Using deep neural networks to solve discrete-continuous dynamic choice models in Overlapping Generations setting

Alexander Kwon *

February 15, 2024

Abstract

In this paper, I approximate the solution of a discrete-continuous dynamic choice model using deep neural networks. The model of interest is an overlapping generation (OLG) model in which the individual needs to make two decisions: how much to consume, and when to retire. Overall, the deep neural networks designed in this paper approximate the analytic solution of consumption and the retirement probability relatively well. However, some points are relatively far from the analytic solution at the kink points and the discontinuity points, especially for the individual of age 1.

*Program in Economics, Graduate Center, CUNY

1 Introduction

In this paper, I approximate the solution of a discrete-continuous dynamic choice model using deep neural networks. The model of interest is an overlapping generation (OLG) model in which individuals need to make two decisions: how much to consume and when to retire. The consumption choice is a continuous variable, while the retirement choice is a discrete variable. According to Iskhakov et al. (2017), solving discrete-continuous dynamic choice models is harder than models with only discrete or continuous choice. The discrete choice introduces kinks and non-concave regions in the value function. Thus, the first-order conditions for solving the dynamic problem might be only necessary conditions rather than sufficient conditions for global optimization. This implies that the solutions to the first-order necessary conditions might correspond to local maximum rather than the global maximum.

Although solving the model can be challenging, the discrete-continuous dynamic choice model can be helpful and important in modeling real-life problems. For example, it can be used to model retirement and Social Security claiming decisions in a dynamic setting; See (French and Jones, 2011; Rust and Phelan, 1997) for instance. There are numerous other potential applications, such as whether to join a union, enter the labor force, and more.

The main question of this paper is whether the deep neural network can approximate the optimal policy functions when they have discontinuities at finite points. Kidger and Lyons (2020) show that a deep narrow network with ReLU activation function can approximate a L^p function, which includes functions with kinks and discontinuities. However, Llanas et al. (2008) argue that the approximation of a discontinuous function with a neural network is more difficult than a continuous function. They illustrate that the approximation of a piecewise continuous function with a smooth activation function is unsatisfactory even with many hidden nodes and training iterations.

I approximate the solutions of the deterministic retirement-consumption choice model presented in Iskhakov et al. (2017) by blending the algorithm suggested by Azinovic et al. (2022) and Maliar and Maliar (2022). I also incorporate discontinuity learning suggested by Della Santa and Pieraccini (2023). I evaluate the deep neural network-based algorithm's performance by comparing its approximation to the analytic solution.

Overall, the deep neural network used in this paper approximates the analytic solution of consumption and retirement probability relatively well. However, there are some discrepancies between the analytic solution and approximation at the kink points and the discontinuity points, especially for the individual of age 1.

The paper is organized as follows. In Section 2, a brief literature review is given. In Section 3, I introduce the economic model. In Section 4, I explain the numerical method. In Section 5, I present the numerical results. In Section 6, I conclude.

2 Literature Review

This paper is closely related to three previous literature: Iskhakov et al. (2017), Azinovic et al. (2022), and Maliar and Maliar (2022). The difference between this paper from the previous papers is that it solves the discrete-continuous dynamic choice model in the OLG setting with the help of deep neural networks. I choose an economic model that has no uncertainty and has small heterogeneity, which implies that the state space is small. Even though Azinovic et al. (2022) and Maliar and Maliar (2022) show that the method using a deep neural network has an advantage over previous computational methods when the state space is large and irregular, this paper focuses on solving a simple model with the analytic solution. However, the policy functions that give the analytic solution contain discontinuity points and kinks.

Iskhakov et al. (2017) propose an endogenous grid method for solving OLG models with discrete-continuous choice. This method extends the endogenous grid method presented by Carroll (2006) which solves discrete choice problems in the model. The basic idea behind the endogenous grid method is to eliminate sub-optimal endogenous grids caused by the discrete choice. While Iskhakov et al. (2017) successfully approximate the solution with or without a taste shock, their method does not use deep neural networks.

Azinovic et al. (2022) suggest that deep neural networks can help solve complicated problems in economics. They suggest an algorithm called "Deep Equilibrium Nets" (henceforth, DEN). DEN successfully solves an OLG model that has a large state space and kinks in the value function with deep neural networks. Although they solve three different models, they do not consider a model with discrete choice.

Maliar and Maliar (2022) solve the Krusell and Smith (1998) heterogeneous agent model that has divisible or indivisible labor choices using deep neural networks. The idea behind solving the model with indivisible labor is to formulate a classification problem. However, the models solved in Maliar and Maliar (2022) are not overlapping generation models.

The paper joins the strand of literature that uses neural networks to approximate a function. For example, Selmic and Lewis (2002) incorporate a basis function, $\varphi_k(x)$, as activation functions to successfully approximate a discontinuous function. However, the exact discontinuity point needs to be known. Llanas et al. (2008) show that a step function

needed for function approximation can be approximated with a single hidden-layer feed-forward neural network. Della Santa and Pieraccini (2023) incorporate a discontinuous layer to both detect discontinuity interfaces and approximate the discontinuous function.

3 The economic model

The model of interest is from Iskhakov et al. (2017). This model has an analytic solution.

Consider a model in which an individual needs to choose consumption at time t (c_t), and decide to work or retire (d_t). $d_t = 0$ means that the individual decides to retire at time t while $d_t = 1$ means the individual decides to work. If the individual chooses to work during this period, the income for the next period increases by a fixed amount. This amount is not received if the individual chooses to retire.

The individual who lives until age T optimizes lifetime utility,

$$\max_{\{c_t, d_t\}_{t=1}^{T}} \sum_{t=1}^{T} \beta^t (\log c_t - \delta d_t). \tag{1}$$

β is the discount factor and $\delta > 0$ is the dis-utility of working. The constraint of the model is $c_t \leq M_t$ in which $M_t = R(M_{t-1} - c_{t-1}) + y d_{t-1}$. R is the non-stochastic gross interest rate and y is the fixed labor income for workers. We can rewrite the constraint by introducing savings, $a_t = M_t - c_t$, as $c_t + a_t = M_t = R a_{t-1} + y d_{t-1}$. There are two non-negativity constraints, $c_t \geq 0$ and $a_t \geq 0$ for all t. If we introduce capital $k_t = a_{t-1}$, the state for an individual at period t is decided by $\{k_t, d_{t-1}\}$ or $M_t = R k_t + y d_{t-1}$. By looking at the state, individuals decide how much to consume and whether to retire at the beginning of each period. After the decisions are made, the payment is given at the end of the period. Assume that those who decide to retire cannot come back and work again.

As shown in Iskhakov et al. (2017), the optimal consumption for the stated problem is not continuous with respect to income level. It has kinks and discontinuities. To illustrate this point, let $V_t(M)$ be the value function of a worker at period t with income M. Let $v_t(M, d_t)$ be the value function depending on the individual choice to work which is expressed with d_t. The choice to work at period t is decided by the Bellman equation,

$$V_t(M) = \max\{v_t(M, 0), v_t(M, 1)\}. \tag{2}$$

Equation 2 implies that the individual chooses to retire if $v_t(M, 1) > v_t(M, 0)$ and, chooses

to work otherwise. Even though $v_t(M, 0)$ and $v_t(M, 1)$ are concave functions of M, the value function $V_t(M)$ will generally not be concave (Iskhakov et al., 2017; Clausen and Strub, 2016). Also, $V_t(M)$ will have a kink point at $v_t(M, 0) = v_t(M, 1)$. Let this point be $\bar{M}_t$. Then, the optimal consumption rule will have a discontinuity at $\bar{M}_t$. If the income level is just below $\bar{M}_t$, the individual can consume more today since the individual will work at period t and will receive labor income y at period $t + 1$. However, if the income level is just above $\bar{M}_t$, the individual will consume less and save more since the individual will retire today and not receive labor income y in the next period. The kink point in $V_t(M)$ propagates backward to period $t - 1$ since $V_{t-1}(M)$ depends on $V_t(M)$. Thus, the kink at period t results in kinks and discontinuities in the consumption function of the earlier periods.

Iskhakov *et al.* (2017) show that under the condition,

$$\beta R \leq 1 \; and \; \delta < (1 + \beta) \log(1 + \beta),$$

problem (1) can be solved with backward induction. The optimal consumption rule of the individual at $T - \tau$ for a given income level M is reported in equation 3. The first condition, $y/R\beta$, is the income threshold where the individual is under liquidity constraints at period $T - \tau$. If income is below $y/R\beta$, the individual consumes all the income due to liquidity constraints. $\bar{M}^{l_j}_{T-\tau}$ is the threshold income level for the individual at period $T - \tau$ that makes the individual not bounded by the liquidity constraint today but will make the individual be bounded j periods later. $\bar{M}^{r_j}_{T-\tau}$ stands for the threshold income level where the individual is indifferent to retiring and working at period $T - \tau + j$. If the income is above $\bar{M}^{r_j}_{T-\tau}$, it is optimal for the individual to retire at period $T - \tau + j$. $\bar{M}_{T-\tau}$ is the income level that decides whether it is optimal to retire at period $T - \tau$. If the income level is above $\bar{M}_{T-\tau}$, it is optimal to retire at period $T - \tau$. Detailed derivation can be found in Iskhakov *et al.* (2017).

$$c_{T-\tau} = \begin{cases} M & \text{if } M \leq y/R\beta \\ [M + y/R]/(1+\beta) & \text{if} y/RB \leq M \leq \bar{M}^{l_1}_{T-\tau} \\ \dots & \dots \\ [M + y\left(\sum_{i=1}^{\tau} R^{-i}\right)](\sum_{i=0}^{\tau} \beta^i)^{-1} & \text{if} \bar{M}^{l_{\tau-1}}_{T-\tau} \leq M \leq \bar{M}^{r_{\tau-1}}_{T-\tau} \\ [M + y\left(\sum_{i=1}^{\tau-1} R^{-i}\right)](\sum_{i=0}^{\tau} \beta^i)^{-1} & \text{if} \bar{M}^{r_{\tau-1}}_{T-\tau} \leq M \leq \bar{M}^{r_{\tau-2}}_{T-\tau} \\ \dots & \dots \\ [M + y/R](\sum_{i=0}^{\tau} \beta^i)^{-1} & \text{if} \bar{M}^{r_1}_{T-\tau} \leq M \leq \bar{M}_{T-\tau} \\ [M](\sum_{i=0}^{\tau} \beta^i)^{-1} & \text{if} M \geq \bar{M}_{T-\tau} \end{cases} \tag{3}$$

Let λ_t be the Lagrangian multiplier for the budget constraint $c_t + a_t = Ra_{t-1} + yd_{t-1}$. Let μ^c_t and μ^a_t be the Lagrangian multiplier for the non-negative constraint of consumption and savings, respectively. The necessary Karush-Kuhn-Tucker (KKT) conditions for the problem are as follows.

$$\beta R \lambda_{t+1} + \mu^a_t = \lambda_t \tag{4}$$

$$\frac{1}{c_t} + \mu^c_t = \lambda_t \tag{5}$$

$$\text{Complementary Slackness condition for consumption : } \mu^c_t c_t = 0 \; \forall t \tag{6}$$

$$\text{Non-negative consumption : } c_t \geq 0 \; \forall t \tag{7}$$

$$\text{Lagrangian Multiplier for consumption : } \mu^c_t \geq 0 \; \forall t \tag{8}$$

$$\text{Complementary Slackness condition for savings : } \mu^a_t a_t = 0 \; \forall \; t \tag{9}$$

$$\text{Non-negative savings : } a_t \geq 0 \; \forall t \tag{10}$$

$$\text{Lagrangian Multiplier for savings : } \mu^a_t \geq 0 \; \forall t \tag{11}$$

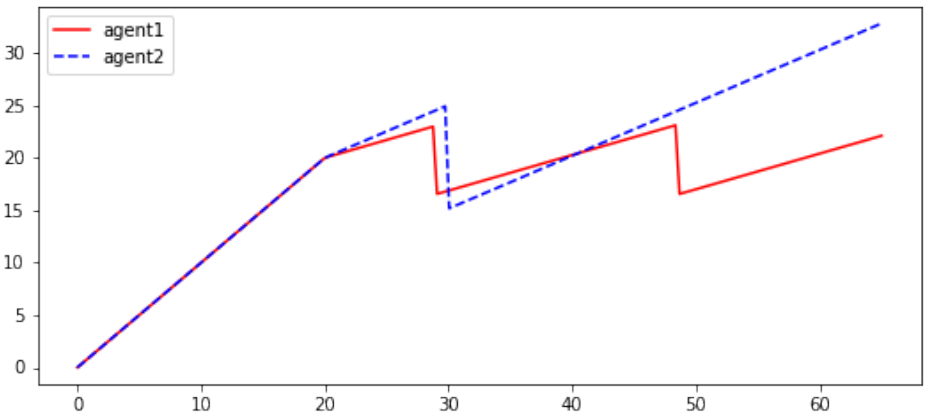
By assuming that $c_t > 0$ for all t, we can derive the Euler equation as in Equation 12. Since zero consumption will not be an optimal solution for positive income, $M_t > 0$, I call Equation 12 as the Euler equation for the problem.

$$Euler\ equation: \frac{1}{c_t} - \frac{\beta R}{c_{t+1}(M_t)} + \mu_t^a = 0\ \forall t \tag{12}$$

$$\text{where } M_t = R(M_{t-1} - c_{t-1}) + yd_{t-1} \tag{13}$$

For simplicity, let's assume that the agent lives for three periods, $T = 3$. The discount factor, β, is 0.95 and the dis-utility of working, δ, is 1. The non-stochastic interest rate is 1.02 ($R = 1.02$), and the fixed labor income y is 20. The optimal consumption for individuals aged 0 and 1 is presented in Figure 1. Let agent 1 represent the individual of age 1, which is the starting age, and agent 2 represent the individual of age 2. In figure 1, the optimal consumption for agents 1 and 2 have kinks and discontinuity points. The kinks happen at the points where individuals are subject to liquidity constraints before, but not after the point. The discontinuity points are the points where the individuals are indifferent to retiring and working now or some period later.

Figure 1: Optimal consumption for agent 1 and agent 2



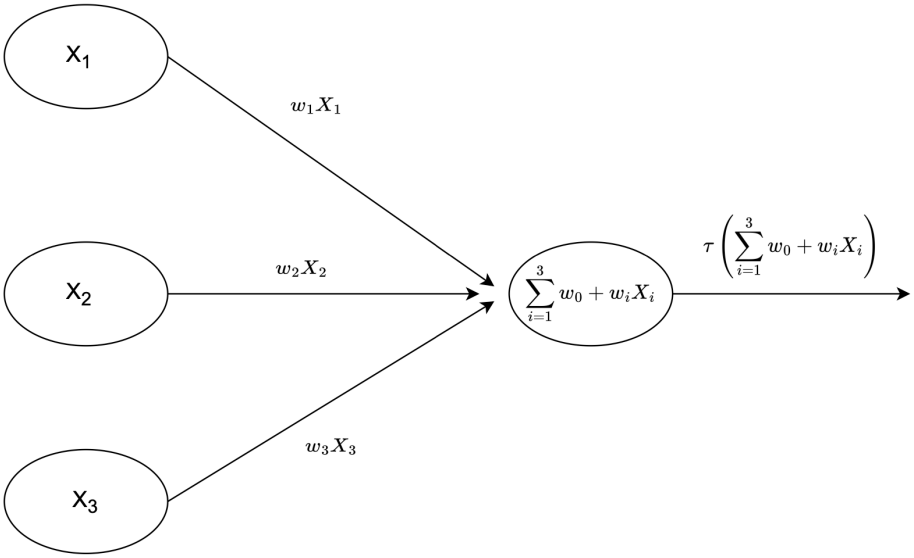
4 Numerical method with deep neural networks

In this section, I introduce the basic neural network setup and apply it to the problem of interest. The methodology implemented in this paper is an extension of Maliar and Maliar (2022) and Azinovic et al. (2022).

4.1 Neural Networks

Figure 2 represents one artificial neuron that receives outputs from 3 previous nodes/neurons, namely inputs for that node. The inputs are combined as a weighted sum added with a bias term. In figure 2, the weights for each input are denoted with w_k for $k = 1, 2, 3$ and the bias term is denoted as w_0. The weighted sum is passed forward to the next layer's nodes after transformation with an activation function denoted as τ. Figure 3 represents a simple structure of neural networks. Each node in figure 3, which is represented as a circle, has a similar structure as 2.

Figure 2: Example of a single neuron



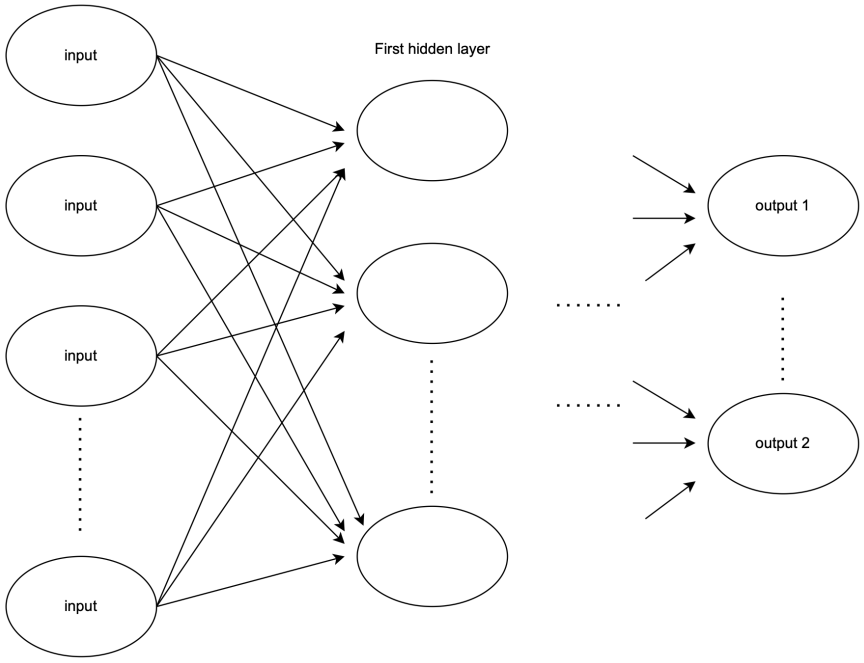
To represent the neural network with notations, I borrow the notations from Azinovic et al. (2022). Let K be the number of layers of the network, m_i be the number of neurons inside layer i, and τ_i be the activation function of layer i. Let $\rho = \{W_1, W_2, \ldots, W_K, b_1, b_2, \ldots, b_K\}$ be the trainable parameters that the neural network will update. Given hyper-parameters $\{K, \{m_i\}_{i=1}^K, \{\tau_i\}_{i=1}^K\}$, and trainable parameters $\rho = \{W_1, W_2, \ldots, W_K, b_1, b_2, \ldots, b_K\}$, a neural network $\mathcal{N}_\rho$ is a map,

$$x \to \mathcal{N}_\rho = \tau_K \left(W_k \ldots \tau_2(W_2\tau_1(W_1x + b_1) + b_2) \cdots + b_K\right) \tag{14}$$

The object of the neural network is to minimize the loss/cost function, $l(p)$. To achieve this goal, the trainable weights are updated with gradient descent,

$$\rho_k^{new} = \rho_k^{old} - \alpha^{learn} \frac{\partial l(p)}{\partial \rho_k^{old}} \; \forall k \in \{1, \ldots, K\} \tag{15}$$

Figure 3: Example of the structure of the neural network



where α^{learn} is the learning rate and $\frac{\partial l(p)}{\partial \rho_K^{old}}$ is the partial derivative of the loss function with respect to ρ_K^{old}. There are multiple variants of gradient descent. In this paper, I use the Adam optimizer with mini-batches. Mini-batches are random subsets of the training data. The gradient descent algorithm in equation 15 is applied to each mini-batches. Adam optimizer computes adaptive learning rates based on an efficient estimation method of first and second moments (Kingma and Ba, 2015).

4.2 Binary classification problem: logistic regression

The logistic regression formulation is used to solve the discrete retirement choice problem following Maliar and Maliar (2022). In this section, I briefly introduce the typical classification problem. Let $X = \{x_1, x_2, \ldots, x_l\}$ be the l independent variables and y_{label} be the binary dependent variable that we want to correctly predict with X. Typically y_{label} takes the value 0 or 1. The binary classification problem is to separate the X space into two groups where one group mostly has $y_{label} = 0$ and the other mostly has $y_{label} = 1$.

The logistic regression formulates the problem as,

$$\log \frac{p}{1-p} = X\theta \tag{16}$$

where p is the probability that a data point in X belongs to class 1 and θ are the coefficients. We can rewrite equation 16 as

$$p = \frac{1}{1 + exp(-X\theta)}. \tag{17}$$

We can estimate the logistic regression with the Maximum log-likelihood.

$$\max_{\theta} \; lnL(\theta) = \frac{1}{L}\sum_{i=1}^{L}[y^i_{label} ln(p(X^i;\theta)) + (1 - y^i_{label}) ln(1 - p(X^i;\theta)] \tag{18}$$

where $p(X^i;\theta)$ is given in equation (14).

The prediction in this paper is made as follows. Let $y_{prediction}$ be the class predicted by the neural network.

$$y_{prediction} = \begin{cases} 1 & \text{if } p(X^i;\theta) > 0.5 \\ 0 & \text{otherwise} \end{cases} \tag{19}$$

4.3 Discontinuity learning

As shown in Figure 1, the optimal consumption function has kinks and discontinuity at some points. I follow Della Santa and Pieraccini (2023) to make the neural network learn by itself where the discontinuity points are and approximate the discontinuous function simultaneously. Della Santa and Pieraccini (2023) introduce discontinuity in the neural network layer by adding a jump. Let $\mathcal{H}(\cdot)$ represent a Heaviside function

$$\mathcal{H}(x) = \begin{cases} 1 & \text{if } x \geq 0 \\ 0 & \text{otherwise.} \end{cases}$$

Then the discontinuous layer is passing the weighted sum of the previous layer to the next layer as follows:

$$\tau(W^T x + b) + \epsilon \odot \mathcal{H}(W^T x + b) \tag{20}$$

in which W are the weights and b is the bias term, and $\epsilon \in \mathbf{R}^d$ is the vector of trainable discontinuity jumps where d is the number of nodes in the next hidden layer, and $\odot$ is a Hadamard product. The intuition is that the degree of the jump and the discontinuity region should be learned by the second term in Equation 20.

I use a sigmoid activation function rather than a Heaviside function for the neural network in this paper. Let S be a sigmoid function, $S(x) = \frac{1}{1+\exp(-x)}$. Then the jump

function that I use is:

$$\mathcal{J}(x) = \begin{cases} 2 & \text{if } S(x) \geq 0 \\ 0 & \text{otherwise.} \end{cases}$$

Moreover, I set the derivative of the jump function as

$$\textit{derivative} = S \times (1 - S),$$

which is used in the back propagation. The intuition is to make the discontinuous layer similar to a classification problem using a sigmoid activation function and make the derivative of the jump function not zero almost surely.

4.4 Applying all the methods to solve the problem

The neural networks I use have 1 input layer, 5 hidden layers, and 1 output layer. I choose 5 hidden layers to ensure that the kinks and discontinuities are learned by the neural network.

Given initial endowments, the input layer receives the state space of the economy as input data, $\{k_{i,t}, d_{i,t-1}, M_{i,t}, y \times d_{i,t-1}, R \times a_{i,t-1}\}_{t=1}^{T=3}$, for $i \in [1, \ldots, \textit{length of episode}]$. The definition of "Episode" in this paper is a set of simulated periods, which is the training data (Azinovic et al., 2022). $y \times d_{i,t-1}$ is labor income at time t and $R \times a_{i,t-1}$ is income from savings in period t. Note that all the other variables are redundant if $\{k_t, d_{t-1}\}$ are provided. Following Azinovic et al. (2022), I include redundant input variables for better results. I assume that the initial endowment, k_1, follows a uniform distribution $U[0, 65]$ and $d_0 = 0$.

I build three neural networks for each of the three agents. The intuition for not building one large neural network for all three agents is to make the weights of the separate neural networks approximate each agent's policy function.

The 5 hidden layers all have 64 nodes with 1 bias node. The first and third hidden layers are the discontinuous layers explained in Section 4.3. The activation function for the discontinuous layers is selected as an elu activation function. I select a smooth activation function since the discontinuous layer already contains a discontinuity with the jump function. The second layer uses the elu activation function and the fourth and fifth layers use the ReLU activation function. The choice was made arbitrarily but the intuition was that since the Universal Approximation in Kidger and Lyons (2020) uses a ReLU activation function, I wanted the weighted combination of the input layer to be passed to

a layer with a ReLU activation function. The use of the elu activation function for the second layer is to mitigate the vanishing gradient problem.

The first and second neural networks have a sigmoid activation function at the output layer which consists of 2 nodes. The first node approximates the percentage of saving out of total income, $\frac{a_t}{M_t}$, and the other node approximates the probability of retirement. The last neural network has a ReLU activation function for the 2 output layer nodes. These 2 output layer nodes each approximate the Lagrangian multiplier for savings, μ_t^a for each t excluding the last period.

Given k_t and d_{t-1}, we approximate the savings, a_t, using the neural network by multiplying M_t to $\frac{a_t}{M_t}$. Then, we have all the right-hand side variables of the budget constraint, $c_t = Rk_t + yd_{t-1} - a_t$, allowing us to calculate consumption, c_t.

Next, we need to formulate a cost function to approximate the KKT conditions 6-11. From Equation 12, the relative Euler equation error is,

$$e^i_{REE_c}(\rho) = \left(\frac{\beta R}{c_{i,t+1}} + \mu^a_{i,t}\right) \times c_{i,t} - 1. \tag{21}$$

The non-negativity constraints are replaced with punishment functions that produce a large positive value if the consumption or saving is negative. Let $\epsilon > 0$ be a small positive number.

$$punish^i_{c_t}(\rho) = 1/\epsilon * \max\{-c_{i,t}, 0\} \tag{22}$$

$$punish^i_{a_t}(\rho) = 1/\epsilon * \max\{-a_{i,t}, 0\} \tag{23}$$

The punishment for negative values of consumption and savings can be omitted in the neural network used here since the output node activation function ensures that both are positive. However, if the output node activation function can generate negative values, punishment is needed.

The next part of the cost function is from Equation 9.

$$Multiplier^i_{a_t} = \mu^a_{i,t} a_{i,t} \tag{24}$$

The remaining part is to design a method to approximate the discrete retirement choice, d_t. The method borrows the idea proposed by Maliar and Maliar (2022) and formulate a logistic regression problem. We need the y_{label} to formulate the maximum log-likelihood as in Equation 18. Maliar and Maliar (2022) approximate the value functions $v_t(M, 0)$ and $v_t(M, 1)$ using a similar method to Chang and Kim (2007). Once the

value functions are approximated, $y_{label,t} = 1$ if $v_t(M, 0) < v_t(M, 1)$ and 0 otherwise. In this paper, I do not approximate the value functions. Instead, I use the result presented in equation 3. Theoretically, we know that the optimal decision to retire at period $T - \tau$ is when the income is above $\bar{M}_{T-\tau}$. Thus, $y_{label,t} = 1$ if $M > \bar{M}_{T-\tau}$ while $y_{label} = 0$ otherwise at period $T - \tau$. The true labels are created for $\forall t \in \{1, \ldots, T\}$ and for all i. After the labels are created, then we can use equation 18 and change it into a minimization problem.

$$CrossEntropy_i = -lnL(\theta) = -\frac{1}{T-1}\left(\sum_{t=1}^{T-1}[y^i_{label,t} ln(p_t(X^i;\theta)) + (1 - y^i_{label,t}) ln(1 - p_t(X^i;\theta)\right) \tag{25}$$

The sum is over $T - 1$ since the last period optimal decision is to retire. Inside the algorithm, the last period optimal decision will be hard coded to $d_T = 0$.

Let $\mathcal{D}_{train}$ be the training data. Define MSE_i as

$$MSE_i = \frac{1}{T-1}\sum_{t=1}^{T-1} e^i_{REE_c}(\rho)^2 + punish^i_{c_t}(\rho)^2 + punish^i_{a_t}(\rho)^2 + \left(Multiplier^i_{a_t}\right)^2.$$

The overall cost function is the sum of the mean squared error of $e^i_{REE_c}(\rho)$, $punish^i_{c_t}(\rho)$, $punish^i_{a_t}(\rho)$, $Multiplier^i_{a_t}$ added with the $CrossEntropy_i$.

$$l(\rho) = \frac{1}{|\mathcal{D}_{train}|}\left[\frac{1}{2}\sum_{x\in\mathcal{D}_{train}} MSE_i + \frac{1}{2}\sum_{x\in\mathcal{D}_{train}} CrossEntropy_i\right] \tag{26}$$

4.5 Training sample

The generation of the training sample follows Azinovic et al. (2022). First, I generate a feasible random starting state, $j = 1$. With the randomly initialized parameters of the neural network, I generate the next period, $j = 2$, capital and retirement decision using the starting state as input of the neural network. Then, I generate $j = 3$ using the $j = 2$ state as input. This process continues for $J - 1$ times. In this way, I created the first "episode", $\mathcal{D}^0_{train}$, where the superscript 0 stands for the first episode. The length of the episode is J. Next, $\mathcal{D}^0_{train}$ is divided into mini-batches and the model is trained for each mini-batch with a variant of gradient descent. If this is completed, the first epoch is over. With the same episode, $\mathcal{D}^0_{train}$, random mini-batches are created and the training is done again. If this is completed, the second epoch is over. If all the specified number

of epochs are over, the last simulated state in $\mathcal{D}^0_{train}$ is used as the starting state for the generation of the second episode. When generating the second episode, the updated parameters of the neural network are used. Azinovic et al. (2022) point out that this way of generating the training data helps the model to train on the states close to the ergodic set of the economy as the episodes move forward. In this paper, the model has no uncertainty, which leads to the same state space without introducing randomness. Thus, one randomness is included. The initial capital endowment follows a uniform distribution, $U[0, 65]$. The meaning is that the initial endowment is given at random.

4.6 Algorithm

This section shows the pseudo-code of the algorithm.

Algorithm 1

J (length of an episode), N^{epochs} (number of epochs on each episode), N^{iter} (maximum number of iterations), ρ^0 (initial parameters of the neural network), $\mathbf{x}_1^0$ (initial state generated with a random process), $i = 0$ (set counter), α^{learn} (the starting learning rate), N^{decay} (learning rate decay step)
while $i < N^{iter}$ **do**
  if $i = 0$ **then**
    With ρ^0 and $\mathbf{x}_1^0$, create $\mathcal{D}^i_{train} \leftarrow \{x_1^i, x_2^i, \ldots, x_J^i\}$
  else if $i \neq 0$ **then**
    With ρ^{i-1} and $\mathbf{x}_J^{i-1}$, create $\mathcal{D}^i_{train} \leftarrow \{x_1^i, x_2^i, \ldots, x_J^i\}$
  end if
  for $j \in [1, \ldots, N^{epochs}]$ **do**
    for $k \in [1, \ldots, length(\rho)]$ **do**

$$\rho_k^{i,j} = \rho_k^{i,j-1} - \alpha^{learn} \frac{\partial l(p)}{\partial \rho_k^{i,j-1}} \; \forall k \in \{1, \ldots, length(\rho)\} \tag{27}$$

      $\rho^{i+1,j=0} \leftarrow \rho^{i,j=N^{epochs}}$, $\rho^{i+1} \leftarrow \rho^{i,j=N^{epochs}}$
      Update the learning rate (Adam)
    end for
  end for
  $i \leftarrow i + 1$
end while
$\rho^{final} \leftarrow \rho^i$

4.7 Model parameters and hyper-parameters of the algorithm

In this section, I report the model parameters and hyper-parameters of the algorithm in table 1. The code is written and run in TensorFlow 2.8.

Table 1: Model parameters and hyper-parameters

Model parameters	value
Number of periods	3
Discount rate: β	0.95
Fixed labor income: y	20
Non-stochastic interest rate: R	1.2
Dis-utility of work: δ	1
Hyper-parameters	**value**
Number of episodes	2000
Length of episode: J	12,000
mini-batch size	2000
epoch	20
Optimizer	Adam optimizer
Learning rate	0.00002
1st and 2nd Neural Network	
Number of input	15
Number of hidden layers	5
Number of nodes in hidden layers	all 64 and 1 bias
Number of output nodes	2
Activation function of hidden layers	elu and ReLU
Activation function of the output layer	sigmoid
3rd Neural Network	
Number of input	15
Number of hidden layers	5
Number of nodes in hidden layers	all 64 and 1 bias
Number of output nodes	2
Activation function of hidden layers	elu and ReLU
Activation function of the output layer	ReLU

5 Numerical results

5.1 Main result

In this section, I compare the predictions made by the neural networks and the analytic solutions of the policy functions. The neural networks make highly accurate predictions

for consumption if the optimal policy function for consumption is not near the jump points. However, some predictions are inconsistent with the analytic solution at or near the jump points, especially for individuals of age 1. Meanwhile, the predictions for the discrete retirement choice are highly accurate.

Figure 4 shows the overall performance of the algorithm. The first panel in Figure 4 shows that the total cost decreased to around -3.3 in logarithm with base 10 after 2000 iterations. The second panel in the first row of 4 shows that the relative Euler equation error is on average -2 while the maximum is below -0.2 for both individuals of ages 1 and 2. The third panel in the first row of 4 shows the average, minimum, and maximum amount of capital for individuals of ages 1, 2, and 3.

The panels in the second row in 4 show the results for the prediction of consumption. Each panel represents the individuals of ages 1, 2, and 3, respectively. The far left panel is for age 1 while the far right panel is for age 3. To compare with the analytic solution, panels in the fourth row are visually more helpful. The blue circles represent the analytic solution while the red dots represent the prediction made by the trained neural network. The consumption is close to the analytic solution but some points are off the blue circles near or at the point where the jump occurs. The average of MSE_i defined in Section 4.4 is around -4.3.

Figure 5 shows larger pictures of the first two panels of the last row in Figure 4. For the individual of age 1, the red dots mostly coincide with the blue circles when the income level is not near the jump points. However, the red dots slightly overshoot and then undershoot the blue circles before the first jump which occurs around 29. Also, some red dots are above the blue circles at the second jump which occurs around 46. For the individual of age 2, the red dots mostly coincide with the blue circles, indicating that the prediction is relatively good for the individual of age 2.

The panels in the third row in 4 show the results for the prediction of retirement. The vertical axis named "prob of working" represents the probability of working depending on the income of the individual at age 1 and 2, respectively. We can see from 4 that the neural networks completely separate the probability of working at the analytic solution which is around 46 for for the individual of age 1 and around 29 for the individual of age 2. The cost for the discrete choice problem which is measured as cross entropy is around -3.3 after 2000 iterations.

The performance of the neural network is highly sensitive to the hyperparameters. For instance, if the learning rate changes or the structure of the network changes, the approximation to the analytic solution for agent 1 becomes worse off.

6 Conclusion

This paper solves an OLG version of the discrete-continuous dynamic choice problem using deep neural networks. The neural networks that I designed make highly accurate predictions for consumption if the optimal policy function for consumption is not near the jump points. However, some predictions are off the analytic solution at or near the jump points, especially for individuals of age 1. Meanwhile, the predictions for the discrete retirement choice are highly accurate. Further research is needed to obtain a better approximation near the jump points. Moreover, theoretical justification is needed for the design of the neural networks in this paper which is highly sensitive to changes.

References

Azinovic, Marlon, Luca Gaegauf, and Simon Scheidegger (2022) "DEEP EQUILIBRIUM NETS," *International Economic Review*, 10.1111/iere.12575.

Carroll, Christopher D. (2006) "The method of endogenous gridpoints for solving dynamic stochastic optimization problems," *Economics Letters*, 91 (3), 312–320, https://doi.org/10.1016/j.econlet.2005.09.013.

Chang, Yongsung and Sun-Bin Kim (2007) "Heterogeneity and Aggregation: Implications for Labor-Market Fluctuations," *American Economic Review*, 97 (5), 1939–1956, 10.1257/aer.97.5.1939.

Clausen, Andrew and Carlo Strub (2016) "A General and Intuitive Envelope Theorem,"Technical report, `https://EconPapers.repec.org/RePEc:edn:esedps:274`.

Della Santa, Francesco and Sandra Pieraccini (2023) "Discontinuous neural networks and discontinuity learning," *Journal of Computational and Applied Mathematics*, 419, 114678, https://doi.org/10.1016/j.cam.2022.114678.

French, Eric and John Bailey Jones (2011) "THE EFFECTS OF HEALTH INSURANCE AND SELF-INSURANCE ON RETIREMENT BEHAVIOR," *Econometrica*, 79 (3), 693–732.

Iskhakov, Fedor, Thomas H. Jørgensen, John Rust, and Bertel Schjerning (2017) "The endogenous grid method for discrete-continuous dynamic choice models with (or without) taste shocks," *Quantitative Economics*, 8 (2), 317–365, https://doi.org/10.3982/QE643.

Kidger, Patrick and Terry Lyons (2020) "Universal Approximation with Deep Narrow Networks," in Abernethy, Jacob and Shivani Agarwal eds. *Proceedings of Thirty Third Conference on Learning Theory*, 125 of Proceedings of Machine Learning Research, 2306–2327: PMLR, 09–12 Jul, `https://proceedings.mlr.press/v125/kidger20a.html`.

Kingma, Diederik P. and Jimmy Ba (2015) "Adam: A Method for Stochastic Optimization," *CoRR*, abs/1412.6980.

Krusell, Per and Anthony A. Smith, Jr. (1998) "Income and Wealth Heterogeneity in the Macroeconomy," *Journal of Political Economy*, 106 (5), 867–896, 10.1086/250034.

Llanas, Bernardo, Sagrario Lantarón, and Francisco Sáinz (2008) "Constructive Approximation of Discontinuous Functions by Neural Networks," *Neural Processing Letters*, 27, 209–226, 10.1007/s11063-007-9070-9.

Maliar, Lilia and Serguei Maliar (2022) "Deep learning classification: Modeling discrete labor choice," *Journal of Economic Dynamics and Control*, 135, 104295, https://doi.org/10.1016/j.jedc.2021.104295.

Rust, John and Christopher Phelan (1997) "How Social Security and Medicare Affect Retirement Behavior In a World of Incomplete Markets," *Econometrica*, 65 (4), 781–831, `http://www.jstor.org/stable/2171940`.

Selmic, Rastko R. and Frank L. Lewis (2002) "Neural-network approximation of piecewise continuous functions: application to friction compensation," *IEEE transactions on neural networks*, 13 3, 745–51.

A Figures

Figure 4: Results

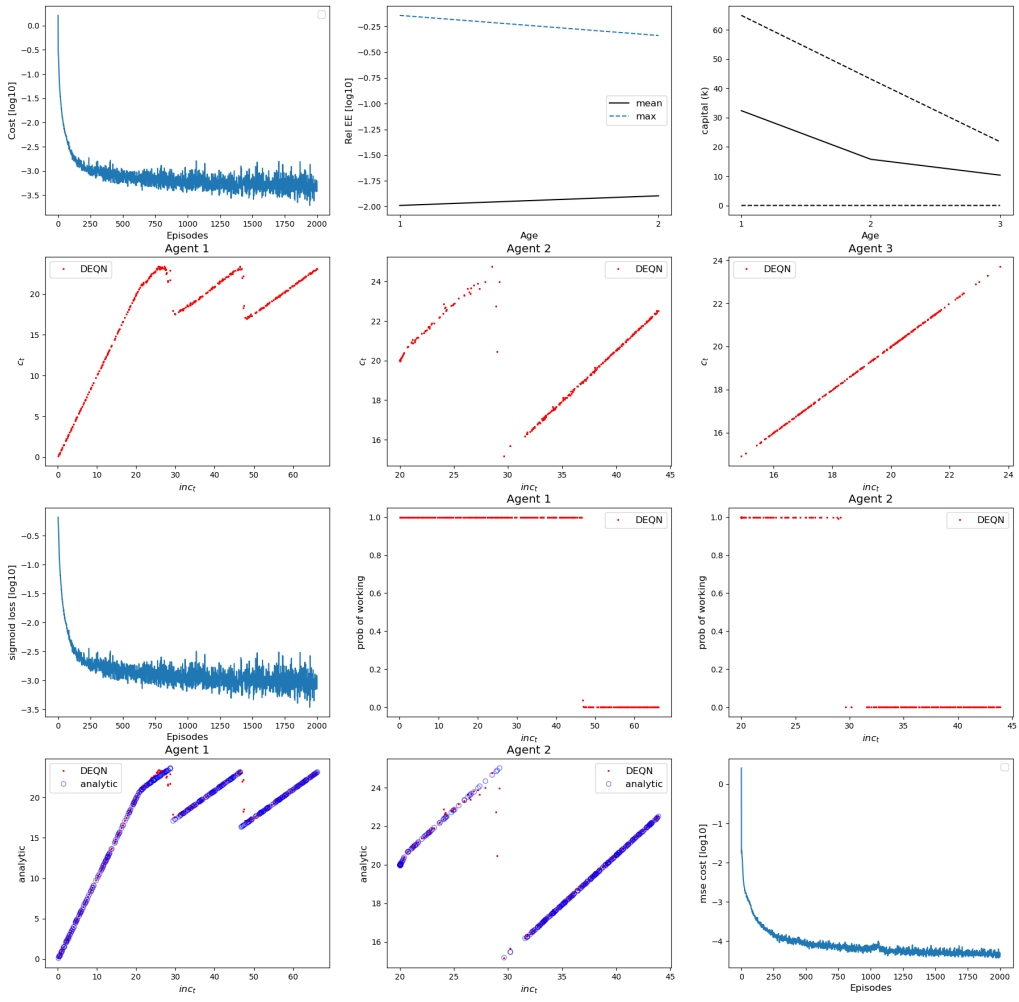


Figure 5: Compare with analytic solution