ch1:概论(看个眼熟之章)

对智能的三个理论:

知识理论:
认为智能行为取决于知识的数量及其一般化的程度,智能就是在巨大搜索空间中迅速找到一个满意解的能力
思维理论:
认知科学认为智能的核心是思维
进化理论:
智能是某种复杂系统所浮现的性质

对人工智能的定义:

- 研究如何使一个计算机系统具有像人一样的智能特征
- 最终目标:造出一个像人一样具有智能,会思维和行动的计算机系统
- 强人工智能:机器可以有知觉,有自我意识
- 弱人工智能:机器只不过看起来像是智能的,不会有自主意识

人工智能的发展到目前为止经历的四个阶段:孕育,形成,发展,深化
人工智能研究的特点:

- 强综合性、理论性、实践性、应用性
- 与传统的计算机软件系统相比:以知识为主要研究对象,多采用启发式,一般都允许出现不正确结果

研究途径:

- 符号主义:
  - 基于物理符号系统假设(具有必要且足够的方法实现智能),有限合理性原理设计的(人类采用了启发式搜索的试探性方法来求得问题的有限合理解)
  - 知识是信息的一种形式,是构成智能的基础。人工智能的核心问题是知识表示、知识推理和知识运用
  - 人类的认识过程就是一种符号处理过程,思维就是符号的计算
  - 物理符号系统功能主要有输入、输出、储存、复制符号；建立符号结构；条件性迁移,依赖已经掌握的符号继续完成行为
  - 执行上述功能与表现出智能是充要条件
  - 指传统的人工智能或者经典的人工智能
- 联结主义:
  - 人类智能的物质基础是神经系统
  - 其基本单元是神经元(以分布式的方式存储信息,以并行方式处理信息,强鲁棒性和容错性,有实现自组织、自学习能力)
  - 神经网络具有独特优势
  - 从结构上对人脑进行模拟(适合于模拟人脑形象思维)
  - 深度学习
- 行为主义:
  - 无需知识表示的智能和无需推理的智能,从行为上模拟和体现智能
  - 智能在与环境交互作用中才表现出来,不应采用集中式的模式,而是需要具有不同的行为模式与环境交互,以此来产生复杂行为
  - 知识的形式化表达和模型化方法是人工智能的重要障碍之一
  - 人工智能可以象人类智能一样逐步进化
  - 表明控制论、系统工程的思想将进一步影响人工智能发展

应用领域:
- 模式识别,对给定事物进行鉴别和分类
- 自然语言处理,使计算机能够理解、生成、检索自然语言(语音和文本)
- 机器学习与数据挖掘
- 人工神经网络与深度学习
- 问题求解与博弈
- 多智能体,分散的智能系统之间如何相互协调各自的智能行为以实现问题的并行求解
- 专家系统
- 计算机视觉
- 自动定理证明
- 智能控制
- 机器人学
- 人工生命

ch2:知识(recite 与混个眼熟并行)

知识的定义:

- 把有关信息关联在一起所形成的信息结构
- 反映了客观世界中事物之间的关系
- 特性:
  - 相对正确性
  - 不确定性(包括随机性,模糊性,不完全性,经验性)
  - 可表示性与可利用性
- 人工智能对知识表示方法的要求:
  - 知识表示方法要有较强的表达能力和足够的精细程度(可理解性,自然性)
  - 便于获取和表示新知识并与已有知识相连接
  - 便于搜索,能够较快地在知识库中找出有关知识
  - 便于推理,要能够从已有知识中推出需要的答案或结论

知识的分类

- 对形成:概念、命题、公理、定理、规则、方法
- 对层次:表层知识和深层知识
- 对确定性程度:确定性知识和不确定性知识
- 对等级:元知识(运用知识的准则)和非元知识(具体知识)
- 对作用:陈述性知识和过程性知识

知识的表示方法:

- 逻辑表示法:
  - 目的:能够采用数学演绎的方式,证明一个新的语句是从哪些已知的正确语句推导出来的
  - 一阶谓词逻辑表示法:属于表层知识,不易表达过程性知识和启发式知识,抛弃了表达内容中所含有的语义信息,推理难以深入
  - 产生式表示法:基本形式是蕴含式,谓词逻辑蕴含式只能表示精确知识；而产生式不仅可以表示精确知识,还可以表示不精确知识
  - 由三部分组成:规则集、黑板、控制策略
  - 推理方式分为定向推理,逆向推理
  - 规则匹配分为精确匹配(经过符号代换之后完全一致),非精确匹配
  - 冲突消解:类比规约-规约冲突
- 层次结构表示法:

  - 指框架表示法和面向对象表示法
  - 框架结构:分为框架(数据库),槽(表),侧面(列)；既可以有包含关系(表中表),也可以有继承关系(外键)
- 网格结构表示法:
  - Petri Net:
    - 可用三元组(P, T, F)来表示
    - P 表示位置集合,表示事物属性或状态
    - T 表示转换集合,表示从一种状态转变为另一种状态
    - F 表示有向弧集合,用于指明转换的方向,有向弧只能存在于 P 和 T 或者 T 和 P 之间
    - 一个位置可以拥有多个令牌用于进行并发控制
  - 语义网络:
    - 一个带标识的有向图
    - 具有结构性,联想性,直观性,非严格性,处理复杂性
- 脚本表示法:概念依赖
- 过程表示法:着重于对知识的利用

知识获取与管理:

- 知识获取分为抽取知识,转换知识,输入知识,检测知识
- 可以非自动知识获取(来源于知识工程师),也可以是自动知识获取(机器学习与数据挖掘)
- 知识管理的任务:
  - 具体地、物理地组建知识库,保存知识
  - 在知识库中安排具体的知识
  - 实现知识的增加、删除、修改、查询等功能
  - 记录知识库的变更
  - 保证知识库的安全
- 组件知识库的:
  - 知识库具有相对独立性
  - 便于对知识的搜索
  - 便于对知识进行维护及管理
  - 便于存储用多种模式表示的知识
- 本体论:
  - 面向内容
  - 本体是对某一概念化所做的一种显式的解释说明
  - 概念化是从特定目的出发对所表达的世界所进行的一种抽象的、简化的观察
  - 描述的是客观事物的存在,本体独立于对本体的描述,本体独立于个体对本体的认识,本身不存在与客观事物的误差
  - 描述的本体代表了人们对某个领域的知识的公共观念
  - 对本体的描述应该是形式化的、清晰的、无歧义的
  - 目的是实现某种程度的知识共享和重用
  - 分为顶级本体,领域本体,任务本体,应用本体
- 知识图谱:
  - 一种揭示实体之间关系的语义网络
  - 知识图谱一般用三元组来表示:
  - G=(E, R, S),G 是知识图谱
  - E 是知识库中的实体集合
  - R 是知识库中的关系集合
  - s 代表知识库中的三元组集合
  - 三元组的基本形式主要包括(实体 1,关系,实体 2),(概念,属性,属性值)

- 逻辑上可分为数据层和模式层,体系结构构建模式有自顶向下和自底向上
- 知识抽取:实体抽取(命名实体识别),关系抽取,属性抽取
- 知识融合:实体对齐(实体匹配),知识加工,知识更新
- 推理方法:基于逻辑的推理,基于图的推理

知识系统:

- 一类具有专门知识和经验的计算机系统,并通过对人类知识和问题求解过程的建模
- 知识系统以知识库和推理机为核心,把知识与系统其它部分分离开,并且知识系统强调知识而不是方法
- 特点:启发性,灵活性,交互性,实用性,易推广
- 知识工程:
  - 建造一个知识系统的过程可以称为知识工程
  - 知识工程包括以下几个方面:知识获取,知识表示,设计知识库和推理机,用适当的计算机语言实现系统。

ch3:确定性推理

命题逻辑:

命题:能判断真假的陈述句
基本等值式具有交换律,结合律,分配律,摩根律,蕴含等值式别忘了
吸收率 1:$p \wedge (p \vee q) \Leftrightarrow p$
吸收率 2:$p \vee (p \wedge q) \Leftrightarrow p$
假言易位式:$p \to q \Leftrightarrow \neg q \to \neg p$

命题逻辑归结法:
合取范式:仅由有限个简单析取式构成的合取式
子句集:合取范式的所有单个析取式
归结过程:建立命题公式,将命题写成合取范式求子句集,归结子句集,若归结式为空子句则 S 矛盾原命题满足(要求原结论取反)

谓词逻辑:

加入了存在,任意表达的一种命题
量词辖域收缩与扩张等值式:(所有式子都可把任意与存在互换)

- $\forall x(P(x) \vee Q) \Leftrightarrow \forall x P(x) \vee Q$
- $\forall x(P(x) \wedge Q) \Leftrightarrow \forall x P(x) \wedge Q$
- $\forall x(P(x) \to Q) \Leftrightarrow \exists x P(x) \to Q$,P(x)在蕴含式左端,辖域改变
- $\forall x(Q \to P(x)) \Leftrightarrow Q \to \forall x P(x)$,P(x)在蕴含式又端,辖域不变

谓词逻辑归结法:
前束范式:一切量词都位于该公式的最左边且这些量词的辖域都延伸到公式的末端
化 SKOLEM 标准型:

- 化为只有合取析取的形式
- 取反深入两次内部
- 变元易名
- 全称存在量词左移
- 消除存在,存在的左边有哪些全称变量则用其函数代替之,最后略去全称

谓词逻辑的任意公式都可以化为与之等值的前束范式,但其前束范式不唯一
谓词公式 G 的 Skolem 标准型同 G 并不等值

若 G 是给定的谓词公式,S 是相应的子句集,则:G 是不可满足的⇔ S 是不可满足的
G 真不一定 S 真,而 S 真必有 G 真。即: S ⇒ G

置换:
一个集合,里面记载了形如a/b的元素,意思是可以将这俩互换
合成:
设 $\theta=\{t_1/x_1,t_2/x_2,...,t_n/x_n\},\lambda=\{u_1/y_1,u_2/y_2,...,u_m/y_m\}$,是两个置换。 则 θ 与 λ 的合成也是一个置换,记作 $\theta\cdot\lambda$。它是从集合:

$$\{t_1\cdot\lambda/x_1,t_2\cdot\lambda/x_2,...,t_n\cdot\lambda/x_n,u_1/y_1,u_2/y_2,...,u_m/y_m\}$$

中删去以下两种元素: 当 $\boldsymbol{t_i\cdot\lambda=x_i}$ 时,删去 $\boldsymbol{t_i\cdot\lambda/x_i}$ (意思就是不能自反);
当 $y_j\in\{x_1,x_2,...,x_n\}$ 时,删去 u_j/y_j (xi 已被左侧集合访问,右侧不再考虑)。
最后剩下的元素所构成的集合。
合一:
公式集 F={F1,F2,...,Fn},一个置换θ,可使 F1θ=F2θ=...= Fnθ,一个公式集的合一可以不是唯一的

归结原理:

- 写出谓词表达式
- 化为 Skolem 标准形求取子句集 S 并对 S 中可归结的子句做归结
- 把待求解的问题也用谓词公式表示出来,然后把它否定并与谓词 Answer(变元必须与问题公式的变元完全一致)构成析取式
- 归结式仍放入 S 中反复归结过程直到得到空子句

完备:即对于正确的逻辑公式,使用该方法可以在有限步内得到结论
控制策略:

- 删除策略:
  - 设有两个子句 C 和 D,若有置换σ使得$C\sigma\subset D$成立,则称子句 C 把子句 D 归类
  - 小的可以代表大的,大的可以删去,是完备的
- 支撑集策略:
  - 选定支撑集后每次归结至少有一个子句来自于支撑集 T 或由 T 导出的归结式,是完备的
  - 通常选目标结论的否定所转换的子句集作为支撑集
- 语义归结:
  - 子句 S 按照一定的语义分成两部分,每部分内的子句间不允许作归结
  - 引入了文字次序,约定归结时其中的一个子句的被归结文字只能是该子句中“最大”的文字
  - 是完备的
- 线性归结
  - 线性归结策略首先从子句集中选取一个称作顶子句的子句 C0 开始作归结
  - 归结的效率取决于选的顶子句好不好
  - 是完备的
- 单元归结
  - 每次归结都有一个子句是单元子句(只含一个文字的子句)
  - 初始子句集中没有单元子句时,单元归结策略无效,称为反之不成立
  - 不完备的
- 输入归结
  - 输入归结策略要求在归结过程中,每一次归结的两个子句中必须有一个是 S 的原始子句
  - 不完备的

Herbrand 定理:
公式 G 永真:G 对于所有解释都为真
H 域:

- 简单来说,H 域包含了所有可以通过反复应用函数符号到常量上而生成的基本项
- 举个例子,对于给定谓词公式集

$$\forall x(P(x) \rightarrow Q(f(x), a))$$

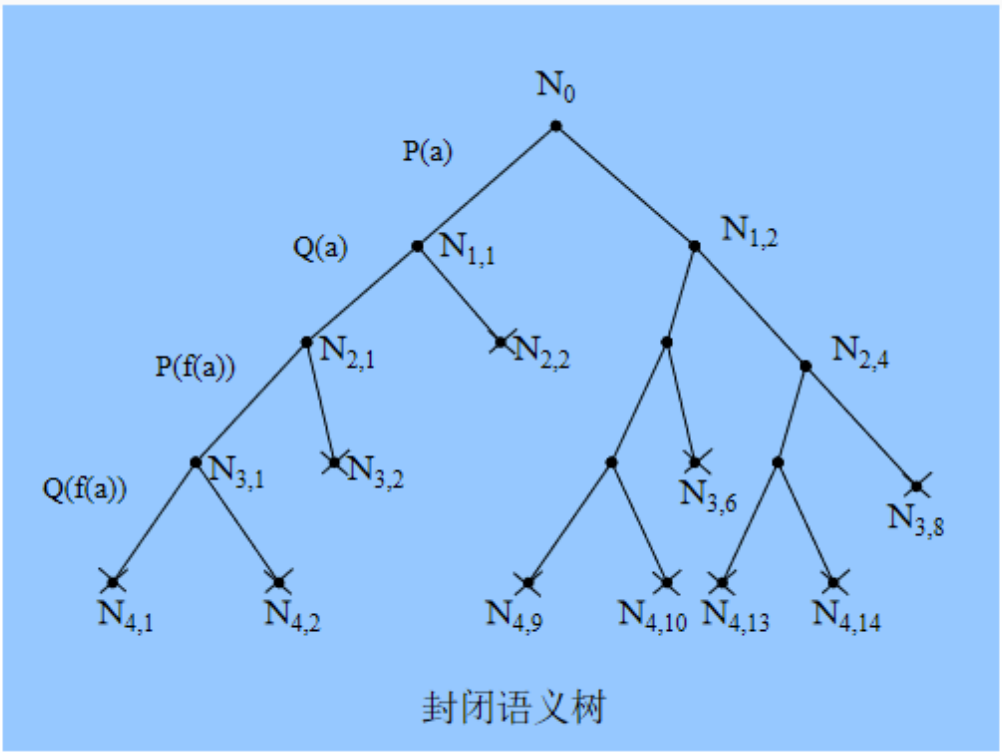
$$\exists y(R(y, g(y)))$$

$$S(b)$$

常量为{a,b},涉及到的函数符号为 f,g,谓词符号位 P,Q,R,S,则$D_H^0 = \{a, b\}, D_H^1 = D_H^0 \cup \{f(a), f(b), g(a), g(b)\}$,以此类推
H 解释:

- 谓词公式 G 在论域 D 上任何一组真值的指定称为一个解释(意思就是带入属性的值)
- 子句集 S 在的 H 域上的解释称为 H 解释

基例:S 中某子句中所有变元符号均以 S 的 H 域中的元素代入时,所得的基子句 C'称为 C 的一个基例



$S = \{\sim P(x) \vee Q(x),\ P(f(y)),\ \sim Q(f(y))\}$

如图就表示的是一棵封闭的语义树,所有延申出去的子节点最后都是到达失败节点,显然这幅图的 H 域没有常量,我们不妨设$D_H^0 = \{a\}$,得$H_\infty = H_1 \cup H_2 \cup ... = \{a, f(a), f(f(a)), ...\}$,谓词符号有 P,Q,设树往左走是 T 往右走是 F,奇数层是 P 偶数层是 Q,当走到的结点带入 S 中,若不可能同时满足所有子句为 T,则封闭
此即:

- 子句集 S 是不可满足的,当且仅当对应于 S 的完全语义树是棵有限封闭树
- 子句集 S 是不可满足的,当且仅当存在不可满足的 S 的有限基例集

- H 是一种半可判定算法,归结原理是语义树的倒塌过程,才是完备的

ch4:不确定性推理

概述:

不确定性推理就是:
- 从不确定性的初始证据出发
- 通过运用不确定性的知识
- 最终推出具有一定程度的不确定性但却是合理或者近乎合理的结论的思维过程

不确定性的基本问题包括:
- 不确定性的表示
- 不确定性匹配算法
- 组合证据的不确定性
- 不确定性的传递算法
- 结论不确定性的合成

不确定性一般分为两类:
- 知识的不确定性:静态强度,可信度
- 证据的不确定性:动态强度

组合证据的可信度计算:
- 最大最小法:

$$T(E_1 \wedge E_2) = \min\{T(E_1), T(E_2))\}, T(E_1 \vee E_2) = \max\{T(E_1), T(E_2))\}$$

- 概率法:

$$T(E_1 \wedge E_2) = T(E_1) \cdot T(E_2), T(E_1 \vee E_2) = T(E_1) + T(E_2) - T(E_1) \cdot T(E_2)$$

- 有界法:

$$T(E_1 \wedge E_2) = \max\{0, T(E_1) + T(E_2) - 1\}, T(E_1 \vee E_2) = \min\{1, T(E_1) + T(E_2)\}$$

不确定性推理方法的两种方式:
- 模型法: 把不确定的证据和不确定的知识分别以某种度量标准对应起来,并给出更新结论不确定的算法无论用何种控制策略推理的结果都是唯一的.分为数值方法与非数值方法,数值方法又分为基于概率的不确定性推理和基于模糊理论的不确定性推理
- 控制法:在控制策略中处理不确定性,通过识别领域中引起不确定性的某些特征来用相应的控制策略来限制或者减少不确定性对系统产生的影响

贝叶斯理论:

全概率公式:

$$P(B) = \sum_{i=1}^{n} P(A_i) \times P(B|A_i)$$

Bayes 定理:

$$P(A_i|B) \times P(B) = P(A_i) \times P(B|A_i)$$

设产生规则为 IF E THEN H,那么 P(H|E)是在证据 E 出现时 H 的确定性程度
逆概率方法:由于 P(H|E)在实际情况中难以得到,我们考虑 Bayes 从 P(E|H)推出,要求子事件两两独立

贝叶斯网络:

一个有向无环图 DAG,节点与结点之间构成了条件概率表 CPT,每一个节点代表一个证据,每一条弧代表一条规则

条件独立:Z 条件下若 X 与 Y 相互独立,P(X|Y,Z)=P(X|Z)

条件概率的迭代(这个过程可以一直进行下去):

$$P(X_1, X_2, ..., X_N) = P(X_1|X_2, ..., X_N) \times P(X_2, ..., X_N)$$

D-分离:

- 串行,$X \to Z \to Y$,X 和 Y 被 Z 是 D 分离的,因为 Z 已知通道就被堵塞
- 分叉连接,$X \leftarrow Z \to Y$,X 和 Y 被 Z 是 D 分离的,因为 Z 已知 XY 就会相互独立
- 汇集连接,$X \to Z \leftarrow Y$,通常情况下 X 和 Y 是独立的,但是如果 Z 或 Z 的后代被观测到已知 X 就可以推出 Y 了

对于贝叶斯网络而言:

- 对给定的证据集,如果对贝叶斯网络中的两个结点之间的无向路径而言
- 证据集中的所有元素都可以堵塞这两个结点,那么称这俩点被该证据集 D 分离
- 可表示为:$P\left(V_i|V_j, \varepsilon\right) = P(V_i|\varepsilon)$

推理需求:

- 因果推理
- 诊断推理
- 辩解推理
- 不被 D 分离就是不被阻塞也就是不条件独立

主观贝叶斯:

目标是通过观察 S 找到证据 E 最终得到结论 H

面对知识的不确定性:设产生规则为 IF E THEN (LS,LN) H P(H),其中 P(H)称为 H 的先验概率,LS 称为充分性度量,代表 E 对 H 的支持程度

$$LS = \frac{P(E|H)}{P(E|\neg H)}$$

LN 称为必要性度量,代表非 E 对 H 的支持程度

$$LN = \frac{P(\neg E|H)}{P(\neg E|\neg H)}$$

LS,LN 共同代表了知识的静态强度

面对证据的不确定性:由于主观给定 P(E|S),即动态强度有所困难,实际中用可信度 C(E|S)代替 P

$$P(E|S) = \begin{cases} \frac{C(E|S)+P(E)\times(5-C(E|S))}{5} & 0 \le C(E|S) \le 5 \\ \frac{P(E)\times(5+C(E|S))}{5} & -5 \le C(E|S) < 0 \end{cases}$$

不确定性的传递方法:

引入几率函数

$$\Theta(x) = \frac{P(x)}{1-P(x)}, \text{则} P(x) = \frac{\Theta(x)}{1+\Theta(x)}$$

当证据肯定存在时,P(E)=P(E|S)=1

$$\Theta(H|E) = LS \times \Theta(H)$$

当证据肯定不存在时,P(E)=P(E|S)=0

$$\Theta(H|\neg E) = LN \times \Theta(H)$$

当证据不确定时:

- 当 P(E|S)=1 时,P(H|S)=P(H|E),证据一定存在
- 当 P(E|S)=0 时,P(H|S)=P(H|¬E),证据一定不存在
- 当 P(E|S)=P(E)时,P(H|S)=P(H),证据 E 与观察 S 无关
- 其他值的时候在 x 坐标轴为 P(E|S),y 坐标轴为 P(H|S)的图中对上述内容线性插值
- 横坐标代表在得到观察的条件下能得到证据的概率,纵坐标代表在得到观察的条件下能直接得到结论的概率,中间的关键点为(P(E),P(H)),计算端点值的时候可以假设某个先验概率为 1 来求再插值

LS 的性质:为 1 时 E 与 H 无关
LN 的性质:为 1 时候¬E 与 H 无关

可信度方法:
根据经验对一个事物和现象为真的相信程度称为可信度
知识的不确定性设产生式为 IF E THEN H (CF(H,E)),是静态强度,取 0-1
证据的不确定性用可信度因子来表示,如 CF(E)=0.6,是动态强度
传递算法,结论的可信度为:CF(H)=CF(H,E)×CF(E)
若静态强度带阈值,表示为(CF(H,E),λ),只有 CF(E)不低于阈值时才可被应用
若产生式的证据项表示为AND $E_i(\omega_i)$,表示 CF(E)计算时需要用该因子进行加权
若产生式的证据项表示为AND $E_i(cf_i)$,表示 CF(E)的计算方式变为计算$\sum(CF(E)-cf_i)$

模糊推理:

特征函数:设 A 是论域 U 上的一个集合,对于任意 u∈U

$$C_A(u) = \begin{cases} 1 & 当u \in A \\ 0 & 当u \notin A \end{cases}$$

则称$C_A(u)$为集合 A 的特征函数
模糊集的本质是把特征函数的取值范围从{0,1}推广到[0,1]上
μ_A把$\forall u \in U$ 映射到$[0,1]$, 即$\mu_A : U \to [0,1]$或$u \to \mu_A(u)$,则称μ_A是定义在 U 上的一个隶属函数,其值称为隶属度
若论域是离散的,则写成

$$A = \sum \mu_A(u_i)/u_i$$

否则用实函数表示即可
扎德记号可以统一表示以上两种方法$A = \int_{u \in U} \mu_{A(u)}/u$采用最大最小法完成模糊集的合取析取,非直接用 1 减就行
模糊关系的合成$R_1 \circ R_2$,类比矩阵乘法,加就是析取乘就是合取
模糊逻辑:产生式表示为 X IS A (CF)
基本概念扩充法:

$$\mu_{极大}=\mu_{大}^{4}(u)\qquad \mu_{很大}=\mu_{大}^{2}(u)$$

$$\mu_{相当大}=\mu_{大}^{1.5}(u)\qquad \mu_{比较大}=\mu_{大}^{0.75}(u)$$

$$\mu_{有点大}=\mu_{大}^{0.5}(u)\qquad \mu_{稍许有点大}=\mu_{大}^{0.25}(u)$$

贴近度方法求匹配度:

$$(A,B)=\frac{1}{2}[A\cdot B+(1-A\odot B)]$$

其中$A\cdot B$是对应各项合取再析取，$A\odot B$是对应各项析取再合取
语义距离法求匹配度:

$$d(A,B)=(\sum_{i=1}^{n}\left(\frac{\left|\mu_{A(u_i)}-\mu_{B(u_i)}\right|^{q}}{n}\right)^{\frac{1}{q}}$$

称为明可夫斯基距离,当 n=2 时称为欧几里得距离,当 n=1 时称为曼哈顿距离或海明距离
若一个证据有多个子条件,可以使用取极小法或者相乘法来求总匹配度
自然演绎与模糊推理都具有三种基本模式:

- 假言推理:前提到结论
- 拒取式推理:结论到前提
- 三段论推理:传递推理

扎德法推理:

对于模糊假言推理,若已知证据为 X IS A' ,则${B_m}'=A'\circ R_m, {B_a}'=A'\circ R_a$
对于模糊拒取推理,若已知证据为 Y IS B',则${A_m}'=R_m\circ B', {A_a}'=R_a\circ B'$
R_m定义为

$$\int_{U\times V}(\mu_A(u)\wedge\mu_B(v))\vee(1-\mu_A(u))/(u,v)$$

R_a定义为

$$\int_{U\times V}1\wedge(1-\mu_A(u)+\mu_B(v))/(u,v)$$

麦姆德尼方法:

在扎德的基础上,计算方式同 m 方法和 a 方法,引入 c 方法,定义

$$R_C=A\times B=\int_{U\times V}\mu_A(u)\wedge\mu_B(v)/(u,v)$$

米祖莫托方法:

在上述基础上引入一系列方法如 s 方法,g 方法,计算方式仍同 m 方法和 a 方法

$$R_S(i,j)=\begin{cases}1 & \mu_A(u)\le\mu_B(v)\\ 0 & \mu_A(u)>\mu_B(v)\end{cases}$$

$$R_g(i,j)=\begin{cases}1 & \mu_A(u)\le\mu_B(v)\\ \mu_B(v) & \mu_A(u)>\mu_B(v)\end{cases}$$

模糊关系的性能分析:

这里的 very 指的是很大,more or less 指的是有点大

- 对于假言推理:
  - A'=A,then B
  - A'=very A,then very B or B
  - A'=more or less A,then more or less B or B
  - A'= not A,then not B or unknown
- 对于拒取推理:
  - B'=not B,then not A
  - B'=not very B,then not very A
  - B'=not more or less B,then not more or less A
  - B'= B,then A or unknown

模糊三段论:

若三段论成立,有$R(A,B)\circ R(B,C)=R(A,C)$反之亦然

上文提及的方法中,m 与 a 不满足三段论,c,s,g 均满足

多维模糊推理:

若前提 A 又可划分为一系列子前提A_i,则称前提条件为复合条件,有三种处理方法:

- 扎德方法
  - 求出子条件的交集,记为 A
  - 求出 R(A,B)
  - 求出子证据的交集,记为 A'
  - $B'=A'\circ R$
- 祖卡莫托方法
  - 求出子条件的B_i'
  - $B'=\bigcap_{i=1}^{n}B_i$
- 苏更诺方法
  - 基于递归来求
  - $B'=A_n'\circ R(A_n,B_{n-1}')$

简单多重模糊推理:

产生式为 IF X IS A THEN Y IS B ELSE Y IS C

R_m'定义为

$$\int_{U\times V}(\mu_A(u)\wedge\mu_B(v))\vee(1-\mu_A(u)\wedge\mu_C(v))/(u,v)$$

R_a'定义为

$$\int_{U\times V}1\wedge[1-\mu_A(u)+\mu_B(v)]\wedge[\mu_A(u)+\mu_C(v)]/(u,v)$$

带有可信度因子的模糊推理:

- 简单模糊推理
  - CF=$\delta_{\text{match}}\times CF_k\times CF_e$
  - CF=$\delta_{\text{match}}\ \min\{CF_k,CF_e\}$
  - CF=$\delta_{\text{match}}\ \max\{0,CF_k+CF_e-1\}$
  - CF=$\min\{\delta_{\text{match}},CF_k,CF_e\}$
- 多维模糊推理

  - 取极小法
  - 相乘法
- 若 Y IS B_1' CF1 且 Y IS B_2' CF2,现要合成$B' = B_1' \cap B_2'$,那么 CF=CF1+CF2-CF1×CF2

证据理论

证据理论满足比概率论弱的公理,能够区分不确定与不知道的差异，能处理由不知道引起的不确定性

粗糙集理论

对象按照属性的取值情况形成若干等价类(考虑数字逻辑相关内容)

ch5:搜索

搜索方式:

- 盲目搜索
- 启发式搜索

回溯策略:

- N 皇后,算法讲过了,略
- 回溯搜索算法存在问题
  - 深度问题:需要对搜索深度加以限制
  - 死循环问题:记录初始状态到当前的路径,遇到循环及时剪枝

图搜索:

- 回溯搜索只保留从初始状态到当前状态的一条路径,但图搜索保留所有已经搜索过的路径
- 本书定义根结点深度为 0
- 一条路径的耗散值等于连接这条路径各节点间所有耗散值的总和,说白了就是 cost
- 状态空间搜索:
  - DFS:不能保证找到最优解,当深度限制不合理时，可能找不到解,最坏情况时，搜索空间等同于穷举
  - BFS:当问题有解时，一定能找到解,效率较低
  - OPEN 表,存放刚生成的结点,记录状态结点和父结点
  - CLOSE 表,存放将要拓展和已拓展的结点,记录编号,状态节点,父节点
  - BFS 是将父节点的子节点放入 OPEN 表的尾部而 DFS 是首部
- 启发式搜索:定义一个评价函数 f,对当前的搜索状态进行评估来找出一个最有希望的节点扩展

A 算法:

- 评价函数的格式:$f(n) = g(n) + h(n)$
- $f^*(n) = g^*(n) + h^*(n)$从 s 经过 n 到 g 的最小耗散值,没有星号的表示是估计值
- $f(n)$代表从起点经过 n 到终点的代价,$g(n)$代表从起点到 n 的代价,$h(n)$代表从 n 到终点的代价
- 与 BFS 和 DFS 不同的是,每轮循环将 OPEN 的节点按 f 从小到大排序

A^*算法:

- 在 A 算法的基础上,满足条件$h(n) \leq h^*(n)$,即估计函数不超实际情况
- $h(n) \equiv 0$,算法等同 BFS
- 最优路径的总代价是一个固定值
  - 从起点看，是$h^*(s)$(从起点到终点还要多少代价)
  - 从终点看，是$g^*(t)$(从起点到终点已花多少代价)
  - 在最优路径上的任何中间节点 n 处,是$f^*(n)$(从起点经 n 到终点的总最优代价)
- 对有限图,如果从初始节点 s 到目标节点 t 有路径存在,则算法 A 一定成功结束
- 对无限图,如果从初始节点 s 到目标节点 t 有路径存在,则 A*一定成功结束

- 对无限图,若有从初始节点 s 到目标节点 t 的路径,则 A*无法结束时,在 OPEN 表中即使最小的一个 f 值也将一直往下增到任意大(可能不具备完备性,如探索一条无限长且不收敛的次优路径),或有 f(n)>f*(s),因为 f*(s)是一个常量
- A*结束前,OPEN 表中必定存在一个节点 n,$f(n)=g(n)+h(n)=g^*(n)+h(n)\leq g^*(n)+h^*(n)=f^*(n)=f^*(s)$
- OPEN 表上任一具有$f(n)<f^*(s)$的节点 n,最终都将被 A*选作扩展的节点
- 可采纳性定理:若存在从初始节点 s 到目标节点 t 有路径则 A*必能找到最佳解结束,但这个时候对完备性有要求
  - 每个节点只有有限数量的后继节点
  - 所有边的代价都必须大于一个正数,不能有零代价边也不能有代价可以无限趋近于零的边
- 如果$h_2(n)>h_1(n)$(目标节点除外),则 A1 扩展的节点数≥A2 扩展的节点数
- 同一个节点无论被扩展多少次，都只计算一次,解决方案
  - 对 h 加以限制
  - 对算法加以改进

h 单调:一个启发函数 h,对所有节点n_i和n_j,其中n_j是n_i的子节点,满足

$$\begin{cases}h(t)=0\\h(n_i)\leq c(n_i,n_j)+h(n_j)\end{cases}$$

若 h(n)是单调的,则 A*扩展了节点 n 之后就已经找到了到达节点 n 的最佳路径
若 h(n)是单调的,则由 A*所扩展的节点序列有$f(n_i)\leq f(n_j)$
与或树:
- 一个问题分解的过程,与树表示子问题需要同时解决才能往上推,或树表示子问题解决一个即可往上推
- 本原问题:直接可解的问题,不用再往下走了
- 端节点:所有叶子节点
- 终止节点:已知是否能被解决的端节点,用于倒推解树
- 可解节点:终止节点或者子节点满足该子树可解条件,否则不可解

博弈树:
- 极大极小搜索过程,一方始终选评估最大往下走,另一方始终选最小往下走
- 优化前把搜索树的生成和估值两个过程分开进行，导致效率低
- $\alpha-\beta$剪枝:
  - 极大节点的下界为α,即玩家 A 能获得的最低分数,初值是负无穷
  - 极小节点的上界为β,即玩家 A 能获得的最高分数,初值是正无穷
  - α和β的传递时先左子树再回根节点再右子树
  - MAX 层只改变$\alpha=\max\{$自己, 下一层α, 下一层$\beta\}$
  - MIN 层只改变$\beta=\min\{$自己, 下一层α, 下一层$\beta\}$
  - 对于每个节点,$\alpha\geq\beta$时剪枝

NP:
- P 问题:多项式时间内可以解决的问题
- NP 问题:给定一个结果,可以在多项式时间内验证此结果是否是问题的解的问题
- NPC 问题:即 NP 完全问题,NP 类中最难的问题,是一个 NP 问题,所有 NP 问题都可以规约为此问题
- NPH 问题:若一个 NPC 问题没有 NP 问题的约束,那么称为 NP 难问题
- 包含关系:

$$P\subset NP, NPH\cap NP=NPC, P\cap NPC=\Phi$$

- NPC 本身难解决,目前找不到多项式时间通解,但它的解很容易验证,所有其他 NP 难题转化为 NPC 的一个特例

ch6 机器学习:

按照有无指导分为有监督学习(有标签数据),无监督学习(只有输入),强化学习(通过与环境交互学习)
按照学习方法分为机械式学习(重复记忆),指导式学习(有明确反馈标签),范例学习(从具体例子中归纳),类比学习(跨领域知识迁移,找相似性),解释学习(基于原理+示例泛化)
按照推理策略分为演绎学习(从普遍到特殊),归纳学习(从特殊到普遍),类比学习,解释学习
按照综合因素分为人工神经网络学习,进化学习(自然选择思想的优化策略),集成学习(组合多个模型来提高性能),概念学习,分析学习,范例学习
机器学习的基本问题:

- 分类问题
- 预测问题
- 聚类问题
- 联想问题
- 优化问题

度量学习成果的有效性:

- 误差:比如用方差
- 正确率和错误率
- 召回率和精度
  - a:判定属于类且判定正确
  - b:判定属于类且判定错误
  - c:判定不属于类且判定正确
  - d:判定不属于类且判定错误
  - 则有

$$\text{Precision}(T) = \frac{|a|}{|a+b|}$$

$$\text{Recall}(T) = \frac{|a|}{|a+d|}$$

决策树:

决策树学习的目的是为了产生一棵泛化能力强,即处理未知示例能力强的决策树
从根结点到每个叶结点的路径对应了一个判定测试序列
决策树学习的关键在于如何选择最优划分属性,一般而言，随着划分过程不断进行，我们希望决策树的分支结点所包含的样本尽可能属于同一类别，即结点的纯度越来越高
经典的属性划分方法(ID3):

- 信息增益
  - 信息熵是度量样本集合纯度最常用的一种指标
  - 假设样本集合 D 共有γ类样本,设第 k 类样本占比为p_k,定义 D 的信息熵为

$$Ent(D) = -\sum_{k=1}^{\gamma} p_k \log_2 p_k$$

Ent 的值越小则纯度越高

  - 若 p=0 则$p\log_2 p=0$,这意味着熵最小值为 0,最大值为$\log_2 \gamma$
  - 对目标的离散属性 a 有 V 个可能的取值$\{a^1,a^2,...,a^V\}$,信息增益定义为

$$\text{Gain}(D,a)=Ent(D)-\sum_{v=1}^{V}\frac{|D^v|}{|D|}\cdot Ent(D^v)$$

    $\frac{|D^v|}{|D|}$实际上代表的是占比
  - 信息增益越大,意味着使用该属性来进行划分所获得的纯度提升越大
  - 显然,信息增益对可取值数目较多的属性有所偏好
- 增益率
  - 固有值定义为

$$IV(a)=-\sum_{v=1}^{V}\frac{|D_v|}{|D|}\log_2\frac{|D_v|}{|D|}$$

  - 增益率定义为

$$\text{Gain\_ratio}(D,a)=\frac{\text{Gain}(D,a)}{\text{IV}(a)}$$

  - 增益率对可取值数目较少的属性有所偏好
- 基尼指数
  - 数据集的纯度可用基尼值度量,基尼值越小,数据集纯度越高

$$Gini(D)=1-\sum_{k=1}^{\gamma}p_k^2$$

  - 属性 a 的基尼指数定义为

$$\text{Gini\_index}(D,a)=\sum_{v=1}^{V}\frac{|D_v|}{|D|}\text{ Gini}(D^v)$$

    应选择那个使划分后基尼指数最小的属性作为最优划分属性

剪枝:

- 判断决策树泛化性能是否提升的方法(其实机器学习都这样):
  - 留出法:预留一部分数据用作验证集
  - 交叉验证(每一个子集都做一次测试集)
  - 随机法(每一个子集都未必能做到一次测试集)
- 可通过剪枝来一定程度避免因决策分支过多,以致于把训练集自身的一些特点当做所有数据都具有的一般性质而导致的过拟合
- 基本策略:
  - 前剪枝(预剪枝)
    - 选取某个属性划分方法
    - 其类别标记为训练样例数最多的类别
    - 分别计算划分前和划分后的验证集精度
    - 若划分后能提高验证集精度，则划分
    - 具有欠拟合风险,杜绝了后续划分可能导致性能提高的可能性
  - 后剪枝
    - 从训练集生成一棵完整的决策树
    - 自底向上地对非叶结点进行考察
    - 若划分前能提高验证集精度,则剪枝
    - 后剪枝比预剪枝保留了更多的分支,欠拟合风险小,训练时间开销大

连续值处理:
- 计算两两的平均值作为候选划分点
- 对每个划分点进行离散的信息增益计算,选择使之最大化的点,将该项数据按这个指标化为两部分

缺失值处理:
- 定义$\hat{D}$表示 D 在属性 a 上没有缺失的样本子集,$\widehat{D^v}$表示在取值a^v的样本子集,$\widehat{D^v}$表示属于$\hat{D}$第 k 类的样本子集
- 分别定义ρ是无缺失值样本的占比,$\hat{p_k}$是无缺失值样本中第 k 类的占比,$\hat{r_v}$是无缺失样本中属性 a 上取值a^v样本所占的比例
- 基于上述定义,

$$Gain(D,a)=\rho\times Gain\left(\hat{D},a\right)$$

  最初设所有取值的权值都为 1
- 当处理非缺失值的时候,每次计算权重都是 1,当处理缺失值的时候,每次计算都按照不同概率划分入子节点
- 划分权值为$\hat{r_v}$

多变量决策树:非叶节点不再是仅对某个属性,而是对属性的线性组合

贝叶斯学习

基于贝叶斯理论的机器学习方法,贝叶斯法则的核心是贝叶斯公式

$$P(H|D)\times P(D)=P(D|H)\times P(H)$$

其中
- $P(H)$称为假设 H 先验概率
- $P(D)$表示训练数据 D 的先验概率,与训练数据相独立
- $P(D|H)$表示 H 成立时 D 的概率,称为类条件概率或似然度
- $P(H|D)$表示给定 D 时 H 成立的后验概率,后验概率是学习的结果,是对先验概率的 D 条件修正
- 观察到的每个训练样例改变某假设的估计概率
- 允许假设做出不确定性的预测

贝叶斯最优假设
- 极大后验假设
  - 在候选假设集合 H 中寻找对于给定数据 D 使后验概率$P(H|D)$最大的那个假设

$$P(H|D)=\frac{P(D|H)P(H)}{P(D)}\propto P(D|H)P(H)$$

    因为 P(D)是不依赖于 H 的常量
  - 不仅看数据,还结合了对假设本身的先验信念,人的经验会影响这一学习
- 极大似然假设
  - 候选假设集合 H 中选择使给定数据 D 似然度$P(D|H)$最大的那个假设
  - 只看数据选择最能解释当前观测数据的假设
- 贝叶斯最优分类器
  - 是对极大后验假设的发展
  - 能从给定训练数据中获得最好性能,但是其算法开销比较大
  - 设 V 表示类别集合,对 V 中任意一个类别v_j,选择使得

$$\sum_{h_i \in H} P(v_j|h_i) \cdot P(h_i|d)$$

最大的作为结论

朴素贝叶斯**(Naïve Bayes):**
对于一个实例D往往具有很多属性$< a_1, a_2, ..., a_n >$,可采用朴素贝叶斯或贝叶斯网络来解决高维数据问题
朴素贝叶斯分类器采用最简单的假设:

- 从样本中获取验证属性的相关参数
- 假定数据各属性之间相互条件独立
- 将上述条件带入h_{MAP}则

$$h_{\text{NB}} = \text{argmax}_{h_i \in H} P(h_i) \Pi_j P(a_j|h_i)$$

这一假定一定程度上限制了朴素贝叶斯方法的适用范围

- 即便如此,许多领域在违背这种假定的条件下,朴素贝叶斯学习也表现出相当的健壮性和高效性
- 在实际计算中,最终可能由于样本数据量较小,出现某个算子为 0,导致算法不够鲁棒,我们引入 m-估计方法
  - 设有

$$P(a_i|H) = \frac{cnt(a_i) + mp}{\sum_{a_i \in W} cnt(a_i) + m}$$

W 是a_i在样本中的所有可能取值,p 是领域专家给出的先验估计概率,m 表示验证时对先验信念的信任程度(读到一次属性a_i相当于读了几个验证例)

  - 估计 p 最常用的方法就是假定均匀分布的先验概率,$mp = 1$,表示读到了一次a_i与该次验证集相同,分母m相应的加上词汇表的大小,这种情况又称为拉普拉斯平滑

聚类:

聚类就是对一堆观测数据进行划分,要求同簇内数据的相似度尽可能大,不同簇间数据的相似度尽可能小
相似度$sim(d_k, d_l)$的定义方式:

- 距离定义
  - 曼哈顿距离
  - 欧几里得距离
  - 明可夫斯基距离
- 密度定义
- 概念定义

聚类方法:

- 分层聚类
  - 在聚类过程中生成一个聚类树
  - 完整聚类树的最顶端代表把整个数据集划作为一个簇,最底端代表把数据集中每一个数据都当作一个簇
  - 聚类树的不同层次可以表示聚类的不同粒度
  - 可以自顶向下构造聚类树,称为分裂聚类法;又可以自底向上构造聚类树,称为凝聚聚类法
  - 需要一个参数指明停止聚类的条件,通常用生成簇的期望个数 k 判断停止条件
  - 在聚类过程中一次性就建好了聚类树,没有回溯调整操作

- Linkage 算法(凝聚)
  - 用距离定义相似度,
$$d(C_1, C_2) = L\{d(x, y)|x \in C_1, Y \in C_2\}$$
  - 单链 Slink,L 是极小化算子,定义两个簇的距离等于两簇最近两个点间的距离
  - 全链 Clink,L 取极大化算子,定义两个簇的距离等于两簇最远两个点间的距离
  - 均链 Alink,L 取平均算子,定义两个簇的距离等于两簇点间距离的平均值
  - 以每个数据点为一个簇,并将其放入有效簇集合中,计算有效簇集合中任意两簇之间的距离,按升序排列成簇间距离队列
  - 不断合并距离最小的两个簇,并把新簇放入有效簇集中,把两个旧簇删除,直至距离队列空,继续向上递归
  - 它会持续合并直到所有数据点都归属于一个大簇
  - 通过切割生成的树状图来得到所需的特定数量的簇
- CURE 算法(凝聚)
  - 也使用簇间距离定义相似度,簇间距离定义为选好的各簇代表点的最短距离
  - 选取距离该簇质心 M 最远的点作为第一个代表点,然后选取距全部现有代表点最远的点作为下一个代表点(同 Slink),如此重复选择若干个,接下来所有这些代表点还要向簇中心收缩,收缩系数为 a
  - 簇的代表点基本上覆盖了簇的形状,收缩用于中和外边界点对簇的过大影响,收缩公式为 $R_i' = (1-\alpha)R_i + \alpha M$
  - 用了两个措施来加速聚类过程:数据采样(抽样先划分簇,再把未划分的归入前面抽样时划分的簇中),分区(利用分治思想来解决不同的抽样簇)
- 划分聚类
  - 把数据集划分为 k 个簇
  - 逐一把数据点放入合适的簇中
  - 重复扫描数据集多次来全局优化
  - 多数情况下,误差取数据点到簇代表点距离的平方和
  - 孤立点和噪声数据对 K 平均方法的影响更大
  - K-means 方法:
    - 从数据集中选择 K 个数据点作为初始簇代表点划分出 k 个簇
    - 剩下的按照距离,被分配给与其最近的簇
    - 计算每个簇的质心,获得新的代表点(簇的理论中心,可以不存在),重复上述过程直至收敛(分配不再变化)
    - 在簇密度悬殊,簇大小悬殊时可能得不到想要的结果,难以得到非球形簇
  - K-medoids 方法:
    - 同 K 平均方法,只是获取新的代表点时要选离质心最近的点
- 基于密度的聚类
  - 对于给定数据集,在一个给定范围的区域中必须至少包含某个数目的点
  - 可以用来过滤噪声,孤立点数据
  - DBSCAN 算法
    - 对于数据集中的每个点计算其$Eps(\varepsilon)$邻(即以该点为中心半径为 Eps 的球形区域)内包含的数据点数量
    - 当这个数量大于或等于预定义的 MinPts(最小点数)时,该点就被判定为核心点
    - 当两个核心点相互处于彼此的 Eps 邻域内时,它们就被认为是密度可达的,被划分为同一个簇
    - 定义边界点为不是核心点,但处于某个核心点的邻域内的点
    - 其他点都是噪声点

  - 局限性在于簇密度差异过大(密度小的一个簇可能因为 MinPts 的设置而全是噪点)与高维数据(维度变大,欧几里得距离会变得非常大,Eps 过于敏感)
- 基于网格的聚类
  - 把数据空间量化为有限数目的单元,形成一个网格结构
  - 聚类高维 or 海量数据时的思想
  - STING 算法(基于分治思想的凝聚)
    - 每个网格节点包含如下属性:网格内数据点个数,网格内数据的平均值,标准偏差,最小值,最大值,该网格内数据的分布类型
    - STING 假设属性服从某种统计分布
    - 网格邻近的可合并,网格邻近定义为两个网格中心的属性小于指定阈值
    - 没有考虑子节点和其它相邻节点之间的关系，可能降低簇的质量和精确性(例如更小的网格可能会形成形状更精确的簇)
  - CLIQUE 算法
    - STING 将整个大 n 维空间划分为小 n 维空间,CLIQUE 从最小的子空间(一维)一直往上拓展到最大的子空间(n 维)
    - 如果一个 k 维子空间是稠密的,那么它在所有 k-1 维子空间上的投影也必然是稠密的
    - 需要定义密度阈值,子空间若没到这个阈值,这个子空间的点就作为没有意义的密集区域
    - 簇的合并类似于图论中的连通分量寻找,如果这两个共享一个超平面就判定为联通
- 基于模型的聚类

ch7 深度学习

线性分类器:

目标:给定一组输入,将其使用超平面分割开不同类来
识别本身存在问题:
- 识别问题本身的复杂性:由于各种因素(视角、光照、形变等)让机器准确识别物体面临诸多挑战
- 解决识别问题的一种常见范式:采用数据驱动的方法,通过大量的数据让模型(如 KNN)去学习识别规律,模型评估是一项困难的工程

不同视角下的线性分类器:
- 代数视角下:
  - 线性分类器就是

$$f(x, W) = Wx + b$$

    x 是把要识别的内容转换为向量分别与各类别的基本模板矩阵,b 是学习是需要加上的偏差
- 视觉视角下:
  - 代数视角下的每个 W 就是对应出的一个特征模板
- 几何视角下:
  - 线性分类器将一个高维空间用多个超平面分割开来,归类不同的输入向量即可得到划分的对应类别

线性分类器无法处理非线性可分的数据,例如多模态分布,异或问题,同心圆环等
W 权重矩阵:
- 定义损失函数,使得损失越低表示分类器性能越好
  - 给定数据集:

$$\{x_i, y_i\}_{i=1}^{N}$$

其中 x 是输入,y 是所属标签
  - 单个样本的损失定义为

$$L_i(f(x_i, W), y_i)$$

W 是学习得来的权重矩阵
  - 整个数据集的损失就是求和之后再取平均
- 多类别支持向量机,Multiclass SVM Loss
  - 又称为合页损失或最大边距损失
  - 是一种损失函数的形式
  - SVM 的目标是找到一个软边距来分割多组数据,在这个损失函数中体现为,当正确类别的得分比错误类别的最高得分低于这个边距时,模型会受到 Loss 的惩罚
  - 定义为

$$L_i = \sum_{j \neq y_i} max\Big(0, s_j - s_{y_i} + 1\Big)$$

这里的软边距我们一般视为 1
  - 合页损失对足够好的预测不进行额外惩罚,只有当正确类别的权重改变过大到出边距才会受到惩罚
  - 损失的取值范围是[0,∞)
  - 如果求和的时候也计入了正确类别的损失,每个样本损失+1,也就是平移了,或者如果 SVM 使用的是求平均而不是求和,也就是进行权重缩放,都是并不影响最终优化的
  - 如果将多类 SVM 损失函数中的线性惩罚项改为平方惩罚项,会更严厉地惩罚那些预测错误很大的样本,这会导致模型更倾向于对离群点进行拟合,从而可能导致过拟合
- 正则化:
  - 倾向于更平滑更简单的模型,防止过拟合
  - 定义损失函数为

$$L(W) = \frac{1}{N} \cdot \sum_{i=1}^{N} L_i(f(x_i, W), y_i) + \lambda R(W)$$

λ是超参数,控制正则化的强度
  - L_1正则化项:

$$\sum |w_i|$$

L1 正则化倾向于使一些权重值变为零,从而使得模型更稀疏实现特征选择
  - L_2正则化项:

$$\sum w_i^2$$

L2 正则化倾向于使权重值更小,使它们接近于零,从而使得模型更平滑
  - L_1和L_2正则化的组合:使用两个因子与两种正则化组合,这种组合正则化可以同时实现特征选择和权重平滑
- 学习:

$$W_{\text{new}} = W_{\text{old}} - \text{学习率} \times \nabla_W L(W_{\text{old}})$$

即梯度下降
- 交叉熵损失:
  - 定义

$$L_i = -\log P(Y = y_i|X = x_i)$$

作为损失部分

- 定义

$$P(Y = y_i|X = x_i) = \text{softmax}(Wx_i + b)$$

其中 softmax 函数定义为

$$\text{softmax}(w_i) = \frac{e_{w_i}}{\sum_j e_{w_j}}$$

损失的取值范围依旧是零到正无穷

神经网络:

基础的线性分类器过于孱弱,可以引入特征变换来将原始空间的非线性问题映射为线性可分问题
特征变换策略:

- 仿射变换或极坐标变换
- Hog(方向梯度直方图)
  - 在每个区域内计算边缘方向的直方图并按边缘强度加权
  - 捕获纹理和位置信息,对小的图像变化具有鲁棒性
- Bow(词袋模型)
  - 创建码本,提取出许多小的图像块块代表图像中的局部特征,不同的特征视为不同视觉单词
  - 编码图像,将每个图像块与码本中的所有视觉单词进行比较,统计每个视觉单词在图像中出现的频率形成一个直方图

但以上手工策略都与神经网络策略性能相差甚远
人工神经网络的拓扑结构可分为:

- 前馈网络:每一层只接受来自前一层的信息,如 BP 反向传播,CNN 卷积神经网络
- 反馈网络:在前馈网络基础上,输出结果也作为输入层的一个输入,如 RNN 循环神经网络
- 竞争网络:同层神经元之间有横向联系
- 全互联网络:极度非线性,任意两个神经元之间都有可能连接

全连接神经网络:

- 分为输入层,输出层,隐藏层
- 单层感知机其实就是一个带了激活函数的线性分类器,多层感知机也属于前馈网络,因为每一层只接受来自前一层的信息
- 通过其多层结构和关键的非线性激活函数,自动学习和实现复杂的特征变换或空间扭曲,为了防止过拟合也要正则化
- 又称多层感知机,称为全连接是因为每一层的每一个神经元都与前一层的所有神经元连接
- 这里的神经元的连接强度用矩阵中的权重来表示,神经元在这里并不是一个数,而是一个概念上的计算单元
- 举例,一个用 ReLU 作为激活函数的三层全连接神经网络定义为

$$f = W_3max(0, W_2max(0, W_1x))$$

激活函数:

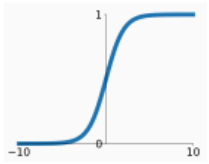
- 比如这里的

$$ReLU(z) = max(0, z)$$

如果不使用激活函数,我们会得到一个线性分类器,因为线性函数的组合仍然是线性函数

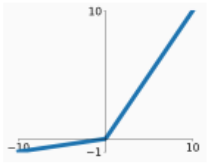
Sigmoid

$$\sigma(x) = \frac{1}{1+e^{-x}}$$



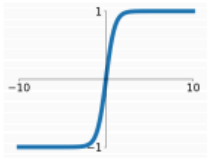
Leaky ReLU

$$\max(0.1x, x)$$



tanh

$$\tanh(x)$$

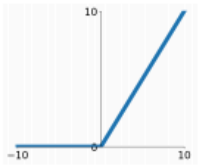


Maxout

$$\max(w_1^T x + b_1, w_2^T x + b_2)$$

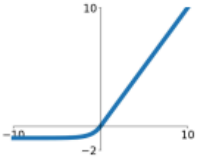
ReLU

$$\max(0, x)$$



ELU

$$\begin{cases} x & x \geq 0 \\ \alpha(e^x - 1) & x < 0 \end{cases}$$



人工神经元:

- 输入就是轴突
- 权重就是突触
- 树突接受加权输入
- 细胞体计算 f
- 激活函数模拟神经元对信号的发放率
- 输出就是下一个轴突

通用逼近定理:

- 通过组合多个基本非线性函数,一个神经网络可逼近任何连续函数,每个隐藏单元都贡献了一个拐点
- 这也说明了神经网络并不是形成凸起,而是学习权重与偏置来模拟逼近能力

损失项是凸的,正则化项也是凸的,并且凸函数的和仍然是凸函数,通常线性分类器的整个优化目标函数是一个凸函数
然而全连接神经网络组合出来的通常是非凸函数,这是优化面临的一个挑战,不引入其他策略容易找到的是局部最优解而不是全局最优解

BP 算法:
即反向传播算法,也称为误差反向传播算法,用于计算超参数的梯度
分为两个阶段,前向传播与反向传播

- 前向传播阶段将数据从输入层流经隐藏层最终到达输出层给出预测结果$\hat{y}$
- 反向传播阶段
  - 目标是计算损失函数 L 对网络中所有权重与偏置的偏导数
  - 对于一个简单的两层网络$L = Loss(f(g(x)))$,展示反向传播的链式法则如何使用

$$\frac{\partial L}{\partial x} = \frac{\partial L}{\partial f} \cdot \frac{\partial f}{\partial g} \cdot \frac{\partial g}{\partial x}$$

只需要将这个链式法则从输出层往前推导到输入层,即可得出途径的所有超参数的梯度,使用梯度下降法即可优化

BP 有梯度消失和梯度爆炸的风险,容易陷入局部最优解,初始化敏感

卷积神经网络:
组成成分:
卷积层:用于提取输入数据的局部特征
激活函数:引入非线性
池化层:降采样,减少参数数量和计算量
全连接层:网络末端整合之前提取的特征
归一化:稳定训练,加速收敛

卷积层:
定义多个大小为通道数×长×宽(权重总数)的滤波器,卷积操作的核心就是将这个滤波器在输入图像的空间维度上滑动(横向和纵向移动),距离用步长定义
卷积的计算方式:

- 将滤波器按照预设好的步长从待卷积的空间从左上角移动到右下角,对每个通道进行矩阵的乘法运算(范围按滤波器的大小)
- 如果有多个滤波器,结果通道数将升维,因为最后的结果是把多个滤波器的通道进行叠加

卷积操作会导致特征图收缩,但我们通过堆叠卷积层可以逐步扩大网络对原始输入图像的感受野
有时也需要解决此问题,引入了填充的概念,在进行卷积操作之前，在输入特征图的边界周围添加额外的像素
每个滤波器需要包含一个偏置项目,因而每个滤波器有权重总数+1 的超参数数量
需要进行的乘加操作数总量就是输出参数总量×滤波器权重数
我们显然能得到卷积层输出尺寸的公式是

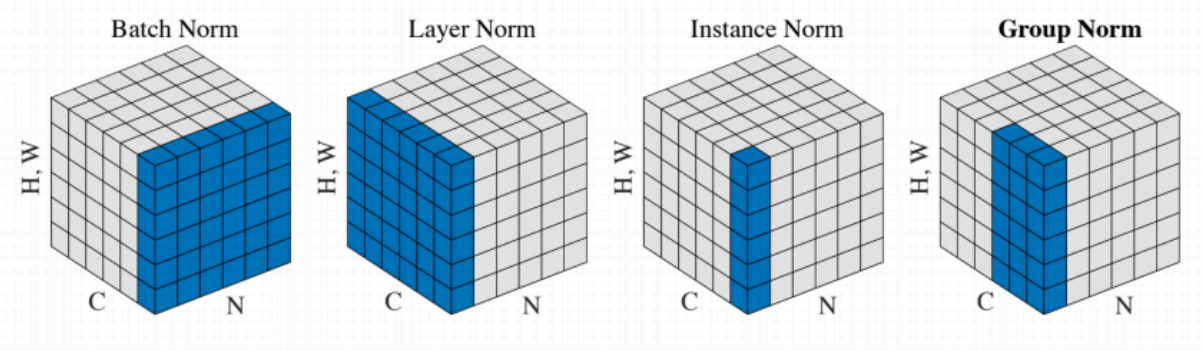
$$\text{Output}_{\text{size}} = \left\lfloor \frac{\text{Input}_{\text{size}} - F + 2P}{S} \right\rfloor + 1$$

其中 F 是滤波器尺寸,P 是填充数,S 是步长

池化层:
降采样,即减小其空间维度同时保留重要信息,降低复杂度,提高鲁棒性
对于池化层同样也有一个池化操作窗口,类似于滤波器的工作原理
其实就是将卷积得到的内容使用池化函数降维,对每个池化窗口执行池化函数的操作:

- 若池化函数取最大池化,从窗口内的所有值中选择最大值作为输出,保留最显著的特征
- 若池化函数取平均池化,计算窗口内所有值的平均值作为输出,保留更平滑的特征

归一化:



这里将输入图层当作一维 H,W 看待,毕竟我们看不到四维
层归一化:归一化是针对每个样本独立进行的
实例归一化:每个样本上的每个通道都独立进行归一化

组归一化:每个样本上的若干通道为一组进行归一化
批归一化:

- BN,Batch Normalization,使其每个小批次内具有零均值和单位方差,归一化是针对每个通道独立进行的
- 减少内部变量偏移,BN 通过固定每层输入的分布来缓解因前一层参数更新导致后一层输入数据分布变化的问题
- 改善优化过程,允许使用更高的学习率来让训练加速
- 对于全连接网络,输入的形状为批量大小×神经元数量(特征维度)
- 对于 CNN 输入的形状为批量大小×通道数×特征图面积
- 过程:
  - 对每个特征维度或每个通道,在当前批次的 N 个样本计算均值μ_B和方差σ_B^2
  - 标准化,公式为

$$\hat{x_i} = \frac{x_i - \mu_B}{\sqrt{\sigma_B^2 + \varepsilon}}$$

    ε是数值稳定性常数,防止除以零
  - 再引入两个可学习的参数缩放因子γ和偏移量β,归一化输出为

$$y_i = \gamma \hat{x_i} + \beta$$

    若模型认为归一化前更好,则令

$$\gamma \approx \sigma_{\text{original}}, \beta \approx \mu_{\text{original}}$$