Kocaeli Üniversitesi Bilgisayar Mühendisliği(İÖ)
Yazılım Laboratuvarı 2 2.Proje Raporu

Hazırlayanlar

Adı:Uğurcan Adı:Hazal

Soyadı:Ergün Soyadı:Türkmen

No:090202003 No:090202036

Projenin

Konusu: Tic-Tac-Toe Oyunun Makine Öğrenmesi Yöntemleriyle Gerçeklenmesi

Platformu: GNU/Linux (Xubuntu 11.10)

Yazıldığı Dil: Python 2.7.x / Qt 4.7.x

Yazıldığı Geliştirme Ortamı: Vim 7.3

Danışmanı: Yrd. Doç. Dr. Sevinç İlhan

İçindekiler

1-Problem Analizi ve Çözüm Algoritmasının Oluşturulması

Bu projede tic-tac-toe oyununun bilgisayara karşı oynanması gerekmektedir. Ayrıca bilgisayarın yapacağı hamlelerin koşul ifadeleriyle değil matematiksel ifadelerle gerçeklenmesi istenmektedir. Ve çözüm yolu olarak makine öğrenmesinde kullanılan algoritmalardan biri olan LMS (Least Mean Squares) önerilmiştir. Bu algoritmada oyun tahtası için belirli koşullar tanımlanır. Mesela bir satır,sütün,köşegende sadece bir adet x veya o bulunması gibi. Bu tanımlanan koşullar bir x değişkeni ve bir indis ile gösterilirler. En sonunda bu değerler bir b vektörü ile ifade edilir. Bir adet de oyun sonu değişkeni Vtrain(b) değişkeni tanımlanır. Bu değişkene bilgisayarın oyunu kazanıp kazanmadığını belirten bir değer atanır. Ayrıca yapılacak hamlelerin belirlenebilmesi için Xn +1 tane ağırlık değeri ve bir öğrenme fonksyonu tanımlanır.

Öğrenme fonksiyonu V(b)

V(b)= w0+w1*x1+w2*x2+....+ wn* xn şeklinde tanımlanır. Bu öğrenme fonksiyonundan dönen değer ile ağırlık değerleri güncellenir.
Ağırlık güncelleme fonksiyonu şöyle tanımlanır.

Wn = wn+ η *(Vtrain(b)-V(b)) * xn

Oyun yazılırken bu algoritma verilen bir hamle veritabanına uygulanır. Buradan oyunda kullanılacak ağırlık değerleri elde edilir. Ondan sonra bilgisayar olası bütün hamleleri analiz ederek her olası hamle için V(b) değerini hesaplar ve en büyük V(b) değerinin hesaplandığı hamleyi yapar.

2- Yazılım Mimarisi

Programlama Dili olarak Python tercih ettim. Bunun tercihte Python dilinin string işlemlerinin ve matematiksel kütüphanelerinin gelişmişliği, iki nesne arasında anahtar değer ilişkisi kurmayı sağlayan özel sözlük yapısı ve Python dilinin girintilemeye dayanan farklı yazım yapısı etkin rol oynadı. Grafik arayüz olarak Nokia tarafından geliştirilen açık kaynaklı Qt uygulama çatısını tercih ettim.
Sınıf-metod açıklamaları şöyledir.
Sınıflar :
Yazlab – Uygulama burada gerçeklenir.
Ui_Yazlab – Grafik arayüz oluşturma kodunu içerir.
Metodlar:
generate(String s) – Oyun tahtasını analiz ederek x değerlerini oluşturur.

Vb() - V(b) değerini hesaplar

lms() - V(b) değerini hesaplar ve bu değerlere göre ağırlık değerlerini günceller.

computer() - Çeşitli hesaplamaları yapıp bilgisayarın yapacağı hamleye karar verir

check() - Oyunun bitip bitmediğini belirler

buttonPressed() - Kullanıcının yaptığı hamleyi oyuna yansıtır.

Clear() - Yeni bir oyun başlatır.

Kaynak Kod

Yazlab sınıfı

```
#!/usr/bin/python
# -*- coding: utf-8 -*-
import sys
import time
from PyQt4 import QtCore
from PyQt4 import QtGui
from ui_yazlab import Ui_MainWindow #pyuic'le donusturulmus ui dosyası import edilir

class MainWindow(QtGui.QMainWindow, Ui_MainWindow):
    def __init__(self):
        QtGui.QMainWindow.__init__(self)
        self.setupUi(self)
        self.count = 1 # Oyun Sayısı
        self.ksi = 0.1 # Ogrenme katsayisi
        self.finished = 0 # Oyun devam ediyor mu bitti mi
        self.game =
{8:self.pushButton_10,0:self.pushButton_2,1:self.pushButton_3,2:self.pushButton_4,3:self.pushButt
on_5,4:self.pushButton_6,5:self.pushButton_7,6:self.pushButton_8,7:self.pushButton_9}
        #Buttonlara indislerle erisebilmek için tanımlanmış sözlük
        self.resultSet = {"positive":200,"negative":-100} # Negative - Positive donusturmesi
        self.board = ["b" for i in range(9)] # Oyun tahtasını belirten dizi
        self.w = [0.5 for i in range(9)]

        for i in self.game: # butun buttonlar aynı fonksyona atanır
            self.game[i].clicked.connect(self.buttonPressed)

        self.label_2.setText("%d.Oyun" %(self.count))
        self.pushButton.clicked.connect(self.clear)
        self.sfile = open("tic-tac-toe.data","a+")
        self.lines = self.sfile.readlines() # Dosyanin her satiri lines dizisinin bir elemanı

        for line in self.lines: # datasette her şey virgülle ayrılıyormus onu yanlıs hatırlayınca attığım
string taklaları
            line = line.replace(",","")
            self.result = self.resultSet[line[line.__len__()-9:line.__len__()-1]]
            line = line[:line.__len__()-9]
            self.generate(line)
            self.lms()
            #print ( "tahta = %s sonuc = %.2f V(b) = %.2f  \n" %(line,self.result,self.vb))
            #print ( "x1 = %.2f x2 = %.2f x3 = %.2f x4 = %.2f x5 = %.2f x6 = %.2f \n" %
(self.x[1],self.x[2],self.x[3],self.x[4],self.x[5],self.x[6]))
            #print ( "w1 = %.2f w2 = %.2f w3 = %.2f w4 = %.2f w5 = %.2f w6 = %.2f \n" %
(self.w[1],self.w[2],self.w[3],self.w[4],self.w[5],self.w[6]))

    def generate(self,s): # X'leri tespit eden fonksyon
        self.x = [0.0 for i in range(9)]
        directions = [s[:3],s[3:6],s[6:9],s[::3],s[1::3],s[2::3],s[::4],s[2:8:2]] # Tahtayı
satır,sütün,diagonal hallere dönüştürüp diziye atadım
        for d in directions:
```

```
            cx = d.count("x")
            co = d.count("o")
            cb = d.count("b")
            if ("xxb" in d  or "bxx" in d):
                self.x[1]+=1
            if ("oob" in d or "boo" in d):
                self.x[2]+=1
            if (cx == 1 and cb == 2):
                self.x[3]+=1
            if (co == 1 and cb == 2):
                self.x[4]+=1
            if (cx == 3):
                self.x[5]+=1
            if (co == 3):
                self.x[6]+=1
            if ("xbx" in d):
                self.x[7]+=1
            if ("obo" in d):
                self.x[8]+=1


    def Vb(self): # Vb'yi hesaplayan fonksyon
        self.vb = self.w[0] + self.w[1] * self.x[1] +  self.w[2] * self.x[2] + self.w[3] * self.x[3] +
self.w[4] * self.x[4] + self.w[5] * self.x[5] + self.w[6] * self.x[6] # + self.w[7] * self.x[7] +
self.w[8] * self.x[8]

    def lms(self): # W'leri güncelleyen fonksyon
        self.Vb()
        self.w[1] = self.w[1] + self.ksi * (self.result - self.vb) * self.x[1]
        self.w[2] = self.w[2] + self.ksi * (self.result - self.vb) * self.x[2]
        self.w[3] = self.w[3] + self.ksi * (self.result - self.vb) * self.x[3]
        self.w[4] = self.w[4] + self.ksi * (self.result - self.vb) * self.x[4]
        self.w[5] = self.w[5] + self.ksi * (self.result - self.vb) * self.x[5]
        self.w[6] = self.w[6] + self.ksi * (self.result - self.vb) * self.x[6]
        #self.w[7] = self.w[7] + self.ksi * (self.result - self.vb) * self.x[7]
        #self.w[8] = self.w[8] + self.ksi * (self.result - self.vb) * self.x[8]

    def computer(self): #Bilgisayarın hamlelerini hesaplayan fonksyon
        max_vb = -200
        max_index = -1
        for i in range(18):
            if (self.board[i%9] == "b"): # Tahtanın bos olan her elemani icin oraya hamle yapılsa vbler
ne olur hesaplayan if kosulu
                would = [x for x in self.board]
                self.result = -100 # oyunun kazanılmadığı her durumda Vtrain(b) - 100 dur diye
dusundum
                if (i/9 < 1):
                    would[i%9] = "o"
                    self.generate("".join(would))
                    self.Vb()
                    self.vb*=-1
                else:
```

```
                would[i%9] = "x"
                self.generate("".join(would))
                self.Vb()
            print ("Vb = %.2f max_vb = %.2f index = %d max_index = %d " %
(self.vb,max_vb,i,max_index))
            print ( "x1 = %.2f x2 = %.2f x3 = %.2f x4 = %.2f x5 = %.2f x6 = %.2f x7 = %.2f x8 =
%.2f \n" %(self.x[1],self.x[2],self.x[3],self.x[4],self.x[5],self.x[6],self.x[7],self.x[8]))
            # Bu printleri linux konsolunda dosyaya yazdırıp rapor oluşturmak için kullanıyorum
            if (self.vb > max_vb):
                max_vb = self.vb
                max_index = i % 9
        if (max_index != -1):
            self.board[max_index] = "x"
            self.game[max_index].setText("X")
            self.check()
            t = "".join(self.board)
            print ("%s\n%s\n%s" %(t[:3],t[3:6],t[6:9]))

    def check(self): # Oyun bitti mi kim kazandı belirleyen fonksyon
        self.generate("".join(self.board))
        t = "".join(self.board)
        if (self.x[5] > 0):
            self.finished = 1
            self.label.setText("Oyunu X kazandi")
            self.result = 200
            print("%d.Oyunu X kazandi" %(self.count))
            print ("%s\n%s\n%s" %(t[:3],t[3:6],t[6:9]))
        if (self.x[6] > 0):
            self.finished = 2
            self.label.setText("Oyunu O kazandi")
            self.result = -100
            print("%d.Oyunu O kazandi" %(self.count))
            print ("%s\n%s\n%s" %(t[:3],t[3:6],t[6:9]))


    def buttonPressed(self): # Hamle yapmak için bir yere tıklandığında
        button = self.sender() # Hangi buttondan tıklandıysa o nesneyi button değişkenine atar
        temp = [key for key,value in self.game.iteritems() if value.objectName() ==
button.objectName()][0]
        # Nesne isminden dizi indisine ulasilir
        if (self.board[temp] == "b" and self.finished == 0): # O nokta bos ve oyun bitmemis iken
            self.board[temp] = "o"
            button.setText("O") # Kullanıcı hamlesini yapar
            self.check()
            if (self.finished == 0):
                self.computer() # Oyun bitmemisse bilgisayar hamlesini yapar

    def clear(self): # Tahtayı sıfırlayıp yeni oyun baslatır w'lerle ellesmediği için program öğrenip
sonraki oyunlarda daha düzgün hamleler yapabiliyor.
        self.count+=1
        self.finished = 0
        self.label.setText("")
```

```
        temp = [key for key,value in self.resultSet.iteritems() if value == self.result][0]
        endgame = ",".join(self.board)+","+temp+"\n"
        #if (endgame not in self.lines):
        #    self.sfile.write(endgame)
        print("----------------------------\n")
        self.label_2.setText("%d.Oyun" %(self.count))
        self.board = ["b" for x in range(9)]
        for i in self.game:
            self.game[i].setText("")

app = QtGui.QApplication(sys.argv)
run = MainWindow()
run.show()
sys.exit(app.exec_())
```

Ui_yazlab Sınıfı

```
# -*- coding: utf-8 -*-

# Form implementation generated from reading ui file 'selam.ui'
#
# Created: Tue May  8 13:06:49 2012
#      by: PyQt4 UI code generator 4.8.5
#
# WARNING! All changes made in this file will be lost!

from PyQt4 import QtCore, QtGui

try:
    _fromUtf8 = QtCore.QString.fromUtf8
except AttributeError:
    _fromUtf8 = lambda s: s

class Ui_MainWindow(object):
    def setupUi(self, MainWindow):
        MainWindow.setObjectName(_fromUtf8("MainWindow"))
        MainWindow.resize(789, 600)
        MainWindow.setWindowTitle(QtGui.QApplication.translate("MainWindow", "Tic-Tac-Toe",
None, QtGui.QApplication.UnicodeUTF8))
        self.centralwidget = QtGui.QWidget(MainWindow)
        self.centralwidget.setObjectName(_fromUtf8("centralwidget"))
        self.pushButton = QtGui.QPushButton(self.centralwidget)
        self.pushButton.setGeometry(QtCore.QRect(680, 10, 88, 27))
        self.pushButton.setText(QtGui.QApplication.translate("MainWindow", "Yeni Oyun", None,
QtGui.QApplication.UnicodeUTF8))
        self.pushButton.setObjectName(_fromUtf8("pushButton"))
        self.pushButton_2 = QtGui.QPushButton(self.centralwidget)
        self.pushButton_2.setGeometry(QtCore.QRect(10, 50, 251, 161))
        font = QtGui.QFont()
        font.setFamily(_fromUtf8("Arial"))
        font.setPointSize(72)
        self.pushButton_2.setFont(font)
        self.pushButton_2.setFocusPolicy(QtCore.Qt.NoFocus)
        self.pushButton_2.setStyleSheet(_fromUtf8("background-color: rgb(255, 255, 255);"))
```

```
self.pushButton_2.setText(_fromUtf8(""))
self.pushButton_2.setObjectName(_fromUtf8("pushButton_2"))
self.pushButton_3 = QtGui.QPushButton(self.centralwidget)
self.pushButton_3.setGeometry(QtCore.QRect(270, 50, 251, 161))
font = QtGui.QFont()
font.setFamily(_fromUtf8("Arial"))
font.setPointSize(72)
self.pushButton_3.setFont(font)
self.pushButton_3.setFocusPolicy(QtCore.Qt.NoFocus)
self.pushButton_3.setStyleSheet(_fromUtf8("background-color: rgb(255, 255, 255);"))
self.pushButton_3.setText(_fromUtf8(""))
self.pushButton_3.setObjectName(_fromUtf8("pushButton_3"))
self.pushButton_4 = QtGui.QPushButton(self.centralwidget)
self.pushButton_4.setGeometry(QtCore.QRect(530, 50, 251, 161))
font = QtGui.QFont()
font.setFamily(_fromUtf8("Arial"))
font.setPointSize(72)
self.pushButton_4.setFont(font)
self.pushButton_4.setFocusPolicy(QtCore.Qt.NoFocus)
self.pushButton_4.setStyleSheet(_fromUtf8("background-color: rgb(255, 255, 255);"))
self.pushButton_4.setText(_fromUtf8(""))
self.pushButton_4.setObjectName(_fromUtf8("pushButton_4"))
self.pushButton_5 = QtGui.QPushButton(self.centralwidget)
self.pushButton_5.setGeometry(QtCore.QRect(10, 220, 251, 161))
font = QtGui.QFont()
font.setFamily(_fromUtf8("Arial"))
font.setPointSize(72)
self.pushButton_5.setFont(font)
self.pushButton_5.setFocusPolicy(QtCore.Qt.NoFocus)
self.pushButton_5.setStyleSheet(_fromUtf8("background-color: rgb(255, 255, 255);"))
self.pushButton_5.setText(_fromUtf8(""))
self.pushButton_5.setObjectName(_fromUtf8("pushButton_5"))
self.pushButton_6 = QtGui.QPushButton(self.centralwidget)
self.pushButton_6.setGeometry(QtCore.QRect(270, 220, 251, 161))
font = QtGui.QFont()
font.setFamily(_fromUtf8("Arial"))
font.setPointSize(72)
self.pushButton_6.setFont(font)
self.pushButton_6.setFocusPolicy(QtCore.Qt.NoFocus)
self.pushButton_6.setStyleSheet(_fromUtf8("background-color: rgb(255, 255, 255);"))
self.pushButton_6.setText(_fromUtf8(""))
self.pushButton_6.setObjectName(_fromUtf8("pushButton_6"))
self.pushButton_7 = QtGui.QPushButton(self.centralwidget)
self.pushButton_7.setGeometry(QtCore.QRect(530, 220, 251, 161))
font = QtGui.QFont()
font.setFamily(_fromUtf8("Arial"))
font.setPointSize(72)
self.pushButton_7.setFont(font)
self.pushButton_7.setFocusPolicy(QtCore.Qt.NoFocus)
self.pushButton_7.setStyleSheet(_fromUtf8("background-color: rgb(255, 255, 255);"))
self.pushButton_7.setText(_fromUtf8(""))
self.pushButton_7.setObjectName(_fromUtf8("pushButton_7"))
```

```
        self.pushButton_8 = QtGui.QPushButton(self.centralwidget)
        self.pushButton_8.setGeometry(QtCore.QRect(10, 390, 251, 161))
        font = QtGui.QFont()
        font.setFamily(_fromUtf8("Arial"))
        font.setPointSize(72)
        self.pushButton_8.setFont(font)
        self.pushButton_8.setFocusPolicy(QtCore.Qt.NoFocus)
        self.pushButton_8.setStyleSheet(_fromUtf8("background-color: rgb(255, 255, 255);"))
        self.pushButton_8.setText(_fromUtf8(""))
        self.pushButton_8.setObjectName(_fromUtf8("pushButton_8"))
        self.pushButton_9 = QtGui.QPushButton(self.centralwidget)
        self.pushButton_9.setGeometry(QtCore.QRect(270, 390, 251, 161))
        font = QtGui.QFont()
        font.setFamily(_fromUtf8("Arial"))
        font.setPointSize(72)
        self.pushButton_9.setFont(font)
        self.pushButton_9.setFocusPolicy(QtCore.Qt.NoFocus)
        self.pushButton_9.setStyleSheet(_fromUtf8("background-color: rgb(255, 255, 255);"))
        self.pushButton_9.setText(_fromUtf8(""))
        self.pushButton_9.setObjectName(_fromUtf8("pushButton_9"))
        self.pushButton_10 = QtGui.QPushButton(self.centralwidget)
        self.pushButton_10.setGeometry(QtCore.QRect(530, 390, 251, 161))
        font = QtGui.QFont()
        font.setFamily(_fromUtf8("Arial"))
        font.setPointSize(72)
        self.pushButton_10.setFont(font)
        self.pushButton_10.setFocusPolicy(QtCore.Qt.NoFocus)
        self.pushButton_10.setStyleSheet(_fromUtf8("background-color: rgb(255, 255, 255);"))
        self.pushButton_10.setText(_fromUtf8(""))
        self.pushButton_10.setObjectName(_fromUtf8("pushButton_10"))
        self.label = QtGui.QLabel(self.centralwidget)
        self.label.setGeometry(QtCore.QRect(10, 10, 471, 34))
        font = QtGui.QFont()
        font.setFamily(_fromUtf8("Arial"))
        font.setPointSize(22)
        self.label.setFont(font)
        self.label.setText(_fromUtf8(""))
        self.label.setObjectName(_fromUtf8("label"))
        self.label_2 = QtGui.QLabel(self.centralwidget)
        self.label_2.setGeometry(QtCore.QRect(480, 10, 171, 34))
        font = QtGui.QFont()
        font.setFamily(_fromUtf8("Arial"))
        font.setPointSize(22)
        self.label_2.setFont(font)
        self.label_2.setText(_fromUtf8(""))
        self.label_2.setAlignment(QtCore.Qt.AlignRight|QtCore.Qt.AlignTrailing|
QtCore.Qt.AlignVCenter)
        self.label_2.setObjectName(_fromUtf8("label_2"))
        MainWindow.setCentralWidget(self.centralwidget)
        self.menubar = QtGui.QMenuBar(MainWindow)
        self.menubar.setGeometry(QtCore.QRect(0, 0, 789, 25))
        self.menubar.setObjectName(_fromUtf8("menubar"))
```

```
        MainWindow.setMenuBar(self.menubar)
        self.statusbar = QtGui.QStatusBar(MainWindow)
        self.statusbar.setObjectName(_fromUtf8("statusbar"))
        MainWindow.setStatusBar(self.statusbar)

        self.retranslateUi(MainWindow)
        QtCore.QMetaObject.connectSlotsByName(MainWindow)

    def retranslateUi(self, MainWindow):
        pass
```

3-Yazılımın Çalıştırılması ve Özellikleri

Oyunu Microsoft Windows üzerinde çalıştırabilmek için http://www.python.org/download/ adresinden Python 2.7 sürümünü ve http://www.riverbankcomputing.co.uk/software/pyqt/download adresinden PyQt-Py2.7 sürümünü işlemci mimarinize uygun olarak indirmeniz gerekmektedir. Bu işlemin ardından yazlab.py dosyasına çift tıklayarak yazılımı çalıştırabilirsiniz.

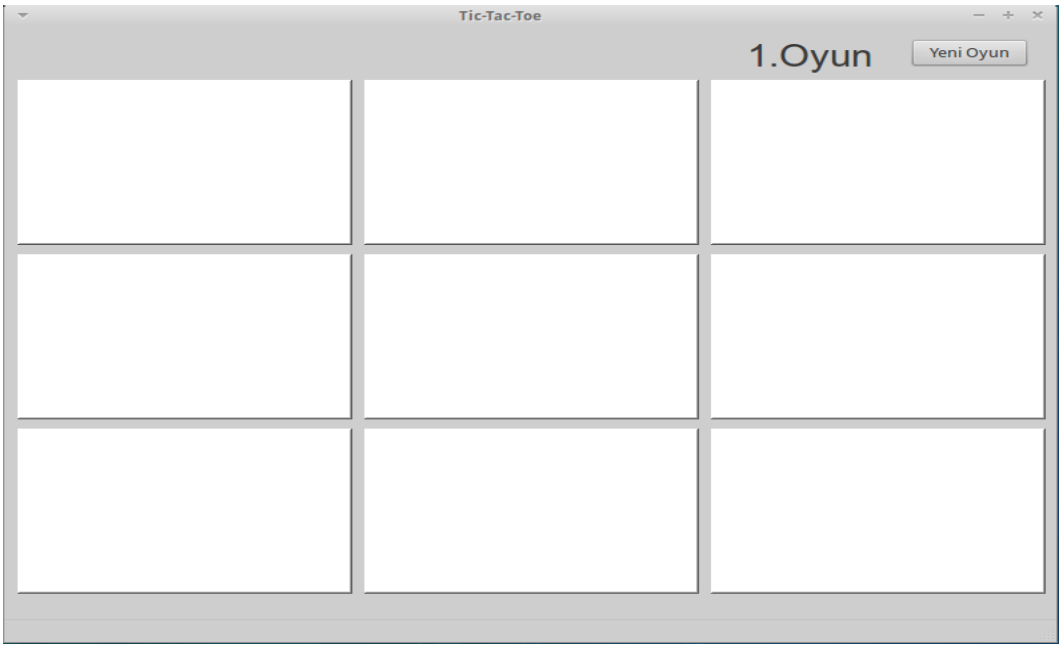
Oyunu bir GNU/Linux dağıtımında çalıştırmak için terminalden yazlab.py'nin olduğu klasöre giriniz ve orada

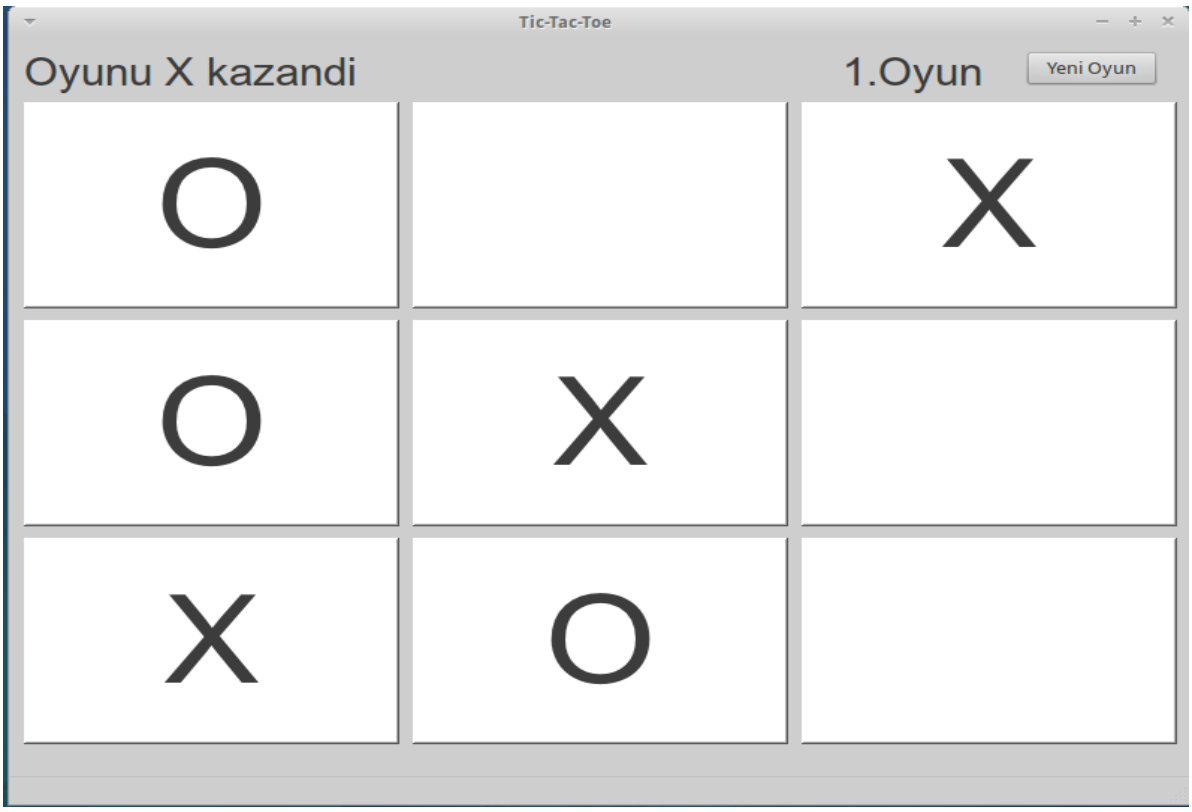
python yazlab.py

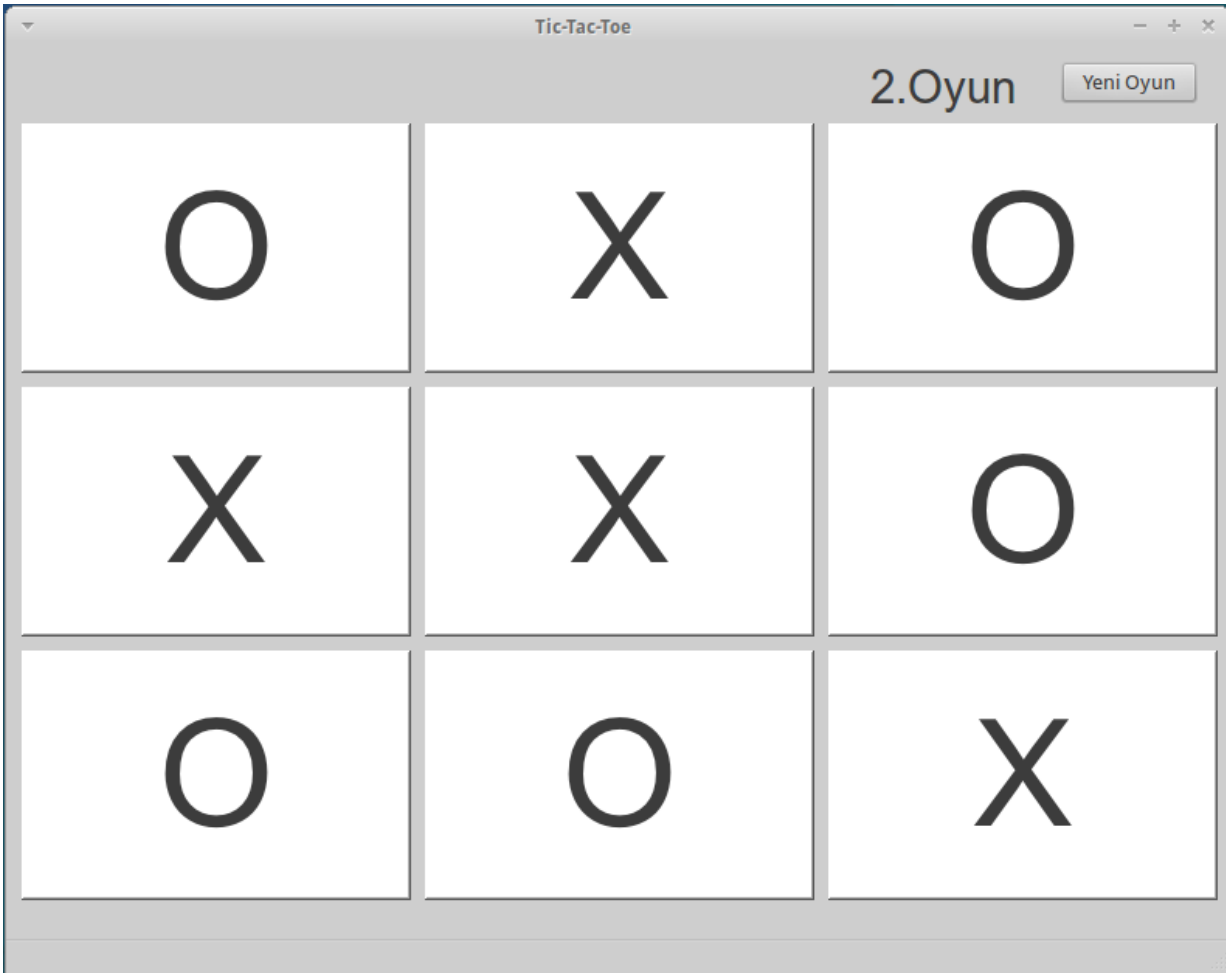
komutunu yazmanız gerekmektedir. Oyun hem hamle ve her oyun için v(b) ve x değerlerini ekrana basmaktadır bunu bir dosyaya debug çıktısı olarak verme özelliği de vardır bunun için tekrar terminale

python yazlab.py > dosya.txt

yazmanız gerekmektedir. Oyun biri kazandığı zaman sol üst kısma bir mesaj yazdırıp daha fazla hamleye izin vermemektedir. Yeni Oyun düğmesine tıklayarak yeni bir oyun başlatabilirsiniz. Oyuna ait ekran görüntüleri




Tic-Tac-Toe
Oyunu X kazandi
1.Oyun
Yeni Oyun
O
X
O
X
X
O


Tic-Tac-Toe
2.Oyun
Yeni Oyun
O
X
O
X
X
O
O
O
X

4-Yazılımın Sonuçları ve Eleştirisi

Oyunu 100 defadan fazla oynayarak test ettim. Ancak hiç birinde bilgisayarı yenmeyi başaramadım. En iyi halde berabere kalabildim. Bu sonuçtan hareketle programın kendinden istenen şeyi sorunsuz bir şekilde gerçekleştirebildiği hükmüne vardım.

5- Kaynakça

http://docs.python.org/
http://wiki.python.org/moin/
http://www.istihza.com/
http://doc.qt.nokia.com/4.7/index.html
PYTHON,Fırat ÖZGÜL, Kodlab Yayınları