Building and predicting a neural network in python

Ravshan Zulunov\textsuperscript{1*}, Umidjon Akhundjanov\textsuperscript{1}, Khurshid Musayev\textsuperscript{1}, Bakhrom Soliyev\textsuperscript{1}, Ahror Kayumov\textsuperscript{1}, and Mukhammadmullo Asraev\textsuperscript{1}

\textsuperscript{1}Ferghana branch of Tashkent university of information technologies named after Muhammad al-Kwarizmi, Ferghana, Uzbekistan

\textbf{Abstract.} The article examines the practical implementation of Artificial Intelligence technologies, Machine Learning and Deep Learning technologies. Methods of construction and prediction of neural networks are considered using special libraries of Python language. Linear regression and NumPy it is suggested to use libraries. Keywords: Artificial Intelligence, Python libraries, neural network, prediction, regression.

\section{1 Introduction}

Enter. The goal of using artificial intelligence is to make computers think like humans. This may sound like a new thing, but this field was born in the 1950s. Sudoku puzzle you need to write a Python program that uses artificial intelligence to solve. The way to achieve this is to write conditional statements and check the constraints to see if we can fit a number in each position. This Python script is an Artificial Intelligence application because we have programmed a computer to solve a problem.

Materials and methods. Machine Learning (ML) and Deep Learning (DL) are both problem solving approaches. The difference between these methods and Python scripting is that ML and DL use training data instead of hard-coded rules, but they can all be used to solve problems with Artificial Intelligence.

Machine learning is a technique that trains a system to solve a problem, rather than precisely programming rules. Going back to the Sudoku example from the previous section, to solve a problem using Machine Learning, we need to collect data from solved Sudoku games and train a statistical model. Statistical models are mathematically formalized ways of predicting the behavior of a phenomenon.

learning with a set of inputs and known outputs. The challenge is to use this data set to train a model that predicts the correct output based on the input. The figure below shows the workflow for training a model using supervised training:

\* Corresponding author: ismoilov_2722262@mail.ru

\textcopyright{} The Authors, published by EDP Sciences. This is an open access article distributed under the terms of the Creative Commons Attribution License 4.0 (https://creativecommons.org/licenses/by/4.0/).
Fig. 1. Workflow for training a machine learning model.

A combination of training data with a machine learning algorithm creates a model. You can then use this model to make predictions based on new data. Scikit-Learn is a popular Python machine learning library that provides many supervised and unsupervised learning algorithms. See scikit-learn for more information on this by train_test_split() is considered to separate the data set [1].

The goal of supervised learning tasks is to make predictions based on new, unseen data. To do this, we assume that this unseen data follows the same probability distribution as the training data set. If this distribution changes in the future, we need to retrain your model using the new training data set.

2 Methods

Feature development: Prediction problems become more complex when you use different types of data as input. The Sudoku problem is relatively simple because we are dealing directly with numbers. What if we want to train a model to predict the mood of a sentence? Or what if you have an image and want to know if there is a cat in it?

Another name for input data is feature, and feature engineering is the process of extracting features from raw data. When you work with a variety of data, you need to find ways to present that data in order to get meaningful information out of it.

A feature engineering technique is lemmatization, where we remove the declension of words in a sentence. For example, the inflectional forms of the verb "to look" such as "to watch", "to watch", "to observe" are shortened to their lemma or main form: "to look" [2].

If we use arrays to store each word of the corpus, by applying lemmatization we get a less sparse matrix. This can improve the performance of some machine learning algorithms. The figure below shows the representation process using lemmatization and the bag-of-words model:
First, the inflectional form of each word is mapped to its lemma. Then the number of occurrences of this word is counted. The result is an array containing the number of occurrences of each word in the text.

3 Results and Discussions

Deep learning: A method that allows you to independently decide which features are important, rather than using feature engineering techniques. This means that with deep learning we can bypass the feature development process.

It's a good thing not to do feature engineering because the process gets more complicated as the data set gets more complex. For example, what information can we extract from an image of a person's face to predict their mood? With neural networks, we don't have to worry about this because the networks can learn the features themselves.

Neural networks are parallel computing devices that attempt to create a computer model of the brain. The main goal is to develop a system to perform various computational tasks faster than conventional systems. These tasks include pattern recognition and classification, approximation, optimization, and data clustering.

An artificial neural network (ANN) is an efficient computing system, the central idea of which is derived from the analogy with biological neural networks. SNTs are also called artificial neural systems, parallel distributed processing systems, and connection systems. SNT takes large units that are interconnected in some pattern to provide communication between them. These units, also called nodes or neurons, are simple processors that work in parallel.

Each neuron is connected to every other neuron by synapses. Each link is associated with a weight that has information about the input signal. This is the most useful information for the neurons when solving a particular problem, because the weight usually triggers the signal being transmitted. Each neuron has its own internal state, which is called an activation signal. After combining the input signals and activation rules, the output signals obtained can be sent to other devices [3].

A neural network is a system that learns to make predictions by performing the following steps:
1. Get login information
2. Prediction
3. Comparing the prediction with the expected result
4. Adjusting its internal state to predict correctly next time

Vectors, layers and linear Regression is one of the building blocks of neural networks. Data is stored as vectors, and in Python we store these vectors in arrays. Each layer transforms the data coming from the previous layer. You can think of each layer as a stage of feature development, as each layer releases some representation of the data previously obtained.

The interesting thing about neural network layers is that the same computations can take information from any data. This means that it does not matter whether we use image or text data. The process of extracting meaningful data and training a deep learning model is the same for both scenarios.

In the figure below, we can see an example of a two-layer network architecture:

![Two-layer neural network](image_url)

**Fig. 3.** Two-layer neural network.

Each level transforms the data from the previous level by applying some mathematical operations.

Training a neural network is like a trial and error process. Imagine that we are playing darts for the first time. On the first shot, we want to hit the center point of the target. Usually the first shot is taken to understand how the height and speed of your hand affects the result. If we see that the dart is above the center point, we adjust our arm to throw it a little lower, and so on [4].

Here are the steps to hitting the center of the dart board:
Remember that you continue to evaluate the error by observing where the dart lands (step 2). We will continue until you finally hit the center of the target.

In neural networks, the process is very similar: we start with random weights and uncertainty vectors, make a prediction, compare it to the desired result, and adjust the vectors to make a better prediction next time. The process continues until the difference between the forecast and the true targets is minimal, leading to overfitting and underfitting scenarios.

Working with neural networks consists of performing operations with vectors. We represent vectors as multidimensional arrays. Vectors are useful in deep learning mainly because of one operation: the dot product. The dot product of two vectors shows how similar they are in direction and is measured by the magnitude of the two vectors.

The basic vectors in a neural network are weight and bias vectors. Roughly speaking, we want our neural network to test whether the input is similar to other inputs it has already seen. If the new input is the same as the previously seen input, the output is the same. Thus, we get the prediction result.

Regression is used when the relationship between a dependent variable and two or more independent variables needs to be evaluated. Linear regression is a method used when the relationship between variables is assumed to be linear. A linear relationship is a relationship in which there is a direct relationship between the independent variable and the dependent variable.

By modeling the relationship between variables as linear, we can express the dependent variable as a weighted sum of the independent variables. Thus, each independent variable is multiplied by a vector called a weight. In addition to the weights and independent variables, we also add another bias vector. Sets the outcome when all other independent variables are equal to zero.

As a real-life example of building a linear regression model, imagine that you want to train a model to predict the price of a house based on its square footage and age. We are linear we decide to model these relationships using regression. The following block of code shows how you can write a linear regression model for a given problem in pseudocode:

\[
\text{price} = (\text{weights_area} \times \text{area}) + (\text{weights_age} \times \text{age}) + \text{bias}
\]
In the above example, there are two weights: weights_area and weights_age. The training process consists of adjusting the weights and biases so that the model can predict the correct price value. For this we need to calculate the forecast error and update the weights accordingly. This is the basics of how a neural network works [5, 6].

Results. Now it's time to see how to apply these concepts with Python. Starting to build our first neural network: The first step in building a neural network is to create an output from the input [7-14]. We do this by creating a weighted sum of the variables. The first thing we need to do is provide input with Python and NumPy (HYPERLINK https://realpython.com/numpy-tutorial/).

4 Conclusions

Processing Neural Network Inputs with NumPy: We use NumPy to represent the input vectors of the network as arrays. But before using NumPy, pure to better understand what is going on It's a good idea to do operations with vectors in Python.

In this first example, we have an input vector and two more weight vectors. The goal is to find which of the weights is most similar to the input direction and magnitude. If we draw them, the vectors look like this:

![Fig. 5. Three vectors in the Cartesian coordinate plane.](image)

weights_2 is more like the input vector because it points in the same direction and has a similar magnitude. So how do you determine which vectors are similar using Python?

First, we define three vectors, one for inputs and two for weights. Then we input_vector and we calculate how similar the weights_1 are. For this we use the dot product. Since all vectors are 2D vectors, the steps for this are:

1. of input_vector multiply the first index by the first index of weights_1.
2. input_vector second index of weights_2 multiply by the second index.
3. We add the results of both multiplications.

IPython console or Jupyter to follow the instructions we can use the notebook. It's good practice to create a new virtual environment every time we start a new Python project, so we
should do that first. Venv It comes with Python 3.3 and above and is convenient for creating virtual environments.

References

6. Salomov, U., Abduraxmonov, S., Urishev, O., & Juraev, N., BIO Web of Conferences 84, 05028 (2024)