Vanillatech ML Workstation

System Manual

Contents

1 Overview

Vanillatech ML serves a neuro-bio inspired generic deep learning algorithm as an API.

1.1 General information and architecture

The core program runs on Microsoft Windows and serves a REST api on tcp port 81. Upon successful connection the server expects a JSON encoded string to the base URL:

POST request
http://localhost:81/query

GET request
http://localhost:81/query?query={JSON}

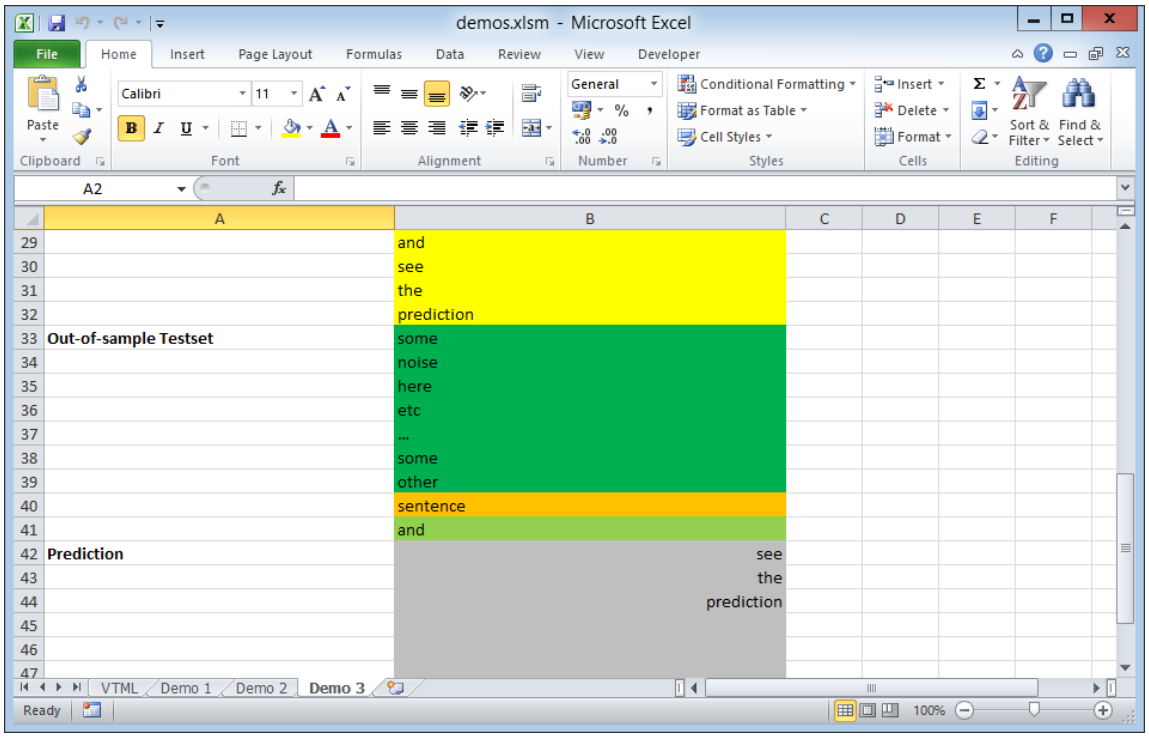
1.2 Setup

- Unzip the archive (vtml.zip)
- Run setup.exe and follow the instructions
- Run the program (VTML.exe), a tray icon titled “ML” will appear.

The server should now be ready to serve requests.

1.3 System test

In order to test the server open the spreadsheet demos.xlsm. Be sure to enable macros. Click on one of the demos in the spreadsheet. The spreadsheet will open an api connection to the server via a macro and train/test a model.



2 Parameters

The program expects a JSON encoded string as model input and will return a list of comma separated values as predictions.

MLstructure{

```
activationThreshold   number
                      default: 0.55
                      example: 0.55

                      Threshold for neurons in the model to fire.

learnmode             boolean
                      default: false

                      If set to false model is read only. If set to true model is in
                      training mode.

activateNewNeurons    boolean
                      default: true

                      If set to true model will learn patterns faster on multiple layers
                      which might lead to overfitting.

outputLMT             integer

                      Defines how many time steps the system is allowed to perform.

featureMatrix         [

                      Additional input feature matrix. Features do not interfere with each
                      other.

                      string]
inputFeature          [

                      Main input feature. Can consist of multiple string values which then
                      represent a spacial pattern.

                      string]
temporalPatternLength integer
                      default: 1

                      Maximum length of dendrites in a model.

}
```

3 Examples

3.1 “Hello world”

The following curl commands train the system with the textual pattern “hello world” three times.

```
curl "http://localhost:81/query?query=\{token:'hello1',temporary:true,learnmode:true,inputFeature:\['hello'\],outputLMT:0\}"
curl "http://localhost:81/query?query=\{token:'hello1',temporary:true,learnmode:true,inputFeature:\['world'\],outputLMT:20\}"
curl "http://localhost:81/query?query=\{token:'hello1',temporary:true,learnmode:true,inputFeature:\['hello'\],outputLMT:0\}"
curl "http://localhost:81/query?query=\{token:'hello1',temporary:true,learnmode:true,inputFeature:\['world'\],outputLMT:20\}"
curl "http://localhost:81/query?query=\{token:'hello1',temporary:true,learnmode:true,inputFeature:\['hello'\],outputLMT:0\}"
curl "http://localhost:81/query ?query=\{token:'hello1',temporary:true,learnmode:true,inputFeature:\['world'\],outputLMT:20\}"
```

The system is now able to recall and complete the pattern by testing “hello” afterwards.

```
curl "http://localhost:81/query?query=\{token:'hello1',temporary:true,learnmode:false,inputFeature:\['hello'\],outputLMT:20\}"
```

The system will output “world”.