Artificial Intelligence and Knowledge EngineeringList 5: Neural networks

Przemysław Dolata, Julita Bielaniewicz

May 2023

1 Purpose of the list

The aim of the exercise is to familiarize yourself with neural networks - one of the models most commonly used in machine learning today. By completing the tasks, you will learn how to apply neural networks to a practical regression problem. You will learn the basics of how this model works and examine the impact of its most important hyperparameters, as well as learn to recognize its typicalphenomena causing a decrease in the quality of prediction.

2 Introduction

The term "neural networks" covers an extremely wide class of systems for information processing, the common feature of which is the structure: they consist of*neurons*, usually organized in*layers*connected with each other by assigned connections*weights*. In this exercise, you will focus on the MLP model.*Multilayer Perceptron*in Polish literature*multilayer perceptron*). It is a one-way neural network with one input layer, one or two hidden layers, and one output layer. The size of the input layer is defined by the dimension of the data vector$X$, similarly the size of the output layer depends on the dimension of the output vector Y; the selection of the number and size of hidden layers is the task of the person designing the network.

The MLP model, like many other neural networks, can be taught by backpropagation in supervised mode, and this is the approach you will use in this assignment. The training database will be composed of a certain set of texts with labels; the goal of the learning process will be to find a text mapping function$X$ to label Y , in other words allowing for automatic labeling of future onestexts.

When selecting neural networks for a specific problem, the key is not only the selection of network architecture and*hyperparameters*describing its structure and learning process, but also the cost function that will be optimized during learning. The choice of this function is usually defined by the type of problem to be solved and the available data. From this point of view, the problems are most often dividedon:

- classification,where the projection is sought$f(\theta, X)Y$ ~~wh~~ere Y is the nominal variable; nominal variables are not ordered, examples are e.g. animal species,vehicle brand,

- regression where the projection is sought $f(\theta, X) \rightarrow Y : Y \subseteq \mathbb{R}$, that is Y is an ordinal variable, and therefore subject to ordering, e.g. age, length.

In classification problems, as a rule, the logistic function is used (*logistic regression*), while in regression problems one of the most frequently used cost functions is the mean squared error. In both variants, the learning process consists in minimizing the cost function. The value of this function is therefore the most important signal to watch when training the model to evaluate progressprocess.

Training neural models using the simple or stochastic gradient method is an iterative process. It is executed in each iteration*forward propagation*, i.e. the model is queried on a sample from the training set, then the value of the error function is calculated, which finally allows to perform*back propagation*(eng.*backpropagation*), i.e. calculationsgradient values for each ∇_θ neuron in the network using*chain rule*. Then the weights of the neurons are updated according to the formula

$$\theta_{t+1} = \theta_t - \eta \nabla$$

Whereη is a called hyperparameter*learning rate*(eng.*learning rate*). However, there are more complicated optimization methods that update the weights in a more sophisticated way. Performing this procedure for all samples in the training set is called*era*(eng.*epoch*); as a rule, training a neural model requires repetitionprocess bymany eras.

MLP class networks, like many other learning algorithms, require data to have a vector representation. Therefore, if the data does not have such a structure, it is crucial to convert them to such a form. In general, this process is called*feature extraction*, and in the context of natural language processing applications, it is often referred to as*tokenization*, i.e. breaking the text into a sequence of uniquely marked "tokens" (e.g. single words or syllables). Since the process of full tokenization is quite complicated, for the purposes of this task you will carry out a simpler variant of text data vectorization based on the so-called dictionary. The procedure will consist in determining the set$d$ the most interesting words and giving them unique indexes; then any text from the database will be converted to a vector$x \in \mathbb{N}^d$, containing on the item$x_i$and the number of occurrences of the word with the index$i$. Remember to at all stagesworking with the network - learning, testing or querying on new data - perform initial data processing in the same way(e.g. make vectorization based on the same dictionary).

3 Tasks

As part of this list, you will tackle the problem of predicting the level of ridiculousness of a given joke from the Jester set. This collection contains about 25,000 funny ratings of 100 different jokes that cover the range$(-10.0, 10.0)$.

To simplify the path and allow you to focus on examining the neural network itself, the feature extraction module is attached to this manual. Your task will be to teach a simple MLP neural network and test key hyperparameters. Use the knowledge gained during the implementation of the previous list regarding the division of data into training and validation sets. Use the Weka package or Python capabilities (mainly sklearnand matplotlib).

It's important to start your experiments with as raw a neural network as possible. If you're using python and the sklearn package, specifically _ classessklearn.neural network.MLPRegressor,make sure you pass the following hyperparameter values:

- solver = 'sgd'
- alpha = 0.0
- learning_rate = 'constant'

For Weka and moduleMultilayer Perceptronset:

- learningRate = 0.001
- momentum = 0.9
- ...and make sure that the column containing the data labels is in numeric format - in theotherwise the network will start in classification mode!

Of course, you don't have to strictly stick to the given values - see: list of tasks, point 6.

Detailed task list:

1. Prepare a training and validation set using the code attached to the listfeature extraction procedures. If you intend to use Weki, it is recommended to perform a one-time data conversion and export to one of the compatible formats. (5 points)
2. Test the operation of the basic MLP model with the default configuration of hyperparameters by training it on data from the Jester set. Trace the behavior of the model over time by visualizing the value of the cost function as a function of the number of epochs, paying attention to the values for the training setand validation set. (20 points)
3. Investigate the effect of learning rate (*learning rate*) on the achieved results: repeat learning for 3 different parameter values. Adjust the length of the learning process (number of epochs) if necessary. Plot the results as in the previous exercise. What happens when the pacelearning is too low? What if too high? (30 points)
4. Investigate the effect of the size of the MLP model on the quality of performance: perform at least 3 experiments for models that differ in the number of neurons. When does the model stop fitting the data well? When does it begin to overfit the training set? (30 points)
5. Choose the best model obtained from the above experiments and test it in practice: find (or write your own) joke text, convert it to a vector using the feature extraction method used in the tasks, and then query the neural model.Does the prediction match your expectations? (15 points)
6. ⋆ Test any other parameter, such as regularization. What is its effect? What problem does it help to fight? How does it affect the results? (bonus up to 10 points)

The result of your work should be the implementation of experiments in Python or Wece, and a report containing the results of these experiments in the form of tables and/or graphs, along with a short summary. Be sure to include the full set of hyperparameters used for each experiment. Send the report to the teacher at least 24 hours before handing over the list.

4 Literature

1. Lectures by prof. Piasecki (Lecture 8)
2. Jester collection
3. Tadeusiewicz R., M. Szaleniec -*Lexicon of neural networks*
4. scikit-klearn user guide, "Neural network models (supervised)"