Functional Neural Networks

Approximated by Weierstrass Polynomials

William Guss

Candidate #000345-0022

May 2015

Mathematics

Nathaniel Nelson

Word Count: 3900

Abstract

In this paper we consider the traditional model of feed-forward neural networks proposed in (McCulloch and Pitts, 1949), and using intuitions developed in (Neal, 1994) we propose a method generalizing discrete neural networks as follows. In the standardized case, neural mappings $\mathscr{N} : \mathbb{R}^n \to [0,1]^m$ have little meaning when $n \to \infty$. Thus we consider a new construction $\mathscr{F} : \mathscr{X} \to \mathscr{Y}$ where the domain and co-domain of $\mathscr{N}$ become infinite dimensional Hilbert spaces, namely the set of quadratically Lebesgue integrable functions L^2 over a real interval E and $[0,1]$ respectively. The derivation of this construction is intuitively similar to that of Lebesgue integration; that is, $\sum_i \sigma_i w_{ij} \to \int_{E\subset\mathbb{R}} \sigma(i) w(i,j)\, d\mu(s)$.

After establishing a proper family of "functional neural networks" $\mathscr{F}$, we show that $\mathscr{N}$ are a specific class of functional neural networks under specific constraints. More specifically in our first lemma, we prove that $\mathscr{F} \equiv \mathscr{N}$ for piecewise constant weight functions $w(i,j)$. Having done so, we then attempt to find an analogue to Cybenko's theorem of universal approximation for neural networks. Firstly, we prove as a corollary of the Weierstrass approximation theorem, that $w(i,j)$ can approximate with arbitrary precision a function satisfying $\|\mathscr{F}\xi - f(\xi)\|_\infty \to 0$ for arbitrary $f : E \to [0,1]$. As a byproduct of the proof, we also establish a closed-form definition for the satisfying $w(i,j)$ and thereby through our first lemma provide novel insight into the actual form of the weight matrix $[w_{ij}]$ for trained $\mathscr{N}$. Finally we propose a universal approximation theorem for functional neural networks; that is, we show through the Riesz Representation Theorem that $\mathscr{F}$ approximates any bounded linear operator on $\mathscr{X}$.

In conclusion, we create a practical analogue of the error-backpropagation algorithm, and implement functional neural networks using Simpson's rule. We suggest that functional neural networks represent an interesting opportunity for the implementation of machine learning systems modeling functional transformation.

Word Count: 100

Contents

1 Introduction

Machine learning is a highly interdisciplinary field which deals with the development of algorithms that can predict and classify novelties based on a set of prior "intuitions" [1]. The field itself employs research from biology, computer science, numerical analysis, and statisitics. In recent years applications of machine learning can be seen holistically throughout our society in web services like Google, Facebook, and Amazon to name a few. Seeing as there is incentive from both private industry and academia, machine learning is ever expanding and developing as an integral field of mathematical and scientific inquiry.

One of the most biologically inspired set of algorithms developed in the field is the artificial neural network (ANN). Although there are numerous mathematical interpretations of neural networks, we will primarily focus on the expansion of one such interpretation, feed-forward neural networks. In order to understand this specific interpretation, some biological foresight is required.

1.1 Biological Neurons

A single neuron consists largely of the cell body or soma, the dendrites, and the axon. Mathematically we wish to examine the process of neural activation, namely the events which lead to the excitation of the axon. Consider a neuron whose anterior neurons (those which are connected dendritically) are activated; that is, the neuron is receiving input along all of its dendrites. These electrical inputs propagate through the dendrites and then become integrated on the Soma as electrical membrane potential [2]. The soma then acts as the primary computational unit and activates the axon when a threshold of input activity is reached. More specifically, when a memberane potential of about -60 mV is reached on the soma, the hillock zone, or axon hillock, activates the axon by applying proteins to an ion channel which creates action potential along the axon [3].

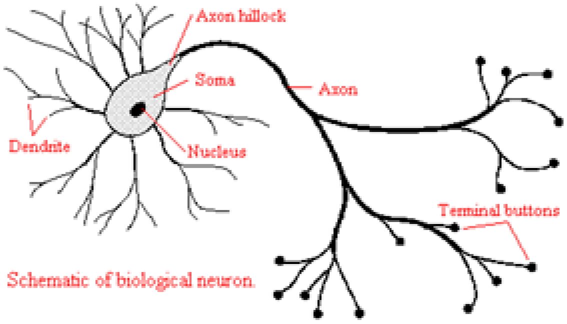

Figure 1: A biological neuron [4].

1.2 Artificial Neurons

With this in mind it is now possible to construct a mathematical model of an artificial neuron. Let A_j,P_j be the set of anterior and posterior neurons of a neuron, j. Then, the cell membrane naturally becomes a linear combination of the dendritic potentials.

Definition 1.1. *We say that* net_j *is the net electric potential over the membrane, if for a natural neural resting potential* β*,*

$$net_j = \sum_{i \in A_j} w_{ij}\sigma_i + \beta$$

where w_{ij} *is the dendritic connection "strength" from the* i^{th} *anterior neuron to* j*,* σ_i *is the action potential being propagated from the* i^{th} *anterior neuron.*

Furthermore, the thresholding of the hillock zone is given by some real valued sigmoidal function g bijective and differentiable over $\mathbb{R}$.

Definition 1.2. *We call* σ_j *the action potential of a neuron* j *if*

$$\sigma_j = g\left(\mathrm{net}_j\right)$$

for some continuous real valued, monotonically increasing function g*.*

This model of the artifical neuron follows from the work that Pitt and McCulloch did in

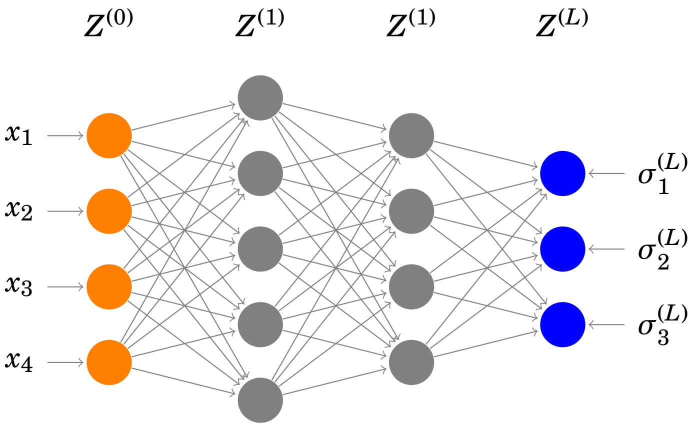

Figure 2: An example of a feed-forward ANN, $\mathscr{N}$ with four layers.

representing neural activity as logical thresholding elements [5].

1.3 Feed-Forward Artificial Neural Networks

Now we have sufficient mathematical basis to define the feed-forward artificial neural network. The concept of a feed-forward ANN is biologically motivated by the functional organization of the visual cortex. It is appropriate to divide the structure of the visual cortex into layers which are denoted V1, V2, V3, and so on. The layers are organized such that a given layer is directly adjacent to and exhibiting full connectedness to the subsequent layer, an example being V1 to V2, V2 to V3, and subsequently for all of the primary layers of the visual cortex. From a functional point of view these layers store levels of visual abstraction like lines and shapes on the lower layers to faces and abstract visual concepts on the highest layers[6].

The goal then of describing a feed-forward artificial neural network is to model this representation of increasing abstraction whilst maintaing adjacency and full topological connectedness. Thus we construct a set of neural layers with cardinality $L+1$, and connections as depicted in Figure 2.

Definition 1.3. *We say $\mathcal{N}$ is a feed-forward neural network if for an input vector* $\boldsymbol{x}$*,*

$$\begin{aligned}\mathcal{N} : \sigma_j^{(l+1)} &= g\left(\sum_{i\in Z^{(l)}} w_{ij}^{(l)}\sigma_i^{(l)} + \beta^{(l)}\right)\\ \sigma_j^{(1)} &= g\left(\sum_{i\in Z^{(0)}} w_{ij}^{(0)}x_i + \beta^{(0)}\right)\end{aligned}$$

where $1 \le l \le L-1$.

For mathematical convenience let us denote $\sigma_j^{(l)}$ as the output of the j^{th} neuron on layer l. In this construction we prefer three different types of neurons, the input neuron, the hidden neuron, and the output neuron. In the case of the input neuron, there is no sigmoidal activation function, and instead we assign each $\sigma_j^{(0)}$ to a real value which is then weighted by the dendritic input strength of each anterior neuron. Moreover an input neuron only exists on the 0th layer. In the case of each hidden layer we adopt the model described for the standard neuron as aforementioned where our sigmoid activation function $g = \tanh(\text{net})$ is the hyperbolic tangent.Finally, the output layer usually have a linear sigmoid activation as to achieve output scaling beyond $[1,-1]$ in the previous layers. Once again the output layer can only exist on the layer L.

1.4 Error Backpropagation

With the functional organization of the network complete, we now need to develop the notion of learning. For the purposes of this paper we will describe a gradient descent method for learning called error-backpropagation. In the mathematical model we find conveniently that the degrees of freedom are then the dendritic weights between any two neurons. Thus these weights must be optimized against some desired output. This leads to the following multi-dimensional error function.

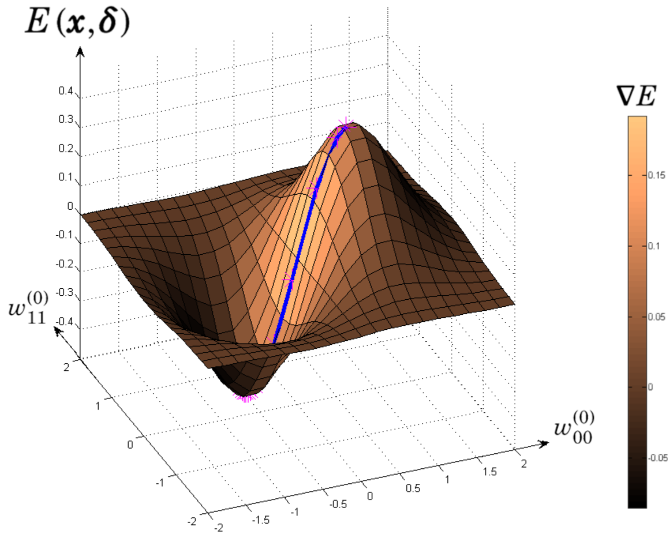

Figure 3: Gradient descent on $E(\boldsymbol{x},\boldsymbol{\delta})$ with descent path shown in blue.

Definition 1.4. *We call E the error function of a neural network $\mathscr{N}$, if for an input vector $\boldsymbol{x}$*

$$E\left(w_{00}^{(0)}, w_{01}^{(0)}, \ldots, w_{ij}^{(L)}\right) = \frac{1}{2} \sum_{i \in Z^{(L)}} \left(\sigma_i^{(L)} - \delta_i\right)^2$$

where $\boldsymbol{\delta}$ is some desired output vector corresponding to $\boldsymbol{x}$.

Then the goal is to optimize this error function such that a reasonable local minimum is found. We then choose to modify each weight in the direction of greatest decrease for the error function.

Definition 1.5. *We call ∇E the gradient of E if*

$$\nabla E = \left(\frac{\partial E}{\partial w_{00}^{(0)}}, \frac{\partial E}{\partial w_{01}^{(0)}}, \ldots, \frac{\partial E}{\partial w_{ij}^{(L)}}\right)$$

for all weights, $w_{ij}^{(l)}$, in feed-forward ANN $\mathscr{N}$.

Conveniently the gradient of a function describes a vector whose direction is the greatest increase of a function. Thus to optimize our weights so that the lowest error is achieved, we update the weights as follows: $\boldsymbol{w}(t+1) = \boldsymbol{w}(t) - \alpha\nabla E$ where alpha is some learning rate, a process which is depicted in Figure 3 [7].

The calculation of each $\frac{\partial E}{\partial w_{ij}^{(L)}}$ is non-trivial given that each weight influences the error function in multiple ways. To find the contribution of a single weight, recall that every single neuron is connected to all neurons in the anterior and posterior layers. So, a single weight will influence not only its posterior neuron's sigmoidal output but also that of every neuron for any path from the posterior neuron to the set of output neurons. Thus, the multiplicity of contribution via different neural routes follows directly from the multidimensional chain-rule. Recall the differential operator $D_g f = \frac{\partial f}{\partial x}$,

$$\begin{aligned}
\frac{\partial E}{\partial w_{ij}^{(l)}} = & D_{\sigma_0^{(L)}} \cdot D_{\text{net}}\sigma_0^{(L)} \cdot D_{\sigma_0^{(L-1)}}\text{net} \cdot \, \cdots \, \cdot D_{\text{net}}\sigma_j^{(l+1)} \cdot D_{w_{ij}^{(l)}}\text{net} \\
& + D_{\sigma_1^{(L)}} \cdot D_{\text{net}}\sigma_1^{(L)} \cdot D_{\sigma_0^{(L-1)}}\text{net} \cdot \, \cdots \, \cdot D_{\text{net}}\sigma_j^{(l+1)} \cdot D_{w_{ij}^{(l)}}\text{net} \\
& \vdots \\
& + D_{\sigma_n^{(L)}} \cdot D_{\text{net}}\sigma_n^{(L)} \cdot D_{\sigma_0^{(L-1)}}\text{net} \cdot \, \cdots \, \cdot D_{\text{net}}\sigma_j^{(l+1)} \cdot D_{w_{ij}^{(l)}}\text{net} \\
& + D_{\sigma_0^{(L)}} \cdot D_{\text{net}}\sigma_0^{(L)} \cdot D_{\sigma_1^{(L-1)}}\text{net} \cdot \, \cdots \, \cdot D_{\text{net}}\sigma_j^{(l+1)} \cdot D_{w_{ij}^{(l)}}\text{net} \\
& \vdots \\
& + D_{\sigma_n^{(L)}} \cdot D_{\text{net}}\sigma_n^{(L)} \cdot D_{\sigma_1^{(L-1)}}\text{net} \cdot \, \cdots \, \cdot D_{\text{net}}\sigma_j^{(l+1)} \cdot D_{w_{ij}^{(l)}}\text{net} \\
& \vdots \\
& + D_{\sigma_n^{(L)}} \cdot D_{\text{net}}\sigma_n^{(L)} \cdot D_{\sigma_m^{(L-1)}}\text{net} \cdot \, \cdots \, \cdot D_{\text{net}}\sigma_j^{(l+1)} \cdot D_{w_{ij}^{(l)}}\text{net} \\
& = \sum_{a_1}^{Z^{(L)}} \sum_{a_2}^{Z^{(L-1)}} \cdots \sum_{a_m}^{Z^{(l+2)}} \frac{\partial E}{\partial \sigma_{a_1}^{(L)}} \frac{\partial \sigma_{a_1}^{(L)}}{\partial \text{net}} \frac{\partial \text{net}}{\partial \sigma_{a_2}^{(L-1)}} \cdots \frac{\partial \sigma_{a_m}^{(l+2)}}{\partial \text{net}} \frac{\partial \text{net}}{\partial \sigma_j^{(l+1)}} \frac{\partial \sigma_j^{(l+1)}}{\partial \text{net}} \frac{\partial \text{net}}{\partial w_{ij}^{(l)}}
\end{aligned}$$

This construction is of serious mathematical interest as feed-forward neural networks have been shown to be universal approximators; that is, they can approximate any $f : A \to B$, where $A, B \in \mathbb{R}^m$ are vector spaces. In this paper we will generalize the notion of the universal approximation for arbitrary vector space mappings to arbitrary approximation of any$f : L^1(\mathbb{R}^n) \to C^\infty(\mathbb{R}^n)$ by examining the structure of feed-forward ANNs as the number of nodes for each layer becomes uncountably bounded in$\mathbb{R}^n$. Such a generalization requires that a continuum of neural components be made, and that a continuous weight tensor or hypersurface must exist in order to maintain the topological connectedness as prescribed by the discrete model.

2 Functional Neural Networks

Although neural networks have proven an extremely effective mechanism of machine learning [1], theoretically they remain a black-box model. In answer to this problem Neal examined the notion of infinite hidden nodes with a network proving that such a construction becomes a Gaussian kernel **neal** Then, Roux and Bengio described a model for affine neural networks with continuous hidden layers in alignment with Neal's reasearch. These authors showed effectively the viability of a "continuous" neural network, but left many similar constructions unexplored. It is the subject of this paper to generalize the construction of a feed-forward ANN which maps uncountably infinite vector spaces, and then to demonstrate the practical implementations of algorithms such as error backpropagation in this generalized form.

2.1 The Core Idea

Suppose that we wish to map one functional space to another with a neural network. Consider the standard model of an ANN as the number of nueral nodes for every layer becomes uncountable. The index for each node then becomes real-balue, along with the weight and input vectors . Let us denote $\xi : \mathbb{R}^n \to \mathbb{R}$ as some arbitrary input function for the neural network. Likewise consider a real-valued weight function, $w^{(l)} : \mathbb{R}^{2n} \to \mathbb{R}$, for a later l which is composed of two indexing variables $i, j \in \mathbb{R}^n$. Finally as the number of neural nodes becomes uncountable we define a real-valued bound for any given layer $R^{(l)} \supset Z^{(l)}$. As the indices become real valued, examination of the sigmoidal sum seen in definition 1.1 leads us to the following derivation

$$\begin{aligned}\sigma^{(1)}(j) &= \lim_{\Delta i \to 0} g\left(\sum_{i \in R^{(0)}} \xi(i) w^{(0)}(i,j)\ \Delta i + \beta\right) \\ &= g\left(\int_{R^{(0)}} \xi(i) w^{(0)}(i,j)\ di + \beta\right)\end{aligned}$$

Repeating the process inductively for all layers in a neural network, leads to a recurrance relation for this construction.

Definition 2.1. *We call $\mathscr{F}$ a functional neural network if,*

$$\begin{aligned}\mathscr{F}: \sigma^{(l+1)}(j) &= g\left(\int_{R^{(l)}} \sigma^{(l)}(i) w^{(l)}(i,j)\, di + \beta\right)\\ \sigma^{(0)}(j) &= \xi(j)\end{aligned}$$

To demonstrate the accuracy of this generalization, let us propose the following lemma which is near that of Roux and Bengio.

Theorem 1. *Suppose $\mathscr{F}$ is a functional neural network with a set of piecewise constant weight functions $W = \{w^{(0)}, w^{(1)}, \dots, w^{(L-1)}\}$ each with constituent pieces of length one. Then given the input function $\xi(i) = x_n, n = \max\{m \in \mathbb{Z} \mid m \le i\}$ for some input vector $\boldsymbol{x}$, $\mathscr{F}$ is discretized; that is assuming $i,j \in \mathbb{R}$ and $Z^{(l)} = R^{(l)} \cap \mathbb{Z}$ then for $0 \le l \le L-1$ we have that*

$$\mathscr{F}(\xi) \equiv \mathscr{N}(\boldsymbol{x})$$

Proof. Let $P(l)$ be the proposition that $\sigma^{(l+1)}(j)$ becomes discretized when $w^{(l)}(i,j)$ and $\sigma^{(l)}(i,j)$ are piecewise constant with constituent functions of length one. Moreover let $w^{(l)}_{ij}$ denote the value of $w^{(l)}(i,j)$ for $(i,j) = \max\left\{(x,y) \in \mathbb{Z}^2 \mid x \le i \wedge y \le j\right\}$. Then by induction we show $P(l), 0 \le l \le L-1$.

Basis Step. Recall that $\sigma^{(0)}(j) = \xi(j)$. Then one would suppose

$$\sigma^{(1)}(j) = g\left(\int_{R^{(0)}} \xi(i) w^{(0)}(i,j)\, di + \beta\right)$$

but because the weight function and the input function are piecewise constant and not guarenteed to be continuous for $R^{(0)}$, we must take the improper integral along each constituent piece of length one. Supposing that each summation in the following is taken over $k \in Z^{(0)}$,

$$\begin{aligned}\sigma^{(1)}(j) &= g\left(\sum_k \lim_{b\to k}\int_{k-1}^{b} \xi(i)w^{(0)}(i,j)\ di + \beta\right)\\ &= g\left(\sum_k w^{(0)}_{kj}\lim_{b\to k}\int_{k-1}^{b} \xi(i)\ di + \beta\right)\\ &= g\left(\sum_k w^{(0)}_{kj}x_b + \beta\right)\end{aligned}$$

Inductive Step. Now assume that for some l we have that

$$\sigma^{(l+1)}(j) = g\left(\sum_{i\in Z^{(l)}} w^{(l)}_{ij}\sigma^{(l)}_i + \beta\right)$$

We now show that by the inductive hypothesis if $P(0) \wedge P(l) \to P(l+1)$, then $P(k)\ \forall k$. Consider the next neural layer defined as

$$\sigma^{(l+2)}(j) = g\left(\int_{R^{(l+1)}} \sigma^{(l+1)}(i)w(i,j)\ di + \beta\right)$$

Then because $w(i,j)$ and $\sigma^{(l+1)}$ are piecewise constant by defention and not necisarrily continous for $R^{(l+1)}$ we must again take the improeper riemann integral over the constituent pieces. Consider $k \in Z^{(l+1)}$

$$\begin{aligned}\sigma^{(l+2)}(j) &= g\left(\sum_k \lim_{b\to k}\int_{k-1}^{b} \sigma^{(l+1)}(i)w(i,j)\ di + \beta\right)\\ &= g\left(\sum_k w^{(l+1)}_{kj}\sigma^{(l+1)}_k \lim_{b\to k}\int_{k-1}^{b}\ di + \beta\right)\\ &= g\left(\sum_k w^{(l+1)}_{kj}\sigma^{(l+1)}_k + \beta\right)\end{aligned}$$

Therefore by the inductive hypothesis the proof follows and $\mathscr{F}(\xi) \equiv \mathscr{N}(\boldsymbol{x})$ for piecewise constant input and weight functions.

□

From the logic of the preceding proof we can estagblish that, the input function need only be properly Lebesgue integrable over $R^{(0)}$. Moreover, we come to an extremely important intutiton; the weight matrix for a given layer l can be thought of as a piecewise constant weifght surface, and the linear combination of weights can be thought of as a piecewise integral tramsformation along a given j on the weight surface. However, functional neural networks allow for infinite weight surfaces and therefore can represent the entire class of integral transforms. With this in mind, it is now possible to consider functional neural networks as universal approximators.

2.2 Universal Approximation of Bounded Linear Operators

In the case of discretized neural networks, George Cybenko and Kolmogorov have shown that with sufficient weights and connections, a feed-forward neural network is a universal approximator of arbitrary $f : \mathbb{R}^n \to \mathbb{R}^m$ [8]; that is, constructs of the form $\mathscr{N}(\boldsymbol{x})$ are dense in $C(I^n, \mathbb{R})$ where I^n is the unit hypercube $[0,1]^n$. Cybenko proved this remarkable result by utilizing the Riez Representation Theorem for Hilbert spaces and the Hahn-Banach theorem. He showed by contradiction that there exists no bounded linear functional $h(x)$ in the form of $\mathscr{N}(\boldsymbol{x})$ such that $\int_{I_n} h(x)\, d\mu(x) = 0$.

Although this theorem has lead to a remarkable interest in ANNs by legitimizing their effectiveness, it still has shed no light on the internal workings of ANNs. Fortunately using the intuitions presented in Theorem 1, it would be adventageous to examine the generalization of Cybenko's theorem to the larger class of functional neural networks. However, there is no clear way with which to do this; discretized neural networks map vector spaces and therefore approximate continuous functions, whereas functional neural networks are defined as arbtrary mappings between Hilbert spaces (more specifically the set of L^2 integrable functions). Moreover letting n approach infinity in Cybenko's proof fails to hold in that there is not an obvious topology for $C(C(\mathbb{R}), C(\mathbb{R}))$. Therefore we must develop an approximation theorem for $\mathscr{F} : C(X) \to C(Y)$ over the set of linear operators.

First, however, let us develop the notion of $\mathscr{F}$ as a universal approximator of arbitrary functions. By Theorem 1, we have that $\mathscr{F} \equiv \mathscr{N}$,and in that sense for piecewise constant $w(i,j)$ and $\xi(i)$ functional neural networks approxmiate any arbitrary mapping from $\mathbb{R}^n \to \mathbb{R}$ where $n = |Z^{(0)}|, m = |Z^{(L-1)}|$ by Cybenko's theorem. However, when considering the fully continuous case the following corollary arises from the Stone-Weierstrass theorem.

Corollary 1. *Suppose $\mathscr{F}$ is a multi-layer functional ANN. Then for some real-valued continous function $f : \mathbb{R} \to \mathbb{R}$, there exists a set of weights W such that $\forall \epsilon > 0$, $\|\mathscr{F}(\xi) - f(\xi)\|_\infty < \epsilon$*

Proof. In this proof we will ommit the inductive step as it is elementary and employs the same logic as the basis step. Consider the first neural layer

$$\sigma^{(1)}(j) = g\left(\int_{R^{(0)}} \xi(i)w^{(0)}(i,j)\, di\right)$$

because we take $w^{(0)}$ to be some approximating polynomial by the Stone-Weierstrass theorem, let $w^{(0)} = \left[(g^{-1})' \circ (h(\Xi, j))\right] h'(\Xi, j)$ approximately, where Ξ is the primative of ξ. Supposing that h is some polynomial satisfying $h(\Xi, j)\Big|_{R^{(0)}} = f(j)$, then by the chain rule of integration

$$\begin{aligned}\sigma^{(1)}(j) &= g\left(\int_{R^{(0)}} \xi(i)\left[(g^{-1})' \circ (h(\Xi, j))\right] h'(\Xi, j)\, di\right)\\ &= g\left(g^{-1}(h(\Xi, j))\right)\Bigg|_{R^{(0)}}\\ &= f(j)\end{aligned}$$

□

At this point it is important to note that the proof given above implies that ξ is disregarded through manipulation of $w^{(0)}$. Instead, h is a function of Ξ which is the primitive of ξ. If we were to not let $h(\Xi, j)\Big|_{R^{(0)}} = f(j)$, then we could claim that for arbitrary h we have proven any functional composition of ξ is possible; that is, in some light sense we have proven Cybenko's theorem in the general case by treatingthe weight set as some hypersurface. Moreover, the intutition follows that if we were to discretize the satisfying h and ξ (Theorem 1) then it is possible that a similar weight surface is developed for a trained $\mathscr{N}$. This result is remarkable as new light is shed on the black-box model of neural networks showing that approximation of h is made in the discrete sense.

Although we have shown through the corollary that approximation of arbitrary functional composition is possible, we have yet to consider values of w in the general sense. In other words, what can be said about the general approximation of bounded linear operators mapping $C(\mathbb{R}^n)$ to $C(\mathbb{R}^n)$ where C denotes the set of continuous (integrable) real valued functions. Evidently, the form of $\mathscr{F}$ resembles the general class of integral transforms, $\int f(x)\cdot E(x,k)\ dx$. Integral transforms are shown to approximate a multitude of operators through varying kernel functions. For example consider some $g(t)$ and the integral transform

$$(\mathscr{P}g)(s) = \int_0^\infty g(t)\delta'(s-t)\ dt$$

where δ is the Dirac-Delta function. Then we have that $(\mathscr{P}g)(s) = \frac{dg}{dt}\Big|_s$ by the proeprties of the Delta function. Similar approximations of linear operators can be made by varying the kernek function E. Thus there is considerable interest in determining the density of integral transforms and thereby functional neural networks in the set of bounded linear operators.

Further investigation into dense forms of bounded linear operators leads us to the Schwartz theorem of bounded linear operator representation by integral kernels. The theorem simply states that all linear operators can be represented in some light sense by integral transforms with arbitrary kernels. However, this theorem is too general for our purposes and we would like to show that in the specific case of some functional neural network $\mathscr{F}$ that for any given layer such that $l \neq 0$ any linear operator can be approximated with point-wise convergence from the Weierstrass polynomial approximation theory.

In order to do this we will return to the Riesz representation theorem that states the following [9].

Theorem 2. *Let $\phi : C(X) \to \mathbb{R}$ be any bounded linear form where X is a compact Hausdorff space and $C(X)$ is the Banach space of continuous functions over X. Then there exists a unique regular Borel measure μ on X such that*

$$\phi(f) = \int_X f(t)\; d\mu(t),\;\; f \in C(X),\;\; t \in X$$

and $\|\phi\| = |\mu|(X)$ where $|\mu|$ is the total variation of μ on X.

As opposed to generalizing Cybenko's theorem to Banach spaces ($\mathbb{R}^{\infty}$), we can actually manipulate the representation theorem to encapsulate bounded linear operators over locally compact Hausdorf spaces. Using the aforementioned logic the universal representation theorem for functional neural networks is now proposed.

Theorem 3. *Given a functional neural network $\mathscr{F}$ then some layer $l \in \mathscr{F}$, the let $K : C(R^{(l)}) \to C(R^{(l)})$ be a bounded linear operator. If we denote the operation of layer l on layer $l-1$ as $\sigma^{(l+1)} = g\left(\Sigma_{l+1}\sigma^{(l)}\right)$, then for every $\epsilon > 0$, there exists a weight polynomial $w^{(l)}(i,j)$ such that the supremum norm over $R^{(l)}$*

$$\left\| K\sigma^{(l)} - \Sigma_l \sigma^{(l)} \right\|_{\infty} < \epsilon$$

Proof. Let $\zeta_t : C(R^{(l)}) \to R^{(l)}$ be a linear form which evaluates its arguments at $t \in R^{(l)}$; that is, $\zeta_t(f) = f(t)$. Then because $\zeta_t$is bounded on its domain, $\zeta_t \circ K = K^{\star}\zeta_t$ is a bounded linear functional. Then from the Riesz Representation Theorem (Theorem 1) we have that there is a unique regular borel measure μ_t on $R^{(l)}$ such that

$$\left(K\sigma^{(l)}\right)(t) = K^{\star}\zeta_t\left(\sigma^{(l)}\right) = \int_{R^{(l)}} \sigma^{(l)}(s)\; d\mu_t(s),$$

$$\|\mu_t\| = \|K^{\star}\zeta_t\|$$

Then if there exists a regular borel measure μ such that μ_t is significantly smaller that μ

for all t, then we have that $d\mu_t(s) = K_t(s)d\mu(s)$ under the assumption that K_t is L^1 integrable over $R^{(l)}$ with the measure μ. Thus it follows that

$$K\left[\sigma^{(l)}\right](t) = \int_{R^{(l)}} \sigma^{(l)}(s)K_t(s)\ d\mu(s) = \int_{R^{(l)}} \sigma^{(l)}(s)K(t,s)\ d\mu(s).$$

Therefore, for any bounded linear operator $K : C(X) \to C(X)$ there exists a unique $K(t,s)$ such that $K[f] = \int_X f(s)K(t,s)d\mu(s)$. Now we show that the operation of Σ_l can approximate any such operator. Because K is of the form of Σ_l where the only difference is the weighting function, we assert the following claim.

Let G be defined as a linear functional applied to a gaussian heat kernel whose application with a function $f : \mathbb{R} \to \mathbb{R}$ yields the following definition,

$$G[f](x) = \frac{1}{b\sqrt{\pi}} \int_{\mathbb{R}} f(u) \mathrm{e}^{\left(\frac{u-x}{b}\right)}\ du.$$

Then it follows that by the Weierstrass approximation theorem that for all $\epsilon > 0$, the supremum norm $\|f - G[f]\|_\infty < \epsilon$. Then because G is a polynomial, f must be a limit of polynomials. So now consider the operation of $K\left[\sigma^{(l)}\right](t)$ with kernel $K(t,s)$. By the weierstrass approximation theorem $K(t,s)$ must be a limit of polynomials and therefore we let $w^{(l)}(i,j)$ assume that limit. That is,

$$\begin{aligned} \lim_{b\to 0} \left\| K\left[\sigma^{(l)}\right] - \int_{R^{(l)}} \sigma^{(l)}(s)\ G[k]\ d\mu(s) \right\|_\infty &= \\ \left\| K\left[\sigma^{(l)}\right] - \Sigma_{l+1}\left[\sigma^{(l)}\right] \right\|_\infty &< \epsilon \end{aligned}$$

Thus we have that as a limit of polynomials the operation of any arbitrary layer of a functional neural network $\mathscr{F}$ approaches any arbitrary linear bounded operator $K : C(R^{(l)}) \to C(R^{(l)})$; that is, functionals of the form Σ_{l+1} are dense in the set of all bounded continuous linear operators. □

References

[1] C. Burch, “A survey of machine learning”, A survey for the Pennsylvania Governor’s School for the Sciences, 2001.

[2] J. S. Griffith, “Mathematical neurobiology - an introduction to the mathematics of the nervous system”, *Academic Press*, 1971.

[3] P. Lisboa, “Introduction”, in *Neural Networks: Current Applications*, P. Lisboa, Ed., Chapman & Hall, 1992, ch. 1.

[4] S. G. Sam Gabrielsson, “The use of self-organizing maps in recommender systems: a survey of the recommender systems field and a presentation of a state of the art highly interactive visual movie recommender system”, Master’s thesis, Uppsala University Devision of Computer Systems, Aug. 2006.

[5] W. S. McCulloch and W. Pitts, “A logical calculus of the ideas immanent in nervous activity”, *The bulletin of mathematical biophysics*, vol. 5, no. 4, pp. 115–133, 1943.

[6] J. H. Kaas, “The organization of visual cortex in primates: problems, conclusions, and the use of comparative studies in understanding the human brain”, in *Functional Organisation of the Human Visual Cortex*, ser. Wenner-Gren International Series, D. O. Balaz Gulyas and P. E. Roland, Eds., vol. 61, Pergamon Press, 1993, ch. 1.

[7] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, “Learning representations by back-propagating errors”, *Nature: Cognitive modeling*, vol. 323, pp. 533–536, Oct. 1988.

[8] G. Cybenko, “Approximation by superpositions of a sigmoidal function”, *Mathematics of Control, Signals, and Systems*, vol. 2, pp. 303–3314, 1989.

[9] D. G. Hartig, “The riesz representation theorem revisited”, English, *The American Mathematical Monthly*, vol. 90, no. 4, 1983, ISSN: 00029890. [Online]. Available: `http://www.jstor.org/stable/2975760`.