応用計量分析２（第1回）

担当教員: 梶野 洸（かじの ひろし）

本日の内容

- 自己紹介
- 講義の概要紹介
    - 目的
    - 講義の進め方
    - 評価方法
    - 講義で必要な前提知識
- はじめに
    - 環境構築（Pythonのインストール）

自己紹介

- 名前: 梶野 洸（かじの ひろし）
- 現所属: IBM Research - Tokyo
    - 数理科学グループ（数学でなにかやる部署）
    - 機械学習の研究やビジネス応用を担当
        - 時系列解析やケモインフォマティクスへの応用など
- 趣味
    - 音楽
    - ボルダリング（弱い...）
    - 料理

企業研究者という職業

民間企業で研究者をやっています

- 企業研究者の主な仕事
    - 研究（論文書いたり）
    - 開発（プログラム書いたり）
    - ビジネス（お客様の課題解決に貢献したり）

- 3つのバランスは会社・人ごとに異なる
    - IBM東京基礎研究所では研究とビジネスの比重が大きい
    - 開発は、ビジネスの一環としてやる程度

A Functional Dynamic Boltzmann Machine

Hiroshi Kajino
IBM Research - Tokyo
KAJINO@jp.ibm.com

Abstract

Dynamic Boltzmann machines (DyBMs) are recently developed generative models of a time series. They are designed to learn a time series by efficient online learning algorithms, whilst taking long-term dependencies into account with help of eligibility traces, recursively updatable memory units storing descriptive statistics of all the past data. The current DyBMs assume a finite-dimensional time series and cannot be applied to a functional time series, in which the dimension goes to infinity (*e.g.*, spatiotemporal data on a continuous space). In this paper, we present a *functional dynamic Boltzmann machine (F-DyBM)* as a generative model of a functional time series. A technical challenge is to devise an online learning algorithm with which F-DyBM, consisting of functions and integrals, can learn a functional time series using only finite observations of it. We rise to the above challenge by combining a kernel-based function approximation method along with a statistical interpolation method and finally derive closed-form

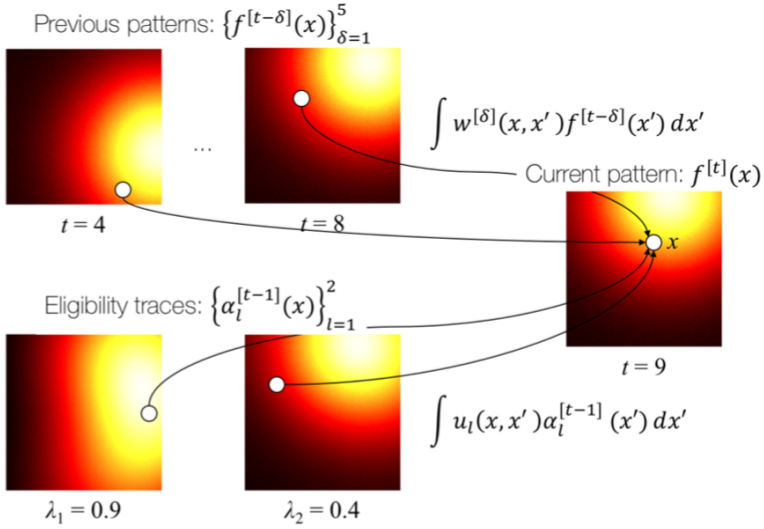


Figure 1: Illustration of F-DyBM, modeling a functional pattern $f^{[t]}(x)$ defined on a two dimensional space. Heat maps represent functional patterns. The current pattern depends on five past patterns and two eligibility traces (which summarize all the past patterns) through weight functions $w^{[\delta]}$ and u_l, respectively.

Molecular Hypergraph Grammar with Its Application to Molecular Optimization

Hiroshi Kajino [1]

Abstract

Molecular optimization aims to discover novel molecules with desirable properties, and its two fundamental challenges are: (i) it is not trivial to generate valid molecules in a controllable way due to hard chemical constraints such as the valency

problem is to obtain $m^\star \in \mathcal{M}$ such that,

$$m^\star = \arg\max_{m \in \mathcal{M}} f(m), \tag{1}$$

where $f : \mathcal{M} \to \mathbb{R}$ is an unknown function that outputs a chemical property of the input molecule to be maximized.

わたしの一日

- 8時くらいに起きる
- 10時〜17時くらいまで働く
    - 研究（論文書いたり）
    - お客様案件（データ解析〜システム実装）
    - 会議
- 帰ってごはんを作る or ボルダリング
- そのあと家で仕事したり
- たまに0時くらいに電話会議

大学の研究者との違い

企業研究所は良くも悪くもビジネスありき

- 研究からビジネスへの橋渡しができる（しなければいけない）
    - 作ったものを実際に使ってもらえる
    - 研究しているだけでは出てこない問題が出てきて面白い
- 会社の方針に左右される
    - 会社が注力したい技術に沿ったことをやったほうがいい
    - 現在は人工知能ブームなので、人工知能に関することをやる人が多い

わたしが教えられそうなこと

実際に役に立っている技術 + それを支える基礎的な理論

- 応用だけやっていても基礎が疎かになると新しい技術に対応できない
- 基礎だけやっていてもモチベーションがわかないかもしれない

講義のねらい

数理的な基礎、プログラミングともに機械学習を課題解決のツールとして身につける

- 数理的な手法を使って課題解決できることがあることを認識する
    - 統計的機械学習の基礎を身につける
    - プログラミングによる実践方法を身につける
- その限界も（なんとなく）認識する
    - すべてが人工知能で解決できるわけではない
    - すべてが数理的な手法で解決できるわけではない

講義の中身

機械学習 の中の **教師なし学習** を中心に、

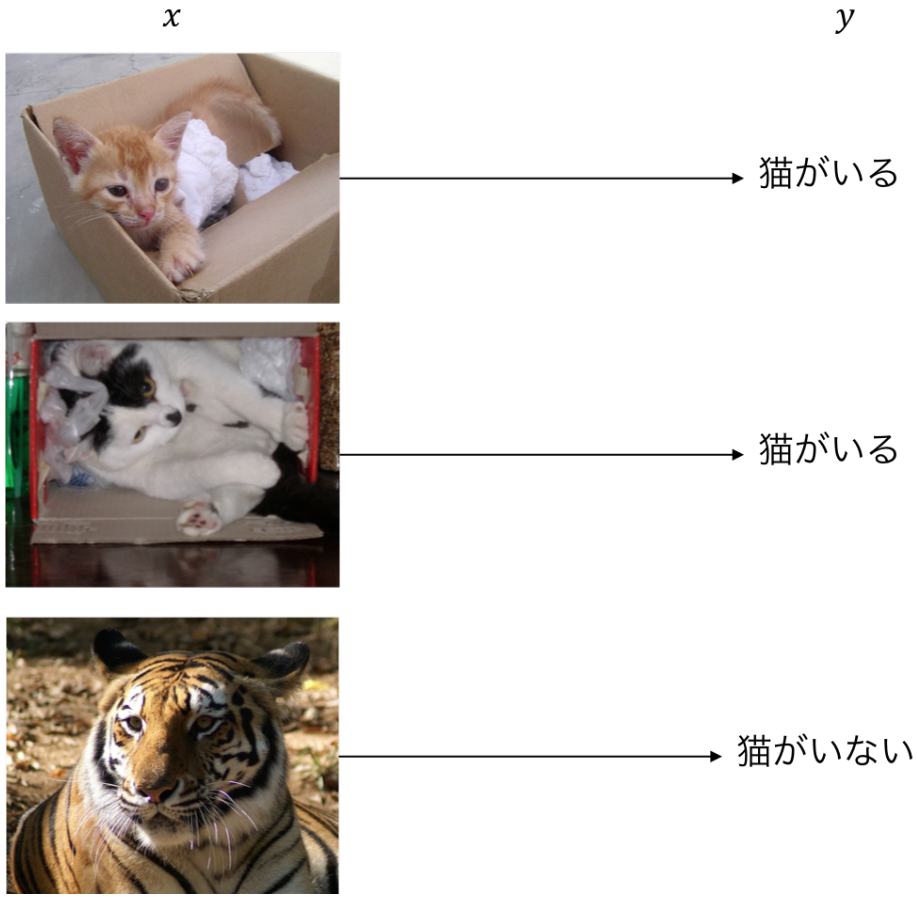
数理的な内容 から **プログラミング方法** までを取扱います。

- 機械学習？
- 教師なし学習？
- 数理？
- プログラミング？

機械学習とは

ある概念を具体例から帰納的に獲得する技術である。

- 具体例: （画像, 猫かどうか）のペア
- 概念: 画像に猫がいるかどうか



(画像は ImageNet より引用)

機械学習とは

ある概念を具体例から帰納的に獲得する技術である。

- 具体例: 動物の画像
- 概念: 動物の画像

$$x$$



(画像は ImageNet より引用)

機械学習とは

ある概念を具体例から帰納的に獲得する技術である。

- 具体例: 動物の画像
- 概念: 動物の（なんとなくの）分類

(画像は ImageNet より引用)

機械学習の分類

教師あり学習

- データ x とラベル y の間の従う規則を推定する
- 一番目の例

教師なし学習

- データ x の従う規則を推定する
- 二番目、三番目の例

教師なし学習の用途

- データセットの概要把握
    - データセットがどのようなデータから成り立っているかが把握できる
    - クラスタリング的な
- 異常検知
    - 学習した規則から外れたデータ=異常と判断できる
- データ生成
    - 画像や音声
    - 化学物質
    - Generative adversarial network を調べてみると面白いかも

講義で必要な前提知識

教養レベルの数学と簡単なプログラミングの知識を仮定します。

線形代数

行列積、逆行列、固有値、

確率・統計

条件付き確率、期待値、分散、（正規分布、最尤推定）

プログラミング

変数、for文、if文

講義の進め方

なるべく講義中に理解してもらえるようにしたいです

- 講義中に演習を交えていきます
    - 数学の確認
    - 簡単なプログラミング
- 質問用に Github を使います
    - ここ (https://github.com/kanojikajino/lecture/issues)

      で `New issue` を選ぶ（Githubへの登録が必要）
    - わからない=私の説明が悪い、と思って気軽に聞いてください
    - なるべく全員に共有したいので、授業後にこっそり聞きに来るのは望ましくないです

講義ページ

ここに講義資料をPDFで置きます

毎回持ってくるもの

- パソコン
- 筆記用具

評価方法

- 期末レポートのみの予定
    - 出席点は特に考えていないです
    - 昨年度の課題はここ (https://kanojikajino.github.io/lectures/2018/applied_analytics)を参照
- （なんとなくの）評価基準
    - ちゃんと自分でやってきたレポートに対しては優しい評価をします
    - 他人のレポート写しやウェブからの剽窃には厳しく評価します

ここまでの話で質問

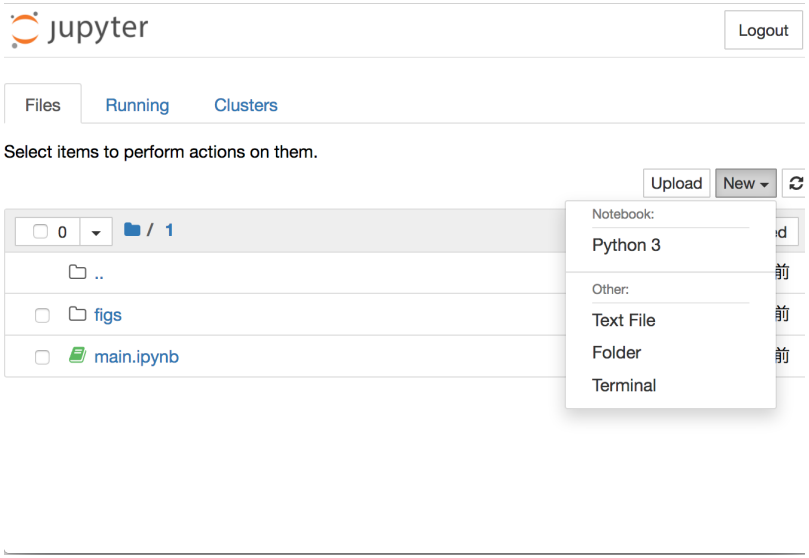
課題兼宿題

PC に Python の環境をインストールしてきてください。

- Anaconda (https://www.anaconda.com/download/ (https://www.anaconda.com/download/)) をインストールする
    - Python 3.7, 64bit を選択する。
- わからなければ「python anaconda (osの名前) インストール」とかで検索する

（以下、情報が古いかもしれないので参考まで）

1. Anaconda のインストール
2. Jupyter notebook の起動（ブラウザが開くはず）
    - Windows の場合「スタートメニュー」⇒「すべてのアプリ」⇒「Anaconda3 (64-bit)」⇒「Jupyter Notebook」
    - Mac の場合「ターミナル」を起動して `jupyter notebook` と入力して return
3. New から Python3 を開く



4. `print('Hello, world!')` と入力して return, 次のような表示になればOK

```
In [1]: print('Hello, world!')
```

```
Hello, world!
```