most important point is that Q_c is very high, rather than its specific value. In particular, for a resonator coupled by a single bus waveguide, the power coupled into the resonator P_c is given by:[42]

$$P_c = P_{in} \frac{4Q_i/Q_c}{(1 + Q_i/Q_c)^2}, \tag{9}$$

where P_{in} is the input power in the waveguide and Q_i is the resonator's intrinsic quality factor, i.e., the quality factor in absence of waveguide coupling and due to other loss mechanisms, such as scattering and absorption. As intrinsic quality factors for these types of resonators are typically limited to $\lesssim 10^8$,[43] from a practical point of view, once the Q_c exceeds the intrinsic loss by more than a couple orders of magnitude, there is not much need to distinguish between further orders-of-magnitude increases since the power coupled into the resonator is nearly zero regardless. Thus, although the model is not capable of capturing the anti-resonance behavior completely, it is capable of estimating where Q_c exceeds the intrinsic loss by more than an order of magnitude or so. This is a more important attribute from the design perspective, where an understanding of the location of the coupling anti-resonances is needed to ensure that they are in regions of frequency space where efficient coupling is not needed.[30]

We now comment on the choice of the optimizer for our neural networks. One of the most

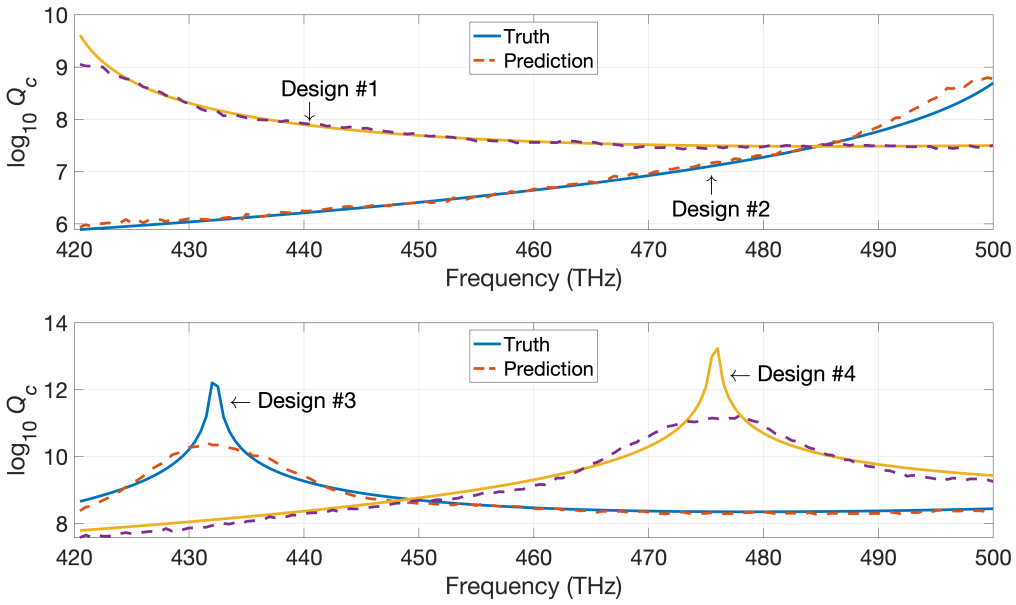

Figure 3: True (CMT-determined) vs. predicted Q_c values for two devices without (top) and with (bottom) a peak (referred in the main text as a coupling anti-resonance) in the frequency range of interest. Here, the geometric parameters defining the resonator-waveguide coupler and the Q_c values between 200 THz and 400 THz are given as inputs in order to predict Q_c between 420 THz and 500 THz.

commonly used optimizers in deep learning has been the Adaptive Momentum Estimation (Adam), which was first introduced in.[38] In updating the parameters of the model at each step of the learning process, Adam considers the variance of the gradients and normalizes the momentum with its variance. However, it was shown in[39] that in order to implement the regularization of Adam in a consistent manner, the weight decay of the learning process must be decoupled from the gradient-based update. This variant of Adam optimizer is known as the AdamW optimizer. Another variant of Adam – known as the Nesterov Adam (NAdam) that was introduced in[40] – incorporates the Nesterov momentum instead of the usual velocity vector in the parameter space of the model. In certain cases, NAdam outperforms Adam and AdamW. A relatively recent paper,,[41] introduces another variant of the Adam optimizer known as the Rectified Adam (RAdam). In Adam (also in AdamW and NAdam), in order to calculate the variance of the gradients, one simply approximates the variance with the square of the gradients. In RAdam, however, a deeper analysis of the variance is performed,

and the variance of the gradients is modified[41] by a factor r_t at each timestamp t of the learning process. The inclusion of the rectified factor, r_t, leads to more stable performance of adaptive learning. In this work, we consider three of the aforementioned variants (AdamW, NAdam, and RAdam) of the Adam optimizer as different choices for the optimizer of our neural networks. Table 2 summarizes the mean R^2 score of the model (calculated across 1000 randomly chosen devices that are not used during the training) for the train and test subsets under each choice of the optimizer. To calculate the uncertainty associated with the result, we have executed the network over five randomly train/test splits. The uncertainty associated with the R^2 scores has been depicted in Figure (4), where the uncertainty is a one standard deviation value.

Table 2: Accuracies of the attention-based model under different choices of optimizers.

	Accuracies of the Model Under Different Optimizers	
optimizer	train R^2-score (%)	test R^2-score (%)
AdamW	94.0 $\pm$ 0.4	96.7 $\pm$ 0.4
NAdam	93.6 $\pm$ 0.4	96.4 $\pm$ 0.4
RAdam	93.7 $\pm$ 0.4	96.6 $\pm$ 0.4

It is observed that the performances of AdamW, NAdam, and RAdam are very close to each other but the first choice (AdamW) offers a slightly higher R^2-scores for the train and test subsets. Hence, AdamW is fixed as our preferred choice of optimizer for the rest of the work carried out in this paper.

In our setup, the target variable is a vector with 160 components ($\boldsymbol{y} \in \mathbb{R}^{160}$). An important question in this context is how the accuracy of the predictions depends on the frequency bandwidth (*i.e.* length of I_ν) for the feature variables. To provide a quantitative answer to the dependence of the accuracy of the model on the width of the frequency band I_ν, we proceed as follows. We execute the model for four different frequency bands. The first frequency band ($I_\nu^{(1)}$: 200 THz $\leq \nu <$ 250 THz) consists of 100 distinct frequencies, and any

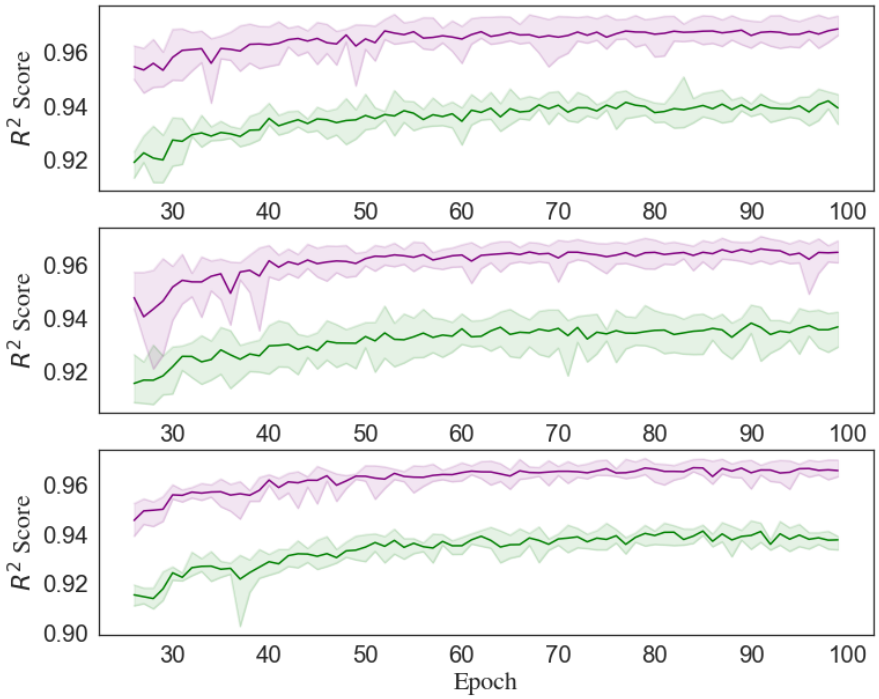

Figure 4: The R^2 scores for the attention-based neural network under different choices of optimizers have been depicted. The top, middle and bottom plots depict R^2 scores for the AdamW, NAdam, and RAdam optimizers, respectively. The green curve represents the mean R^2 score for the train dataset, while the purple curve represents the mean R^2 score of the test data per epoch. The semi-transparent filled regions associated with the green and purple curves represent the uncertainties in R^2 scores per epoch for five randomly formed train/test splits.

subsequent band increases the number of distinct frequencies by including 100 new points. In this study, in addition to the attention-based NN, we also implement two other architectures: FCNN and RNN. The FCNN consists of 6 hidden layers. The first two, the second two, and the last two layers of the FCNN model include N_n, $2N_n$, and $4N_n$ neurons respectively, where $N_n = 512$. As before, the standard ReLU is used as the choice of the activation function for hidden layers. The optimizer is set to AdamW. The step size, learning decay rate, and learning rate are chosen to be 10, 0.3, and 0.001, respectively. The RNN model consists of 5 layers with 256 neurons for each layer. The step size, learning decay rate, and learning rate are chosen to be 10, 0.5, and 0.001, respectively. The sequence length is 25. Figure 5 plots the accuracy (test R^2-scores) for FCNN, RNN, and an attention-based NN model. For

long input sequences, all three networks have high accuracies, $\geq 95\%$. When the sequence length gets shorter, the accuracy of the RNN implementation drops much faster than the FCNN and attention-based NN. It should be noted that even for the shortest interval that we consider here (*i.e.* $I_\nu^{(1)}$), the attention-based NN model performs fairly well, with $R^2 \approx 88$ %. It is not shown here for the sake of brevity but we can predict the Q_c values from 250 THz to 500 THz almost at the same accuracy level with $I_\nu^{(1)}$: $200 \leq \nu < 250$ THz. This means that once we complete the training, we need to spend n-hours of computing time instead of $6n$-hours and we ask neural networks to predict the Q_c values at higher frequencies, which takes only a few milliseconds. This 83 % reduction in computing time could be a tremendous saving, especially for computationally intensive studies and applications.

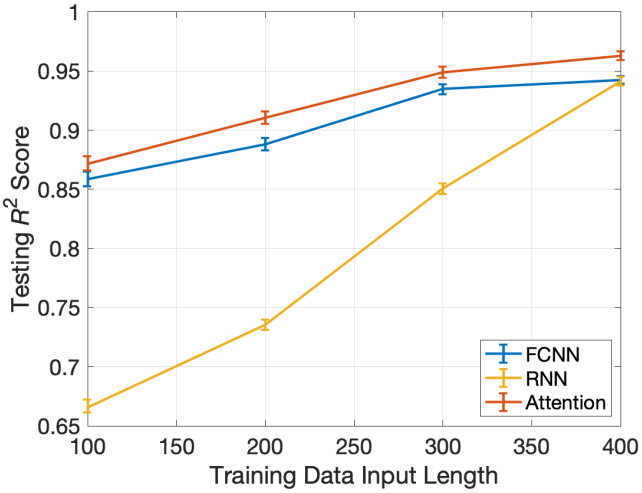

Figure 5: The R^2 scores for the FCNN (blue), RNN (red), and the attention-mechanism (orange) models for the interpolation problem with different frequency bandwidths for the feature variables. Error bars represent standard errors. From $I_\nu^{(1)}$: 200 THz $\leq \nu <$ 250 THz to $I_\nu^{(4)}$: 200 THz $\leq \nu <$ 400 THz, corresponding to training datasets with the increasing length of input from 100 to 400.

Based on the results depicted in Figure (5), we confidently claim that the implementation of an attention mechanism indeed increases the accuracy of RNN-type neural networks for interpolation problems. However, when we take into account the training time spent by each of the established models – 3 minutes for the FCNN and an hour for the attention-based

NN using the 32 GPUs that come with Apple's M1 Max processor [3] – the FCNN could be a more practical choice for the interpolation applications that must be completed in short periods of time, where the lack of extremely high accuracy is not a crucial matter. As we increase the training dataset size from $I_\nu^{(1)}$ to $I_\nu^{(4)}$, the amount of memory used is increased from 0.8 GB to 1.2 GB.

Note that for the method outlined above, the purpose is neither the inverse photonic design nor extremely accurate Q_c prediction. If the goal is the inverse design of a ring resonator with a pulley-type coupling, then the proper choices would be GANs and BtB networks to determine the optimum parameter. If the goal is to efficiently couple the light at multiple widely separated wavelengths from a waveguide to a ring resonator, then using the adjoint method to identify scatterers between the ring resonator and the waveguide, such as it is done in,[10] would be an appropriate choice. If the goal is to determine the coupling efficiency with very high accuracy, then one needs to calculate the electric field distributions inside the waveguide and ring resonator and follow the CMT formalism for the desired set of design parameters (dimensions, refractive indices of the materials, wavelengths, etc.). We emphasize that when the goal is to design a ring resonator with a pulley-type coupling via numerical optimization or a grid-based approach, appropriately chosen machine learning algorithms can accelerate this process. One should calculate the entire Q_c spectrum (or in practice, M_1 unique values) for N devices. Then for the rest, one can rely on machine learning algorithms and calculate Q_c values for M_2 values only, where $M_1 \ll M_2$ for as many as new devices one desires, as long as the physical parameters of these new devices are covered in the training dataset. Roughly speaking, one obtains a reduction by a factor of M_2/M_1 in the overall computation time.

[3] Certain commercial items are identified to foster understanding, and do not constitute an endorsement by NIST.

Extrapolation

To study extrapolation, we change the way we create our training and test datasets. Here instead of a random splitting, we use certain values of RW and W for training and for testing, we choose RW and W values that are beyond the ranges used in training. Table 3 lists the corresponding RW and W values. In this case, the training and testing datasets have 3680 and 230 devices, respectively.

Table 3: RW and W values used for the training and testing datasets for the extrapolation problem.

Parameter	Training Dataset	Testing Dataset
RW (nm)	855, 865, 875, 885	895
W (nm)	550, 575, 600, 625	650

Figure 6 shows comparisons of the true (solid curves) vs. the predicted (dashed curves) behavior of Q_c for four randomly chosen devices among the 230 test devices used in the extrapolation study. Similar to the results presented in Fig. 3, the accuracy is high in the absence of an anti-resonance peak and when there is a peak, the model is capable of estimating the anti-resonance width (defined, for example, as the frequency range with $Q_c > 10^{10}$) accurately.

Figure 7 plots the accuracy (test R^2-score) for FCNN, RNN, and an attention-based NN in a similar fashion to Figure 5. Again, the accuracy of FCNN and attention-based NN increases steadily with increasing training dataset size and the attention-based NN outperforms the FCNN when the training data is sufficiently large. However, we do not observe a similar steadiness in the performance of the RNN model. In fact, the performance of the RNN model does not improve until the number of samples (devices) used in the training reaches at least 300.

Unlike the interpolation problem, here we have to use a large number of samples in the training data to achieve high accuracy in the context of extrapolation problems. Due to

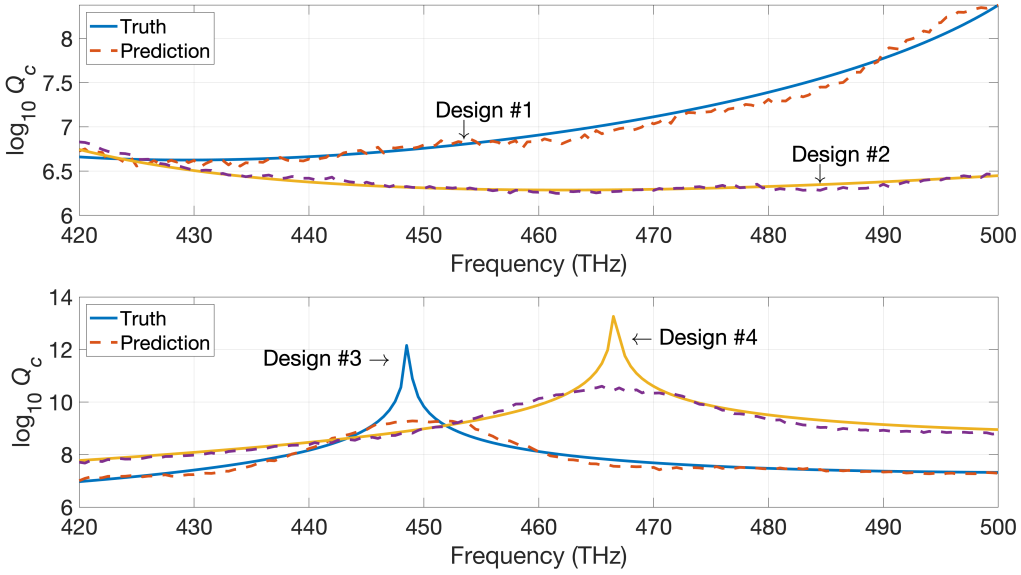

Figure 6: True (CMT-determined) vs. predicted Q_c values for two devices without (top) and with (bottom) a peak (referred in the main text as a coupling anti-resonance) in the frequency range of interest. In contrast to Fig. 3, here we perform an extrapolation study where training is done using CMT-based Q_c calculations for certain values of RW and W, and testing is done for RW and W values outside of the testing range.

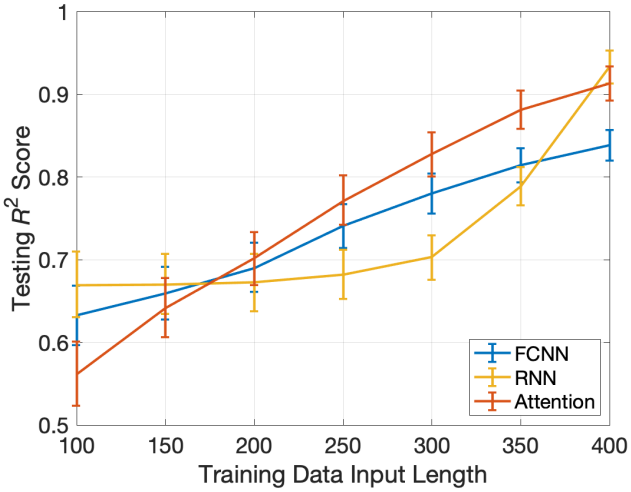

Figure 7: The R^2 scores for the FCNN (blue), RNN (red), and the attention-mechanism (orange) models for the extrapolation problem with different frequency bandwidths for the feature variables. Error bars represent the standard deviation observed in the last five epochs.

its stable performance, we again recommend attention-based NN models for extrapolation applications that typically require high accuracy.

If we take the case where we use the 200 THz to 400 THz dataset for training and achieve to predict the Q_c values for the 400 THz to 500 THz range with a 91 % accuracy as an example, we can conclude that there is a 33 % reduction in computation time. This is not a reduction as significant as the one we see in the interpolation problem. Considering the fact that if one chooses geometric parameters far from the ones used for the training, then the accuracy is very likely to decrease for this kind of non-linear device characterization. The amount of decrease highly depends on the correlation between the input parameter and output, which is the Q_c value in our case. A safer approach would be preparing training data that (i) spans the entire testing design space and (ii) has the parameters that are sampled at moderate-enough rates. The rules regarding how to choose the optimum sampling rate are beyond the scope of this paper and the reader is referred to.[44] The amount of possible reduction in computing time while maintaining accuracy heavily depends on the non-linearity level of the design under investigation. The training data needs to be large enough to learn the influence of each feature on the output and wide enough (in the frequency domain) to carry the unique signature of each device. In our interpolation study, a quarter-octave was enough to distinguish these moderately nonlinear structures but for devices with a weaker nonlinearity, training data covering smaller fractions of an octave could be enough and an even higher amount of acceleration could be achieved.

In our future studies, we plan to add new parameters such as tapering of the gap and waveguide to the parameter space and use attention-based neural networks as a forward-solver feeding a numerical optimization algorithm to achieve qualitatively new coupling regimes beyond what has thus far been shown with constant waveguide width and constant gap pulley couplers. In particular, rather than demonstrating desirable coupling levels in targeted spectral bands within the octave bandwidth of a microresonator frequency comb,[30] the ability to flatten the Q_c across the entire octave will be studied.

Conclusion

In this work, we employ a number of well-known deep learning architectures such as the fully-connected and recurrent neural networks to establish successful predictive models for the behavior of the frequency-dependent coupling quality factor of pulley-coupled microring resonators. In addition to the aforementioned architectures, we also take the advantage of a more recent deep learning architecture, namely the attention mechanism, in order to establish deep learning models with even higher accuracies. Due to its recentness, the implementation of attention mechanisms to construct predictive deep learning models is explained in detail. Predictions of the established models are treated as either interpolation or extrapolation problems, based on how the training and testing datasets are arranged. In both situations (*i.e.* interpolation and extrapolation), and among the constructed deep models in this study, the attention mechanism yields the most accurate and stable results in predicting the coupling quality factor of the pulley-coupled microring resonators. We explored the dependence of the precision of the predicted results on the bandwidth of the input frequencies and found that the accuracy of the attention-based models increases steadily with increasing the size of the training dataset. We numerically show that for the interpolation problem, the attention mechanisms can achieve predicting the coupling efficiency with over 90 % accuracy for spectral ranges six times wider than the spectral ranges used in training data, which means a six times reduction is possible during a large-scale numerical optimization study. We conclude that once models are trained with sufficiently large datasets, the deep learning models can offer a new promising avenue to accelerate spectral studies in electromagnetics, photonics, and acoustics, as the approach advertised in this work is physics-agnostic and can thus be applied to a wide range of problems.

Acknowledgements

The authors would like to express their gratitude to anonymous reviewers, Dr. Vladimir Aksyuk, Dr. Jordan Stone, and Logan Courtright for their useful comments on earlier versions of this paper.

Funding Sources

Gregory Moille and Kartik Srinivasan acknowledge support from the NIST-on-a-chip and DARPA APHI programs. C. R. M acknowledges support from the AFOSR grant (FA9550-19-S-0003).

References

(1) Jiang, J.; Chen, M.; Fan, J. A. Deep neural networks for the evaluation and design of photonic devices. *Nat. Rev. Mater.* **2021**, *6*, 679–700.

(2) Ma, W.; Liu, Z.; Kudyshev, Z. A.; Boltasseva, A.; Cai, W.; Liu, Y. Deep learning for the design of photonic structures. *Nat. Photonics* **2021**, *15*, 77–90.

(3) Piccinotti. D.; MacDonald, K. F.; Gregory, S. A.; Youngs, I.; Zheludev, N. I. Artificial intelligence for photonics and photonic materials. *Rep. Prog. Phys.* **2021**, *84*, 012401.

(4) Lalau-Keraly, C. M.; Bhargava, S.; Miller, O. D.; Yablonovitch, E. Adjoint shape optimization applied to electromagnetic design. *Opt. Express* **2013**, *21*, 21693–21701.

(5) Su, L.; Piggott, A. Y.; Sapra, N. V.; Petykiewicz, J.; Vucković, J. Inverse design and demonstration of a compact on-chip narrowband three-channel wavelength demultiplexer. *ACS Photon* **2017**, *5*, 301–305.

(6) Molesky, S., Lin, Z., Piggott, A.Y.; Jin, W.; Vucković, J.; Rodriguez, A. W. Inverse design in nanophotonics. *Nature Photon,* **2018**, *12*, 659–670.

(7) Hughes, T. W.; Minkov, M.; Williamson, I. A. D.; Fan, S. Adjoint Method and Inverse Design for Nonlinear Nanophotonic Devices. *ACS Photonics* **2018**, *5 (12)*, 4781–4787.

(8) Wang, K.; Ren, X.; Chang, W.; Lu, L.; Liu, D.; Zhang, M. Inverse design of digital nanophotonic devices using the adjoint method. *Photon. Res.* **2020**, *8*, 528–533.

(9) Yeung, C.; Ho, D.; Pham, B.; Fountaine, K. T.; Zhang, Z.; Levy, K.; Raman, A. P. Enhancing Adjoint Optimization-Based Photonic Inverse Design with Explainable Machine Learning. *ACS Photonics* **2022**, *9 (5)*, 1577–1585.

(10) Jin, W.; Molesky, S.; Lin, Z.; Fu, K.-M. C., Rodriguez, A. W. Inverse design of compact multimode cavity couplers, *Opt. Express* **2018**, *26*, 26713–26721.

(11) Jiang, J.; Sell, D.; Hoyer, S.; Hickey, J.; Yang, J.; Fan, J. A. Free-Form Diffractive Metagrating Design Based on Generative Adversarial Networks. *ACS Nano* **2019**, *13 (8)*, 8872–8878.

(12) Tang, Y.; Kojima, K.; Koike-Akino, T.; Wang, Y.; Wu, P.; Xie, Y.; Tahersima, M. H.; Jha, D. K.; Parsons, K.; Qi, M. Generative deep learning model for inverse design of integrated nanophotonic devices, *Laser & Photonics Rev.* **2020**, *2000287*.

(13) Peurifoy, J.; Shen, Y.; Jing, L. *et al.* Nanophotonic particle simulation and inverse design using artificial neural networks. *Science Advances* **2018**, *4 (6)*, eaar4206.

(14) Christensen, T.; Loh, C.; Picek, S.; Jakobović, D.; Jing, L.; Fisher, S.; Ceperic, V.; Joannopoulos, J. D.; Soljačić, M. Predictive and generative machine learning models for photonic crystals, *Nanophotonics* **2020**, *9 (13)*, 4183–4192.

(15) Chugh, S.; Gulistan, A.; Ghosh, S.; Rahman, B. M. A. Machine learning approach for computing optical properties of a photonic crystal fiber. *Opt. Express* **2019**, *27*, 36414–36425.

(16) Singh, R.; Agarwal, A.; Anthony, B. W. Mapping the design space of photonic topological states via deep learning. *Optics Express* **2020**, *28 (19)*, 27893–27902.

(17) Li, R.; Gu, X.; Li, K.; Li, Z.; Zhang, Z. Predicting the Q factor and modal volume of photonic crystal nanocavities via deep learning. in *Proc. SPIE 11903, Nanophotonics and Micro/Nano Optics VII* **2021**, 1190305.

(18) Sharabi, Y.; Patsyk, A.; Ziv, R.; Segev, M. Deep Learning Method for Quantum Efficiency Reconstruction. *2021 Conference on Lasers and Electro-Optics (CLEO 2021)*, paper STh4J.7.

(19) Simsek, E.; Mahabadi, S. E. J.; Carruthers, T. F.; Menyuk, C. R. Photodetector Performance Prediction with Machine Learning. in *Frontiers in Optics + Laser Science (FiO+LS 2021*, paper FTu6C.4.

(20) Nishizaki, Y.; Valdivia, M.; Horisaki, R.; Kitaguchi, K.; Saito, M.; Tanida, J.; Vera, E. Deep learning wavefront sensing. *Opt. Express* **2019**, *27*, 240–251.

(21) Ashtiani, F.; Geers, A.J.; Aflatouni, F. An on-chip photonic deep neural network for image classification. *Nature* **2022**, *606*, 501–506.

(22) Karanov, B.; Chagnon, M.; Thouin, F.; Eriksson, T. A.; Bülow, H.; Lavery, D.; Bayvel, P.; Schmalen, L. End-to-End Deep Learning of Optical Fiber Communications, *J. Lightwave Technol.* **2018**, *36*, 4843–4855.

(23) Liu, D.; Tan, Y.; Khoram, E.; Yu. Z. Training Deep Neural Networks for the Inverse Design of Nanophotonic Structures. *ACS Photonics* **2018**, *5 (4)*, 1365–1369.

(24) An, S.; Fowler, C.; Zheng, B. *et al.* A Deep Learning Approach for Objective-Driven All-Dielectric Metasurface Design. *ACS Photonics* **2019**, *6 (12)*, 3196–3207.

(25) Bahdanau, D.; Cho, K.; Bengio, Y. Neural Machine Translation by Jointly Learning to Align and Translate. *arXiv preprint arXiv:1409.0473,* **2014**.

(26) Bogaerts, W.; De Heyn, P.; Van Vaerenbergh, T.; De Vos, K.; Kumar Selvaraja, S.; Claes, T.; Dumon, P.; Bienstman, P.; Van Thourhout, D.; Baets, R. Silicon microring resonators. *Laser & Photon. Rev.* **2012**, *6*, 47–73.

(27) Rabus, D. G.; Sada, C. *Integrated Ring Resonators: A Compendium,* 2nd ed.; Springer, 2020.

(28) Chin, M. K.; Ho, S. T. Design and modeling of waveguide-coupled single-mode microring resonators. *J. Light. Technol.* **1998**, *16 (8)*, 1433–1446.

(29) Hosseini, E. S.; Yegnanarayanan, S.; Atabaki, A. H.; Soltani, M.; Adibi, A. Systematic design and fabrication of high-Q single-mode pulley-coupled planar silicon nitride microdisk resonators at visible wavelengths. *Opt. Express* **2010**, *18*, 2127–2136.

(30) Moille, G.; Li, Q.; Briles, T. C.; Yu, S.-P.; Drake, T.; Lu, X.; Rao, A.; Westly, D.; Papp, S. B.; Srinivasan, K. Broadband Resonator-Waveguide Coupling for Efficient Extraction of Octave Spanning Microcombs. *Opt Lett.* **2019**, *44 (19)*, 4737–4740.

(31) Kippenberg, T. J.; Gaeta, A. L.; Lipson, M.; Gorodetsky, M. L. Dissipative Kerr solitons in optical microresonators. *Science* **2018**, *361 (6402)*.

(32) Diddams, S. A.; Vahala, K.; Udem, T. Optical frequency combs: Coherently uniting the electromagnetic spectrum. *Science* **2020**, *369 (6501)*.

(33) Moille, G.; Westly, D. A.; Perez,E. F.; Metzler, M.; Simelgor,G.; Srinivasan, K. Integrated Buried Heaters for Efficient Spectral Control of Air-Clad Microresonator Frequency Combs. *APL Photonics* **2022**, *7 (121604)*.

(34) Sherstinsky, A. Fundamentals of Recurrent Neural Network (RNN) and Long Short-Term Memory (LSTM) network. *Physica D: Nonlinear Phenomena* **2020**, *404 (132306)*.

(35) Virot, L. Ultrafast on-chip germanium photodiode. *Nat. Photon.* **2021**, *15*, 868–869.

(36) Simsek, E. Machine Learning Exercises on One Dimensional Electromagnetic Inversion. *IEEE Trans. on Anten. and Prop.,* **2021**, *69 (10)*, 6797–6805.

(37) Raff, E. *Inside Deep Learning, Math, Algorithms, Models*; Manning Publications, 2021.

(38) Kingma, D.; Ba, J. Adam: A Method for Stochastic Optimization. *arXiv preprint arXiv:1412.6980,* **2014**.

(39) Loshchilov, I. ; Hutter, F. Decoupled Weight Decay Regularization. *arXiv preprint arXiv:1711.05101,* **2017**.

(40) Dozat, T. Incorporating Nesterov Momentum into ADAM, in *Proc. 4th Int. Conf. Learn. Represent. (ICLR 2016) Workshop,* San Juan, Puerto Rico.

(41) Liu, L.; Jiang, H.; He, P.; Chen, W.; Liu, X.; Gao, J.; Han, J. On the Variance of the Adaptive Learning Rate and Beyond. *arXiv preprint arXiv:1908.03265,* **2019**.

(42) Manolatou, C.; Khan, M. J.; Fan, S.; Villeneuve, P. R.; Haus, H. A.; Joannopoulos, J. D. Coupling of modes analysis of resonant channel add-drop filters. *IEEE J. Quantum Electron.* **1999**, *35 (9)*, 1322–1331.

(43) Ji, X.; Roberts, S.; Corato-Zanarella, M.; Lipson, M. Methods to achieve ultra-high quality factor silicon nitride resonators. *APL Photonics* **2021**, *6 (071101)*.

(44) Katharopoulos, A.; Fleuret, F. Not All Samples Are Created Equal: Deep Learning with Importance Sampling. *arXiv preprint arXiv:1803.00942,* **2018**.