|  |  |  |
| --- | --- | --- |
| **Unit 1** | | |
| 1. | What are the different types of mathematical objects in Linear Algebra? **Scalars, Vectors, Matrices, Tensors** | |
| 2. | Differentiate between scalar, vectors, metrices and tensors. | |
| 3. | Write a short note on the different multiplication techniques to multiply matrix. - **Matrix Product, Dot Product** | |
| 4. | Explain with suitable example how to multiply matrices and vectors. | |
| 5. | Write a note on identity and inverse of matrices. | |
| 6. | What is linear dependency? Explain with suitable example.   * In the theory of vector spaces, a set of vectors is said to be linearly dependent if at least one of the vectors in the set can be defined as a linear combination of the others * Having n ≥ m is only a necessary condition for every point to have a solution. It is not a sufficient condition, because it is possible for some of the columns to be redundant. * Consider a 2 × 2 matrix where both of the columns are identical. This has the same column space as a 2 × 1 matrix containing only one copy of the replicated column. * In other words, the column space is still just a line, and fails to encompass all of R2, even though there are two columns. * Formally, this kind of redundancy is known as linear dependence. | |
| 7. | Explain the terms linear dependence, linear independence and span. | |
| 8. | What are norms? Explain the different types of norms. | |
| 9. | Write a short note on Eigen Decomposition.   * Many mathematical objects can be understood better by breaking them into constituent parts, or finding some properties of them that are universal, not caused by the way we choose to represent them. * For example, integers can be decomposed into prime factors. * we can discover something about the true nature of an integer by decomposing it into prime factors * we can also decompose matrices in ways that show us information about their functional properties that is not obvious from the representation of the matrix as an array of elements. * One of the most widely used kinds of matrix decomposition is called eigen decomposition * In Eigen Decomposition we decompose a matrix into a set of eigenvectors and eigenvalues. * An eigenvector of a square matrix A is a non-zero vector v such that multiplication by A alters only the scale of v: * The scalar λ is known as the eigenvalue corresponding to this eigenvector * If v is an eigenvector of A, then so is any rescaled vector sv for s ∈ , s R = 0. * The eigen decomposition of A is then given by A = V diag(λ)V −1. | |
| 10. | Write a short note on Singular value Decomposition.   * The singular value decomposition (SVD) provides another way to factorize a matrix, into singular vectors and singular values. * The SVD allows us to discover some of the same kind of information as the eigen decomposition. * Every real matrix has a singular value decomposition, but the same is not true of the eigenvalue decomposition. * For example, if a matrix is not square, the eigen decomposition is not defined, and we must use a singular value decomposition instead. * The singular value decomposition is similar, except this time we will write A as a product of three matrices: A = UDVT. * Most useful feature of the SVD is that we can use it to partially generalize matrix inversion to non-square matrices | |
| 11. | Explain the concept of overflow and underflow.  **Underflow**   * All real numbers, have some approximation error when we represent the number in the computer. * In many cases, this is just rounding error * Rounding error is problematic, especially when it compounds across many operations * One form of rounding error that is particularly devastating is underflow * Underflow occurs when numbers near zero are rounded to zero * Many functions behave qualitatively differently when their argument is zero rather than a small positive number   **Overflow**   * Highly damaging form of numerical error is overflow. * Overflow occurs when numbers with large magnitude are approximated as ∞ or −∞. * Further arithmetic will usually change these infinite values into not-a-number values. | |
| 12. | Write a short note on Poor conditioning.   * Conditioning refers to how rapidly a function changes with respect to small changes in its inputs * Functions that change rapidly when their inputs are perturbed slightly can be problematic for scientific computation because rounding errors in the inputs can result in large changes in the output * Consider the function f(x) = A−1x. When A ∈ Rn×n has an eigenvalue decomposition, its condition number is * This is the ratio of the magnitude of the largest and smallest eigenvalue * When this number is large, matrix inversion is particularly sensitive to error in the input * This sensitivity is an intrinsic property of the matrix itself * Poorly conditioned matrices amplify pre-existing errors when we multiply by the true matrix inverse | |
| 13. | Explain gradient descent optimization with its algorithm.   * Most deep learning algorithms involve optimization of some sort. * Optimization refers to the task of either minimizing or maximizing some function f (x) by altering x * The function we want to minimize or maximize is called the objective function or criterion. * When we are minimizing it, we may also call it the cost function, loss function, or error function. * Suppose we have a function y = f (x), where both x and y are real numbers. The derivative of this function is denoted as f ‘(x) or as dy/dx * We can reduce f (x) by moving x in small steps with opposite sign of the derivative. This technique is called gradient descent * Points where f(x) = 0 are known as critical points or stationary points * A local minimum is a point where f (x) is lower than at all neighboring points, so it is no longer possible to decrease f(x) by making infinitesimal steps * local maximum is a point where f (x) is higher than at all neighboring points, so it is not possible to increase f (x) by making infinitesimal steps * The gradient generalizes the notion of derivative to the case where the derivative is with respect to a vector * This is known as the method of steepest descent or gradient descent. | |
| 14. | What are the different points that we can come across while performing gradient decent operation?   * Points where f(x) = 0 are known as critical points or stationary points * A local minimum is a point where f (x) is lower than at all neighboring points, so it is no longer possible to decreasef(x) by making infinitesimal steps * local maximum is a point where f (x) is higher than at all neighboring points, so it is not possible to increase f (x) by making infinitesimal steps | |
| 15. | What is constraint optimization?   * Sometimes we wish not only to maximize or minimize a function f(x) over all possible values of x * Instead, we may wish to find the maximal or minimal value of f (x) for values of x in some set S. * This is known as constrained optimization. * Points x that lie within the set S are called feasible points in constrained optimization terminology * One simple approach to constrained optimization is simply to modify gradient descent taking the constraint into account * If we use a small constant step size €, we can make gradient descent steps, then project the result back into S. * When possible, this method can be made more efficient by projecting the gradient into the tangent space of the feasible region before taking the step or beginning the line search * The Karush–Kuhn–Tucker (KKT) approach1 provides a very general solution to constrained optimization * With the KKT approach, we introduce a new function called the generalized Lagrangian or generalized Lagrange function. | |
| 16. | What role does derivatives play in gradient decent algorithm?   * The derivative of this function is denoted as f ‘(x) or as dy/dx * The derivative is useful for minimizing a function because it tells us how to change x in order to make a small improvement in y * When f ‘(x) = 0, the derivative provides no information about which direction to move. | |
| 17. | Write a short note on rounding error. | |
| **Unit 2** | |
| 19. | Explain the architecture of Deep feed forward neural network.   * Deep feedforward networks, also often called feedforward neural networks, or multilayer perceptron (MLPs) * These are the quintessential deep learning models * The goal of a feedforward network is to approximate some function f∗ * Feedforward neural networks are called networks because they are typically represented by composing together many different functions * The model is associated with a directed acyclic graph * we might have three functions f (1), f (2), and f (3) connected in a chain, to form f (x) = f(3)(f (2)(f (1)(x))). * These chain structures are the most commonly used structures of neural networks. * In this case, f (1) is called the first layer of the network, f (2) is called the second layer, and so on * The final layer of a feedforward network is called the output layer. * During neural network training, we drive f (x) to match f∗(x). * The training data does not show the desired output for each of these layers, hence these layers are called hidden layers. |
| 21. | What is deep feed forward networks? Explain with examples.   * Deep feedforward networks, also often called feedforward neural networks, or multilayer perceptron (MLPs) * These are the quintessential deep learning models * The goal of a feedforward network is to approximate some function f∗ * These models are called feedforward because information flows through the function being evaluated from x, through the intermediate computations used to define f, and finally to the output y * Feedforward neural networks are called networks because they are typically represented by composing together many different functions * The model is associated with a directed acyclic graph |
| 23. | Write a short note on Cost function or Loss function.  **Cost Function**   * To apply gradient-based learning we must choose a cost function * An important aspect of the design of a deep neural network is the choice of the cost function * The cost functions for neural networks are more or less the same as those for other parametric models, such as linear models * We use the cross-entropy between the training data and the model’s predictions as the cost function * The total cost function used to train a neural network will often combine one of the primary cost functions described here with a regularization term |
| 24. | Write a note on cross entropy error. |
| 25. | Write a short note on gradient-based learning.   * The largest difference between the linear models we have seen so far and neural networks is that the nonlinearity of a neural network causes most interesting loss functions to become non-convex. * This means that neural networks are usually trained by using iterative, gradient-based optimizers that merely drive the cost function to a very low value * The iterative gradient-based optimization algorithms used to train feedforward networks * Computing the gradient is slightly more complicated for a neural network, but can still be done efficiently and exactly. * To apply gradient-based learning we must choose a cost function, and we must choose how to represent the output of the model |
| 26. | Write a short note on softmax activation function.   * To represent a probability distribution over a discrete variable with n possible values, we may use the softmax function * This can be seen as a generalization of the sigmoid function which was used to represent a probability distribution over a binary variable. * Softmax functions are most often used as the output of a classifier, to represent the probability distribution over n different classes * More rarely, softmax functions can be used inside the model itself, if we wish the model to choose between one of n different options for some internal variable * Like the sigmoid, the softmax activation can saturate * In the case of the softmax, there are multiple output values. * These output values can saturate when the differences between input values become extreme * When the softmax saturates, many cost functions based on the softmax also saturate, |
| 28. | Write a short note on RELU. (Rectified Linear Units)   * Rectified linear units use the activation function g(z) = max{0, z}. * Rectified linear units are easy to optimize because they are so similar to linear units * The only difference between a linear unit and a rectified linear unit is that a rectified linear unit output zero across half its domain * This makes the derivatives through a rectified linear unit remain large whenever the unit is active * The gradients are not only large but also consistent * The second derivative of the rectifying operation is 0 almost everywhere, and the derivative of the rectifying operation is 1 everywhere that the unit is active * Rectified linear units are typically used on top of an affine transformation: * Rectified linear units and all of the generalizations of them are based on the principle that models are easier to optimize if their behavior is closer to linear |
| 29. | Explain universal approximation theorem. |
| 30. | Write a short note on L1 Regularization.   * While L2 weight decay is the most common form of weight decay, there are other ways to penalize the size of the model parameters. * Another option is to use L1 regularization. * Formally, L1 regularization on the model parameter w is defined as: * L1 weight decay controls the strength of the regularization by scaling the penalty Ω using a positive hyperparameter α * L1 regularization results in a solution that is more sparse * The sparsity property induced by L1 regularization has been used extensively as a feature selection mechanism |
| 31. | Write a short note on L2 Regularization.   * Regularization has been used for decades prior to the advent of deep learning * One of the simplest and most common kinds of parameter norm penalty: the L2 parameter norm penalty commonly known as weight decay * This regularization strategy drives the weights closer to the origin1 by adding a regularization termto the objective function * L2 regularization is also known as ridge regression or Tikhonov regularization. |
| 32. | Explain dataset augmentation. |
| 33. | Write a short note on semi supervised learning.   * In the paradigm of semi-supervised learning, both unlabeled examples from P (x) and labeled examples from P (x, y) are used to estimate P (y | x) or predict y from x * semi-supervised learning usually refers to learning a representation h = f (x) * The goal is to learn a representation so that examples from the same class have similar representations * Instead of having separate unsupervised and supervised components in the model, one can construct models in which a generative model of either P (x) or P(x, y) shares parameters with a discriminative model of P(y | x). |
| 34. | Write a short note on multitask learning.   * Multi-task learning is a way to improve generalization by pooling the examples arising out of several tasks. * Multi-task learning can be cast in several ways in deep learning frameworks * The lower layers of a deep network can be shared across the tasks * Improved generalization and generalization error bounds can be achieved because of the shared parameters, |
| 35. | What is early stopping? How it is useful in regularization?   * When training large models with sufficient representational capacity to overfit the task, we often observe that training error decreases steadily over time, but validation set error begins to rise again. * Every time the error on the validation set improves, we store a copy of the model parameters * When the training algorithm terminates, we return these parameters, rather than the latest parameters * The algorithm terminates when no parameters have improved over the best recorded validation error for some pre-specified number of iterations * The early stopping meta-algorithm for determining the best amount of time to train * This meta-algorithm is a general strategy that works well with a variety of training algorithms and ways of quantifying error on the validation set. * This strategy is known as early stopping * It is probably the most commonly used form of regularization in deep learning. * Its popularity is due both to its effectiveness and its simplicity. * Early stopping is a very unobtrusive form of regularization, in that it requires almost no change in the underlying training procedure |
| 36. | Explain parameter tying and parameter sharing. |
| 37. | Write a short note on bagging.   * Bagging (short for bootstrap aggregating ) is a technique for reducing generalization error by combining several models * The idea is to train several different models separately, then have all of the models vote on the output for test examples * This is an example of a general strategy in machine learning called model averaging * Techniques employing this strategy are known as ensemble methods. * The reason that model averaging works is that different models will usually not make all the same errors on the test set * Bagging involves constructing k different datasets. * Each dataset has the same number of examples as the original dataset, but each dataset is constructed by sampling with replacement from the original dataset. * This means that, with high probability, each dataset is missing some of the examples from the original dataset and also contains several duplicate examples |
| 38. | Explain dropout algorithm.   * Dropout provides a computationally inexpensive but powerful method of regularizing a broad family of models. * To a first approximation, dropout can be thought of as a method of making bagging practical for ensembles of very many large neural networks * Specifically, dropout trains the ensemble consisting of all sub-networks that can be formed by removing non-output units from an underlying base network * In most modern neural networks, based on a series of affine transformations and nonlinearities, we can effectively remove a unit from a network by multiplying its output value by zero * We present the dropout algorithm in terms of multiplication by zero for simplicity, but it can be trivially modified to work with other operations that remove a unit from the network * Dropout training is not quite the same as bagging training * In the case of dropout, the models share parameters, with each model inheriting a different subset of parameters from the parent neural network * Dropout is more effective than other standard computationally inexpensive regularizers * One advantage of dropout is that it is very computationally cheap. |
| 39. | What is tensor flow? Explain how it can help to understand computational graph. |
| 40. | Explain the fundamental elements of tensor flow. |
| 41. | What is back propagation? How back propagation works?   * The back-propagation algorithm, often simply called backprop, allows the information from the cost to then flow backwards through the network, in order to compute the gradient. * back-propagation refers only to the method for computing the gradient * back-propagation is often misunderstood as being specific to multilayer neural networks, but in principle it can compute derivatives of any function * Many machine learning tasks involve computing other derivatives, either as part of the learning process, or to analyze the learned model. * The backpropagation algorithm can be applied to these tasks as well, and is not restricted to computing the gradient of the cost function with respect to the parameters |