RIPS 2018 Readings

Table of Contents

Machine Learning

Machine learning is a subset of artificial intelligence in the field of computer science that often uses statistical techniques to give computers the ability to "learn" (i.e., progressively improve performance on a specific task) with data, without being explicitly programmed.

General

Articles, Blogposts, and Tutorials

- Awesome Machine Learning
- A Neural Network in 11 lines of Python (Part 1)
- A Neural Network in 13 lines of Python (Part 2 - Gradient Descent)

Naive Bayes

Naive Bayes is a simple technique for constructing classifiers: models that assign class labels to problem instances, represented as vectors of feature values, where the class labels are drawn from some finite set.

Articles, Blogposts, and Tutorials

- 6 Easy Steps To Learn Naive Bayes Algorithm
- Wikipedia Page

Convolutional Neural Networks

Convolutional neural network (CNN, or ConvNet) is a class of deep, feed-forward artificial neural networks, most commonly applied to analyzing visual imagery. Convolutional networks were inspired by biological processes in that the connectivity pattern between neurons resembles the organization of the animal visual cortex.

Articles, Blogposts, and Tutorials

- Graph Convolutional Networks
- kegra: Deep Learning on Knowledge Graphs with Keras
- Implementation of Graph Convolutional Networks in TensorFlow
- Wikipedia Page

Papers

- Convolutional Networks on Graphs for Learning Molecular Fingerprints
- Convolutional Neural Networks on Graphs with Fast Localized Spectral Filtering
- Deep Convolutional Networks on Graph-Structured Data
- Gated Graph Sequence Neural Networks
- Semi-Supervised Classification with Graph Convolutional Networks
- Spectral Networks and Locally Connected Networks on Graphs

Recurrent Neural Networks

A recurrent neural network (RNN) is a class of artificial neural network where connections between nodes form a directed graph along a sequence. This allows it to exhibit dynamic temporal behavior for a time sequence. Unlike feedforward neural networks, RNNs can use their internal state (memory) to process sequences of inputs. This makes them applicable to tasks such as unsegmented, connected handwriting recognition or speech recognition.

Articles, Blogposts, and Tutorials

- Recurrent Neural Networks Tutorial, Part 1 – Introduction to RNNs
- Recurrent Neural Networks Tutorial, Part 2 – Implementing a RNN with Python, Numpy and Theano
- Recurrent Neural Networks Tutorial, Part 3 – Backpropagation Through Time and Vanishing Gradients
- Recurrent Neural Networks Tutorial, Part 4 – Implementing a GRU/LSTM RNN with Python and Theano
- The Unreasonable Effectiveness of Recurrent Neural Networks
- Wikipedia Page

Long-Storm Term Memory Recurrent Neural Networks

Long short-term memory (LSTM) units (or blocks) are a building unit for layers of a recurrent neural network (RNN). A common LSTM unit is composed of a cell, an input gate, an output gate and a forget gate. The cell is responsible for "remembering" values over arbitrary time intervals; hence the word "memory" in LSTM. Each of the three gates can be thought of as a "conventional" artificial neuron, as in a multi-layer (or feedforward) neural network: that is, they compute an activation (using an activation function) of a weighted sum. Intuitively, they can be thought as regulators of the flow of values that goes through the connections of the LSTM; hence the denotation "gate". There are connections between these gates and the cell.

Articles, Blogposts, and Tutorials

- Anyone Can Learn To Code an LSTM-RNN in Python (Part 1: RNN)
- Understanding LSTM Networks

Other Sources

- Andrej Karpathy Blog
- i am trask
- WildML

Other Sources

Natural Language Processing and Information Extraction

Natural language processing (NLP) is an area of computer science and artificial intelligence concerned with the interactions between computers and human (natural) languages, in particular how to program computers to process and analyze large amounts of natural language data. Challenges in natural language processing frequently involve speech recognition, natural language understanding, and natural language generation. I recommend starting with "Text

Mining: the State of the Art and the Challenges" for an overview of text mining.

General

Articles, Blogposts, and Tutorials

- edX Course on Natural Language Processing
- Oxford Deep NLP 2017 Course
- Regular Expressions 101

Papers

- Accurate Unlexicalized Parsing
- Evolving Better Stoplists for Document Clustering and Web Intelligence
- On Stopwords, Filtering and Data Sparsity for Sentiment Analysis of Twitter
- Preprocessing Techniques for Text Mining - An Overview

Information Extraction

Information extraction (IE) is the task of automatically extracting structured information from unstructured and/or semi-structured machine-readable documents.

Articles, Blogposts, and Tutorials

- OpenIE 5.0
- Reverb

Papers

- CERES: Distantly Supervised Relation Extraction from the Semi-Structured Web
- From Information to Knowledge Harvesting Entities and Relationships from Web Sources
- Identifying Relations for Open Information Extraction
- OpenIE-based approach for Knowledge Graph construction from text
- Open Information Extraction from the Web
- Open Information Extraction: The Second Generation
- Retrieval Effectiveness on the Web
- Risk Information Extraction and Aggregation
- Text Mining: The State of the Art and the Challenges

word2vec and doc2vec

Linguistic Regularities in Continuous Space Word Representations seems to have started it all. Here are the links for documentation on word2vec and doc2vec.

Articles, Blogposts, and Tutorials

- A Gentle Introduction to Doc2Vec
- Vector Representations of Words
- Word2Vec Tutorial - The Skip-Gram Model
- Word2Vec Tutorial Part 2 - Negative Sampling

Papers

- An Empirical Evaluation of doc2vec with Practical Insights into Document Embedding Generation
- Distributed Representations of Sentences and Documents
- Distributed Representations of Words and Phrases and their Compositionality
- Efficient Estimation of Word Representations in Vector Space
- Linguistic Regularities in Continuous Space Word Representations
- Neural Network Doc2vec in Automated Sentiment Analysis for Short Informal Texts

Semantic Web

The Semantic Web can be thought of as a “web of data.” The ultimate goal of the Semantic Web is to enable computers to do more useful work and to develop systems that can support trusted interactions over the network. The term “Semantic Web” refers to W3C’s vision of the Web of linked data. The trend toward the Semantic Web is based on the goal of semantic interoperability of data, which enables application independence, improved search facilities, and improved machine inference.

General

Articles, Blogposts, and Tutorials

- DaConta's *Semantic Web*
- Wikipedia's Page on Semantic Web

Knowledge Graphs

Knowledge Graphs are an increasingly popular data structure for representing relational information. They assume a knowledge base in the form of relational triples (subject, predicate, object) and model these triples using a graph an ordered pair G=(V,E) where V is a set of concept nodes and E is a set of predicate edges.

Articles, Blogposts, and Tutorials

- Automated Fact-Checking presentation by Joshua Chen

Papers

- Computational Fact Checking from Knowledge Networks
- Discriminative Predicate Path Mining for Fact Checking in Knowledge Graphs
- OpenIE-based approach for Knowledge Graph construction from text
- A Review of Relational Machine Learning for Knowledge Graphs
- Towards Computational Fact-Checking

Computational Fact-Checking

Center for Complex Networks and Systems Research

Articles, Blogposts, and Tutorials

- Automated Fact-Checking presentation by Joshua Chen

Papers

- Anatomy of an online misinformation network
- Computational Fact Checking by Mining Knowledge Graphs
- Computational Fact Checking from Knowledge Networks
- Computational Fact Checking through Query Perturbations
- Discriminative Predicate Path Mining for Fact Checking in Knowledge Graphs
- The Epistemology of Intelligent Semantic Web Systems
- Finding Streams in Knowledge Graphs to Support Fact Checking
- Towards Computational Fact-Checking

Statistical Relational Learning

Statistical relational learning (SRL) is a subdiscipline of artificial intelligence and machine learning that is concerned with domain models that exhibit both uncertainty (which can be dealt with using statistical methods) and complex, relational structure. Note that SRL is sometimes called Relational Machine Learning (RML) in the literature. Typically, the knowledge representation formalisms developed in SRL use (a subset of) first-order logic to describe relational properties of a domain in a general manner (universal quantification) and draw upon probabilistic graphical models (such as Bayesian networks or Markov networks) to model the uncertainty; some also build upon the methods of inductive logic programming.

General

Papers

- Community Detection in Graphs
- Philosophers are Mortal: Inferring the Truth of Unseen Facts
- A Review of Relational Machine Learning for Knowledge Graphs

Managing Relational Data

Resource Description Framework

The Resource Description Framework is an XML-based language to describe resources. Please note: RDFs use URIs (Uniform Resource Identifiers) so rather than seeing something like "tree bark" in an RDF you will see something like "www.example.org/ontology/plant/tree/#bark." Additionally, there are two kinds of notations for RDFs, the subject-predicate-object notation, called **N3** notation, which is more intuitive and the **serialized** format which is all of the triples aggregated and nested in a huge mess. To give an example, consider the following three sentences:

- Buddy Belden owns a business.
- The business has a Web site accessible at http://www.c2i2.com/~budstv.
- Buddy is the father of Lynne.

The N3 or subject-predicate-object notation is:

- <#buddy> <#owns> <#business>.
- <#business> <#has-website> <http://www.c2i2.com/~budstv >.
- <#buddy> <#father-of> <#lynne>

The serialized RDF looks like this:

```
<rdf:RDF
    xmlns:RDFNsId1='#'
    xmlns:rdf='http://www.w3.org/1999/02/22-rdf-syntax-ns#'>
    <rdf:Description rdf:about='#Buddy'>
        <RDFNsId1:owns>
            <rdf:Description rdf:about='#business'>
                <RDFNsId1:has-website
rdf:resource='http://www.c2i2.com/~budstv' />
            </rdf:Description>
        </RDFNsId1:owns>
        <RDFNsId1:father-of rdf:resource='#Lynne'/>
    </rdf:Description>
</rdf:RDF>
```

- Getting Started with RDFLib
- RDF 1.1 Concepts and Abstract Syntax
- Resource Description Framework

Data Sets:

- Freebase
- Linked Open Data Cloud
- PubChem

SPARQL Protocol and RDF Query Language

- Querying with SPARQL
- SPARQL Endpoint Interface to Python
- SPARQL Protocol and RDF Query Language
- SPARQL Wrapper