Introduction to AI

February 6, 2019

Introduction examples

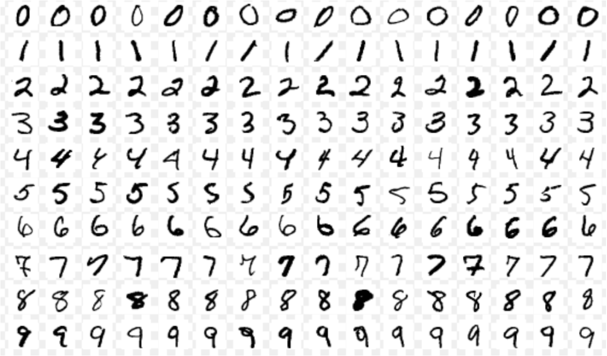


Figure: MNIST database [LeCun and Cortes, 2010]

Introduction examples

- Boston Dynamics robot (video)
- `https://www.youtube.com/watch?v=LikxFZZO2sk`
- `https://www.youtube.com/watch?v=g0TaYhjpOfo`

Introduction examples

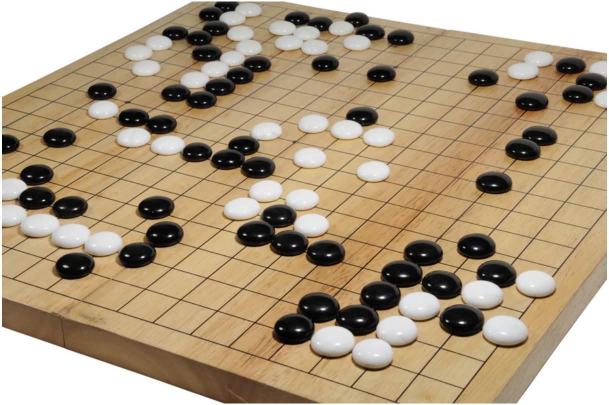


Figure: Go game, beaten by AlphaGo in 2017 [Silver et al., 2016]

Introduction examples

-
  - Boston Dynamics robot
  - MINST classification
  - AlphaGo
- All do different things but are gathered under the term "AI".

Introduction

- People doing "AI" can actually come from rather different scientific communities. Which ones according to you ?

Introduction

- People doing "AI" can actually come from rather different scientific communities:
  - Statistics

Introduction

- People doing "AI" can actually come from rather different scientific communities:
  - Statistics
  - Optimization

Introduction

- People doing "AI" can actually come from rather different scientific communities:
  - Statistics
  - Optimization
  - Other mathematical fields :

Introduction

- People doing "AI" can actually come from rather different scientific communities:
  - Statistics
  - Optimization
  - Other mathematical fields : graph theory, combinatorics, information theory

Introduction

- People doing "AI" can actually come from rather different scientific communities:
  - Statistics
  - Optimization
  - Other mathematical fields : graph theory, combinatorics
  - Statistical physics

Introduction

- People doing "AI" can actually come from rather different scientific communities:
  - Statistics
  - Optimization
  - Other mathematical fields : graph theory, combinatorics
  - Statistical physics
  - Robotics

Introduction

- People doing "AI" can actually come from rather different scientific communities:
  - Statistics
  - Optimization
  - Other mathematical fields : graph theory, combinatorics
  - Statistical physics
  - Robotics
  - Cognitive sciences / neuroscience / psychology

Introduction

- People doing "AI" can actually come from rather different scientific communities:
    - Statistics
    - Optimization
    - Other mathematic fields : graph theory, combinatorics
    - Statistical physics
    - Robotics
    - Cognitive sciences / neuroscience / psychology
- For instance, *data science* is mostly a mix between statistics, optimization, graph theory

Introduction

- It seems to be rather varied
- We will focus on *Machine Learning*, a term slightly more specific than "AI"
- In Machine Learning, some parameters are learned in an *automatic way* in order to solve a problem or to optimize a solution

Introduction

- It seems to be rather varied
- We will focus on *Machine Learning*, a term slightly more specific than "AI"
- In Machine Learning, some parameters are learned in an *automatic way* in order to solve a problem or to optimize a solution
- AI is not a recent research topic, it started with computer science. Machine Learning has been trendy for some years now because of some good performance on **some specific problems**

Introduction

In the three first examples, according to you which one is NOT a Machine Learning system ?

Introduction

- Alpha Go : Machine Learning (Reinforcement Learning)

Introduction

- Alpha Go : Machine Learning (Reinforcement Learning)
- MNIST : Machine Learning (Supervised Learning)

Introduction

- Alpha Go : Machine Learning (Reinforcement Learning)
- MNIST : Machine Learning (Supervised Learning)
- Boston Dynamics : no Machine Learning (just plain robotics)

Overview

The problem

- For a certain input x, you want to predict an output y
- To do so, you learn from a number of **labeled examples** (x_i, y_i)
- In the case where what you want to predict is a **class**, it is a **classification problem**
- In the case where what you want to predict is a general function $y = f(x)$, it is a **regression problem**

The problem

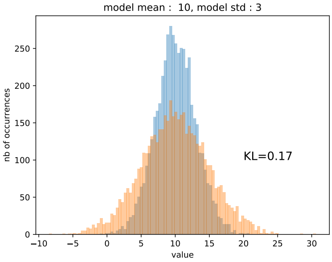
- For a certain input x, you want to predict an output y
- To do so, you learn from a number of **labeled examples** (x_i, y_i)
- In the case where what you want to predict is a **class**, it is a **classification problem**
- In the case where what you want to predict is a general function $y = f(x)$, it is a **regression problem**
- Example : MNIST (classification)
- Question : how do you choose and constrain your function f ?

The problem

- From a large number of samples x_i, you want to retrieve information on their **structure**
- For instance you want to learn a **distribution**, or a **clustering** of your data.
- Examples : social networks

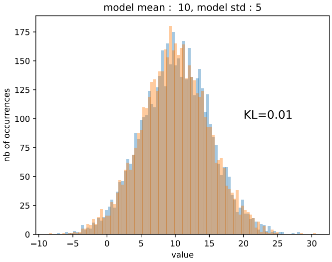
The problem

- From a large number of samples x_i, you want to retrieve information on their **structure**
- For instance you want to learn a **distribution**, or a **clustering** of your data.



The problem

- From a large number of samples x_i, you want to retrieve information on their **structure**
- For instance you want to learn a **distribution**, or a **clustering** of your data.



The problem

- From a large number of samples x_i, you want to retrieve information on their **structure**
- For instance you want to learn a **distribution**, or a **clustering** of your data.
- Question : how do you constrain your distribution to fit your data ?

The problem

- A **more general paradigm** that describes an **agent** in a **world**.
- The standard formalization was the one proposed by Richard Sutton [Sutton and Barto, 2016]
- At each time, the world is in a state s. An agent perfoms an **action** a according to a **policy** π. When performing an action, the agent receives an **reward** r.

The problem

- A **more general paradigm** that describes an **agent** in a **world**.
- The standard formalization was the one proposed by Richard Sutton [Sutton and Barto, 2016]

The problem

- A **more general paradigm** that describes an **agent** in a **world**.
- The standard formalization was the one proposed by Richard Sutton [Sutton and Barto, 2016]
- Example : a Chessplayer, AlphaGo, a game AI, automatic vacuum cleaner

The problem

- At each time, the world is in a state s. An agent perfoms an **action** a according to a **policy** π. When performing an action, the agent receives an **reward** r.
- The agent wants to learn an **optimal policy**, meaning the policy that maximises its reward.

This paradigm has many variants

- State s, action a, policy π, reward r.
- Is the policy **deterministic** ? Is it **stochastic** ?
- Does the agent have a **model** of its environment ?
- How many steps ahead whould the agent look ?

This paradigm has many variants

- State s, action a, policy π, reward r.
- Is the policy **deterministic** ? Is it **stochastic** ?
- Does the agent have a **model** of its environment ?
- How many steps ahead should the agent look ?
- All these conditions change the way the problem should be addressed and solved. The **Bellman equations** rule the updates of the optimal policy.

Example problem

- Typical Machine Learning situation : should I explore my environment more or exploit what I have learnt so far ?
- Concept of ϵ-greedy policy

Final remark

- These paradigms can be mixed

Final remark

- These paradigms wan be mixed
- Mosty, this means that
  - unsupervised learning can be used in a supervised learning problem (semi supervised learning)
  - unsupervised learning and supervised learning can be used in a reinforcement learning problem

Some famous methods

Let's look at some classical methods in ML

Linear separator

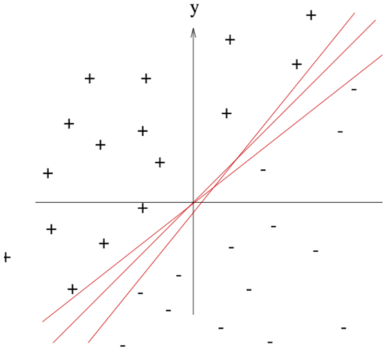
- Very favorable supervised learning case
- 

Figure: Linearly separable problem (image : wikipedia)

Linear separator

- Unfortunately, most problems are *not linearly separable*

K means clustering

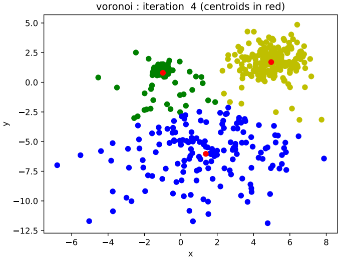
- A famous unsupervised clustering method
- 

Figure: K means clustering

Kmeans

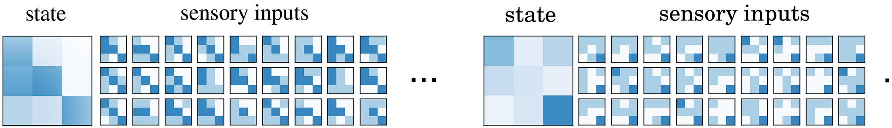


Figure: Other example of kmeans clustering, this time in 9 dimensions [Le Hir et al., 2018]

Kmeans : Esperance Maximisation algorithm

- Classical Machine Learning algorithm (EM)
- Blackboard
- What could be the drawbacks of this algorithm ?

Exercise 1 : Kmeans clustering

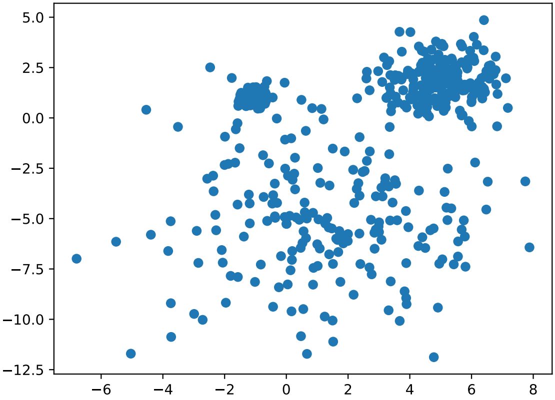


Figure: Data we want to cluster

Kmeans clustering

cd kmeans

- Modify the **kmeans_ex.py** file so that it performs the kmeans algorithm.
- There are **two mistake series** between lines 50 and line 82, you will need to fix them.

You should obtain something like this:

Kmeans

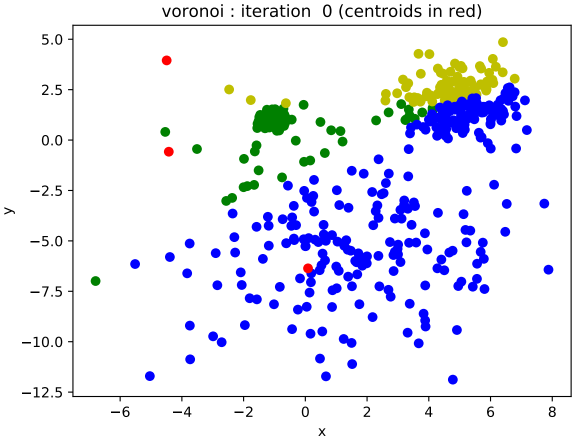


Figure: Voronoi 0th iteration

Kmeans

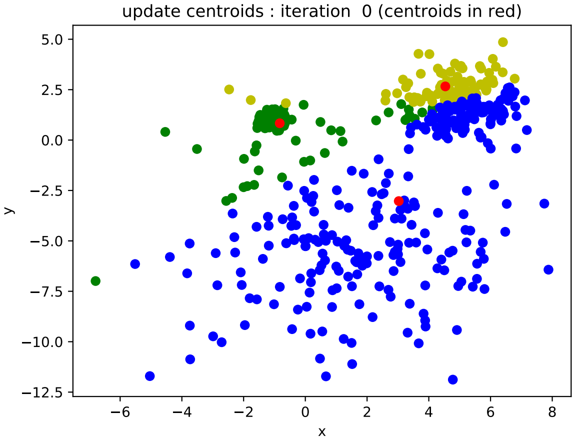


Figure: Centroids 0th iteration

Kmeans

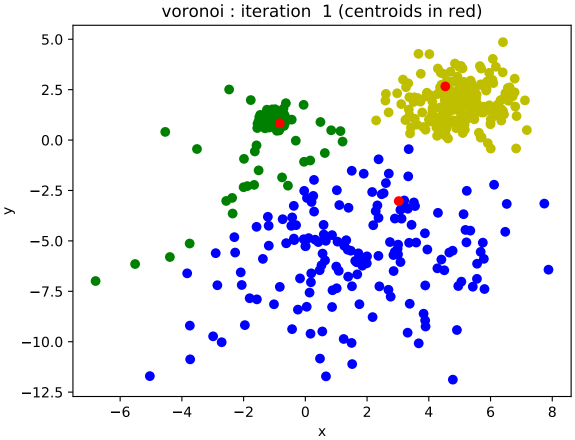


Figure: Voronoi 1st iteration

Kmeans

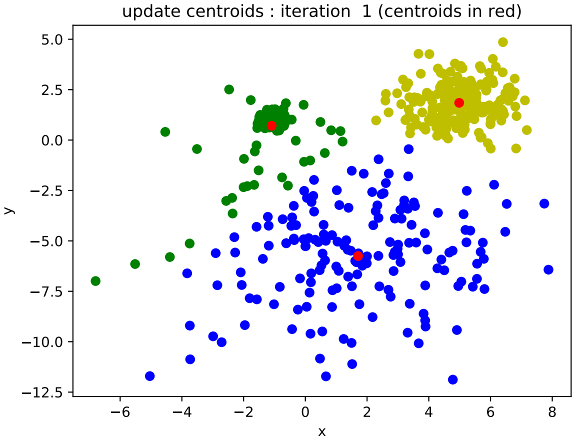


Figure: Centroids 1st iteration

Kmeans : Esperance Maximisation algorithm

- What would you do if the algorithm falls in a local optimum ?

Neural network

- A **neuron** is a simple elementary function
- A neural network a more complex function built with several neurons

Neural networks

- A **neuron** is a simple elementary function
- A neural network a more complex function built with several neurons
- A **Deep Neural Network** is a big neural networks ont more than two stacked layers of neurons

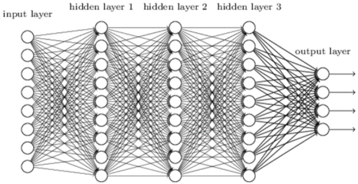


Figure: A deep neural network : source
`https://datawarrior.wordpress.com/2017/10/31/interpretability-of-neural-networks/`

AlexNet

- **AlexNet** [Krizhevsky et al., 2012] is an example of Deep Neural Network : famous for a good performance at the ImageNet recognition challenge. Is is a **Convolutional Neural Network**

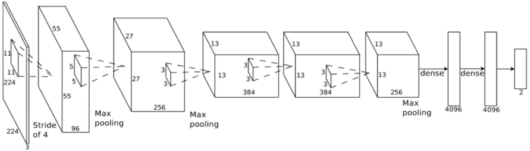
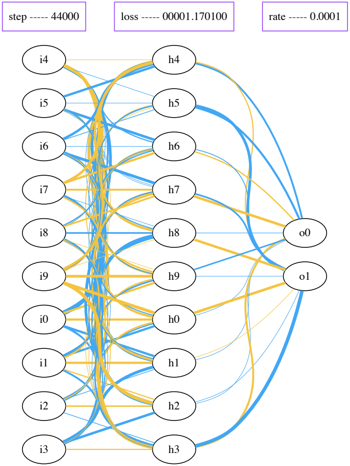


Figure: AlexNet

Neural Networks

We will do exercises on neural networks tomorrow.



Spectral Clustering

- How can you cluster data if you do not have a **distance** between them ?

Spectral Clustering

- How can you cluster data if you do not have a **distance** between them ?
- A **Similarity** is a more general notion that allows you to compare data
- It can be used in unsupervised learning contexts [Le Hir et al., 2018]

Research and problems in AI

What makes AI a hard problem ?

Curse of dimensionality

- First aspect : algorithmic complexity
- The objects considered are in high dimensional spaces, and in high number
- Even "simple" situations grow very complex (Atari games)

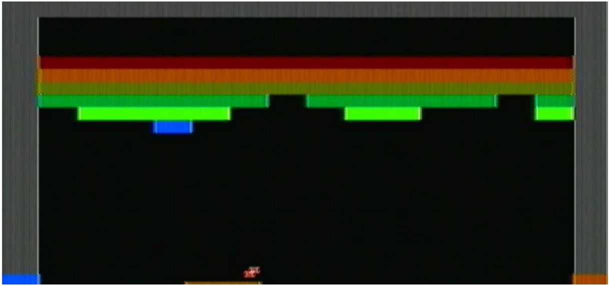


Figure: One Atari game

Curse of dimensionality

- Even "simple" situations grow very complex (Atari games)

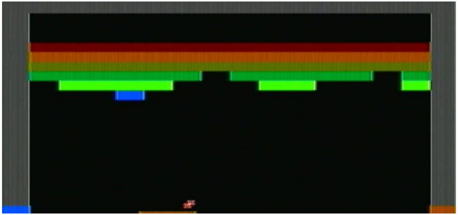


Figure: One Atari game

- Especially true for reinforcement learning
- If it is algorithmically hard to solve an Atari game, how hard would a real world problem be ?

Non convexity, non linearity

- We try to optimize crazy functions : in extremely high dimensional spaces, with crazy shapes. (blackboard)
- So the power of the mathematical tools si limited and **experimentation** is needed.
- So there lacks **grounding** to the results

Overfitting

- What could be the drawbacks of using a very simple model (very few parameters) ?

Overfitting

- What could be the drawbacks of using a very simple model (very few parameters)
  - Weak expressive power

Overfitting

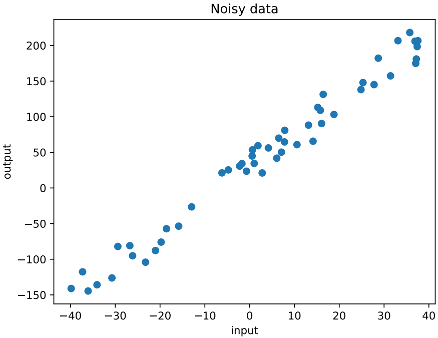
- What could be the drawbacks of using a very simple model (very few parameters)
  - Weak expressive power
- What could be the drawbacks of having a very complex model (that contains a very large number of parameters, e.g. millions as in a very deep neural network) ie a very high expressive power ?

Overfitting

- What could be the drawbacks of using a very simple model (very few parameters)
  - Weak expressive power
- What could be the drawbacks of having a very complex model (that contains a very large number of parameters, e.g. millions as in a very deep neural network) ie a very high expressive power ?
  - Harder to optimize
  - Harder to interpret
  - Can **overfit**

Exercise 2

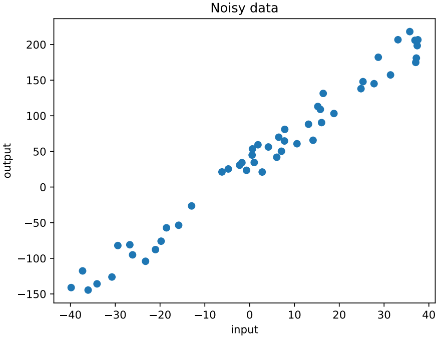
- We will learn a model of the following data :



- We want to perform supervised learning in order to be able to predict the output y for a new sample x.

Exercise 2

- We want to perform supervised learning in order to be able to predict the output y for a new sample x.



- To illustrate the problem of overfitting, we will use **polynoms** as models.

Exercise 2

- **cd overfitting**. Use the dataset contained in **linear_noisy_data.csv**, load it from **fit_data_ex.py** in order to assess the impact of the **degree** of the polynom on overfitting.
- You just need to use the functions written in the file (no modification needed)

Exercise 2

- The higher the degree of the polynom, the more parameters it has and the better it can fit the training points :

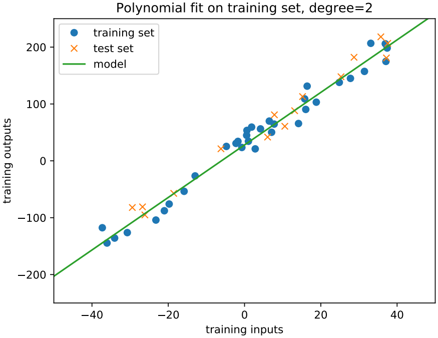


Figure: degree 2

Exercise 2

- The higher the degree of the polynom, the more parameters it has and the better it can fit the training points :

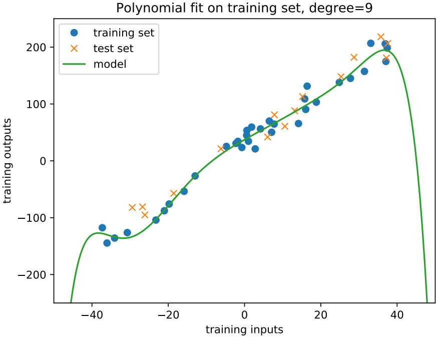


Figure: degree 9

Exercise 2

- The higher the degree of the polynom, the more parameters it has and the better it can fit the training points :

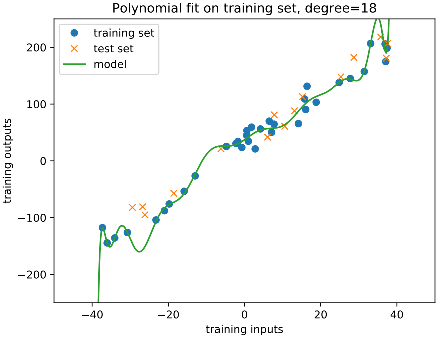


Figure: degree 19

Exercise 2

- However, the error on the test set increases and the model looses **signification**

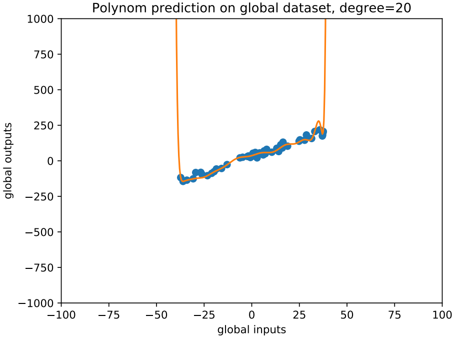


Figure: Useless solution

Exercise 2

- In that case the best solution should be a very simple polynom (degree 1)

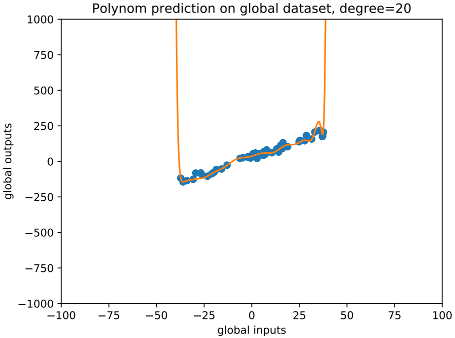


Figure: Useless solution

Trying to prevent overfit

- The problem of overfitting is linked to that of **generalisation** : to what extent are we allowed to extrapolate the knowledge obtained on the training samples to new samples ?
- To improve generalisation, one can use :
  - a **validation set**
  - **regularization**

Regularization methods

- Penalize the magnitude of the weight in a neural network
- Remove neurons in a neural network (pruning)
- use smooth functions (continuous)

Deep learning

- Deep learning is powerful for some situations but is subject to the above shortcomings
- Some researchers try to have a better understanding of their behavior. Some famous ones are Yoshua Bengio (Montral), Geoffrey Hinton (Toronto), Stphane Mallat (Paris)

The End

References

Krizhevsky, A., Sutskever, I., and Hinton, G. E. (2012).
ImageNet Classification with Deep Convolutional Neural Networks.
Advances In Neural Information Processing Systems, pages 1–9.

Le Hir, N., Sigaud, O., and Laflaquière, A. (2018).
Identification of Invariant Sensorimotor Structures as a Prerequisite for the Discovery of Objects.
Frontiers in Robotics and AI, 5(June):1–14.

LeCun, Y. and Cortes, C. (2010).
MNIST handwritten digit database.

Silver, D., Huang, A., Maddison, C. J., Guez, A., Sifre, L., van den Driessche, G., Schrittwieser, J., Antonoglou, I., Panneershelvam, V., Lanctot, M., Dieleman, S., Grewe, D.,