236606 - Deep Learning
Tutorial 1

Yonatan Geifman    Izik Golan
Ran El-Yaniv

Computer Science Department
Technion - Israel Institute of Technology

Spring 2018
Table of Contents

1 Administration

2 Linear Algebra
   • Basic Definitions
   • Norms
   • Angles and Projections
Table of Contents

1 Administration

2 Linear Algebra
   • Basic Definitions
   • Norms
   • Angles and Projections
Course Staff

- Lecturer
  - Ran El-Yaniv: rani@cs.technion.ac.il

- Teaching Assistants
  - Yonatan Geifman: yonatan.g@cs.technion.ac.il
  - Izik Golan: izikgo@cs.technion.ac.il
Course Website

- The course website can be found at
  - https://webcourse.cs.technion.ac.il/236606/
  - Contains all information about the course, such as: announcements, course staff, syllabus, office hours, contact info, course material, homework, grades, etc.

- Contact
  - We have opened a dedicated email address for this course. All queries should be addressed to this address.
  - deep.learning@cs.technion.ac.il
Homework

- There will be 5 homework assignments. Most of them wet, but some will contain dry sections.
- All assignments are mandatory, and will comprise 40% of the final grade.
- Each assignment will have identical weight.
- Submission in pairs only.
- There will be no late submission, except for reserve duty.
Final Project

There will be a final project in pairs, comprising 60% of the final grade.

You will have to solve a challenging task we will define, using the tools you’ve acquired during the course.

At the end of the semester, we will hold a competition between the projects based on their performance on the task. The 1st, 2nd, and 3rd places will receive a bonus of 8, 5, and 3 point to their final grade, respectively.

The deadline for submitting the project is 31/07/2018. However, teams who submit the project after the semester ends will not be able to participate in the contest and earn the bonus.
The wet assignments will be coded in python 3.6.

The first 2 assignments will use the scientific computing package NumPy. The remaining will mainly use the deep learning framework Keras.

For the deep learning assignments and final project, you will be given access to a machine with GPU, pre-installed with all relevant packages.
Python Environment

- We highly recommend using **PyCharm** as your IDE. A professional license can be obtained using your Technion email address.

- In order to use NumPy (especially for Windows users), we recommend working with the **Anaconda** distribution, which already comes with a compiled version of NumPy alongside the Python interpreter, and many more useful packages and features.
Table of Contents

1 Administration

2 Linear Algebra
   - Basic Definitions
   - Norms
   - Angles and Projections
Linear Algebra Reminder

A good understanding of linear algebra is essential for understanding and working with many machine learning algorithms, especially deep learning algorithms. Hence, we will review the basic concepts necessary for this course.
** Scalars, Vectors, Matrices and Tensors 

- **Scalar**
  - A single number. Usually denoted by a lowercase letter, e.g. $v \in \mathbb{R}$

- **Vector**
  - An array of numbers. Usually denoted by a lowercase bold letter, e.g. $\mathbf{v} \in \mathbb{R}^n$ for a vector which is an array of $n$ real numbers.
Scalars, Vectors, Matrices and Tensors

**Matrix**
- A 2-D array of numbers, so each element is identified by two indices instead of just one. Usually denoted by an uppercase bold letter, e.g. \( \mathbf{A} \in \mathbb{R}^{m \times n} \) for a matrix with \( m \) rows and \( n \) columns of real numbers.

**Tensor**
- An \( n \)-D array of numbers, so each element is identified by \( n \) indices. Usually denoted by an uppercase bold letter with no italic, e.g. \( \mathbf{A} \in \mathbb{R}^{m_1 \times m_2 \times \cdots \times m_n} \) for an \( n \) dimensional tensor of real numbers.
Matrix Operations

- **Transpose**
  - \((A^\top)_{i,j} = A_{j,i}\)

- **Trace**
  - \(\text{Tr}(A) = \sum_i A_{i,i}\). Sum of diagonal entries.

- **Addition**
  - Element-wise addition. Matrices must be the same shape.

- **Multiplication**
  - \(C = AB \Rightarrow (C)_{i,j} = \sum_k A_{i,k}B_{k,j}\)
  - Defined only if the number of columns in \(A\) is equal to the number of rows in \(B\).

- **Hadamard Product**
  - \(C = A \odot B\)
  - Element-wise multiplication. Matrices must be the same shape.
Norm Definition

Norms are a way to measure the magnitude of a vector. A real-valued function \( f \) is considered a norm if it satisfies the following conditions:

- \( f (\mathbf{x}) = 0 \iff \mathbf{x} = 0 \)
- \( f (\mathbf{x} + \mathbf{y}) \leq f (\mathbf{x}) + f (\mathbf{y}) \) (Triangle inequality)
- \( \forall \alpha \in \mathbb{R}, f (\alpha \mathbf{x}) = |\alpha| f (\mathbf{x}) \) (Homogeneity)
The $L^p$ Norm

A commonly used norm, defined by:

$$\|x\|_p = \left(\sum_i |x_i|^p\right)^{\frac{1}{p}}$$

where $p \in \mathbb{R}, p \geq 1$
The $L^p$ Norm - Examples

- $\|x\|_1 = \sum_i |x_i|$
  Used when we care about discriminating between zero and almost-zero elements.

- $\|x\|_2 = \sqrt{\sum_i x_i^2}$
  The **Euclidean** norm - distance from origin. We often use the squared form: $x^\top x$

- $\|x\|_\infty = \max_i |x_i|$
  The **Max** norm.

- $\|x\|_0$
  Number of non-zero elements in $x$. This is not actually a norm (why?).

- $\|A\|_F = \sqrt{\sum_{i,j} A_{i,j}^2} = \sqrt{\text{Tr} (A^\top A)}$
  A norm on Matrices named **Frobenius** norm. Identical to $L^2$ norm where the matrix is flattened to a vector.
In order to define an angle between two vectors, we need the notion of **Inner Product**. For our purposes, we will just say that $\langle x, y \rangle = x^\top y = x \cdot y$ defines an inner product.

We can construct a norm from an inner product by

$$\|x\| = \sqrt{\langle x, x \rangle}$$

The **angle** between two vectors is defined as

$$\cos (\angle (x, y)) = \frac{\langle x, y \rangle}{\|x\| \|y\|}$$

Two vectors are called **orthogonal** if the cosine of the angle between them is 0 (= their inner product is 0).
Projections

- Formally, a **projection** is defined as a linear transformation $P$ from a vector space to itself, such that $P = P^2$.

- An **orthogonal projection** is a projection for which the range $U$ and the null space $V$ are orthogonal sub-spaces, or

\[
\forall x, y : \langle Px, y - Py \rangle = \langle x - Px, Py \rangle = 0
\]
Vector Projections

A vector projection of vector $x$ on a non-zero vector $y$ is the orthogonal projection of $x$ on a line parallel to $y$. It can be calculated by

$$x \parallel y = \frac{\langle x, y \rangle}{\langle y, y \rangle}y = \|x\| \cos (\angle (x, y)) \frac{y}{\|y\|} = \frac{x^\top y}{\|y\|^2}y$$

Projection of $a$ on $b$ ($a_1$)
Distance of Point to Hyperplane

In machine learning, we sometimes deal with some \( n \)-dimensional Euclidean space (\( \mathbb{R}^n \)), and the predictions are based on some hyperplane defined by

\[
H_{w,b} = \{ x \in \mathbb{R}^n : w^\top x + b = 0 \}.
\]

Given a vector \( x_0 \), how do we find its distance from the hyperplane?
Distance of Point to Hyperplane
Distance of Point to Hyperplane
Distance of Point to Hyperplane