Knowledge Representation

13

13 Knowledge Representation*

13.1 Symbolic representation

13.2 Neural representation

13.3 Neurosymbolic representation

KR

Knowledge Representation (KR): How knowledge can be modeled and utilized by a computer system

to perform tasks such as reasoning, problem-solving, decision-making, and learning

- Symbolic KR (symbolism, classic AI)
  – Symbols standing for things in the world
  Standpoint: AI needs to represent world knowledge
  like how humans do
- Neural KR (connectionism, generative AI)
  – Embedding spaces (compression) for things in the world
  Standpoint: AI requires data to represent world knowledge in a hidden way
  rather than explicitly representing world knowledge
- Neurosymbolic KR
  – Integration of symbolism and connectionism

KR and AI

KR is fundamental research of AI

- Much of AI involves building systems that are knowledge-based
  reasoning over explicitly represented knowledge
  – language understanding, planning, diagnosis, etc.
- Much of ML involves building systems that are data-driven
  deep learning
  – LLMs, etc.
- Some AI, to a certain extent mix
  – game-playing, motor control, etc.

How much of intelligent behavior is knowledge-based??

Challenges each other

Symbolic representation

- Knowledge
- Production systems
- Structured descriptions
- Semantic web
- Change
- Mental states
- Ontology
- Qualitative physics
- Frame and semantic networks
- Knowledge graph
- Explanation and diagnosis
- Commonsense

Declarative vs. procedural representation

Declarative representation: "know-what"

knowledge expresses in declarative sentences (in a logic)

E.g., descriptive knowledge about the computation of matrixes

Procedural representation: "know-how"

knowledge expresses in procedures (in a programming language)

E.g., the procedure of computing matrixes

Theorem proving (like resolution) is a general-purpose domain-independent method of reasoning

does not require the user to know how knowledge will be used

Want to communicate to theorem-proving procedure by some guidance based on properties of the domain (like LEAN)

– perhaps a specific method to use

– perhaps merely a method to avoid

Knowledge

Knowledge: things in the world

–*Language*

E.g., a logical language

–*Representation*, symbols standing for things

E.g., (logical) sentences

–*Reasoning*

E.g., proofs and model checking

In philosophy, the study of knowledge is called epistemology

Plato defined knowledge as "justified true belie" (ongoing debate)

Belief: not necessarily true and/or held for appropriate reasons

Knowledge base

Separation between the knowledge base and reasoning procedure should be maintained

Knowledge base (KB): to store structured and unstructured inform.

- needed to know facts about the world
- to distinguish from database
  - – not just tables with numbers and strings
- scaled up with Internet documents/hypertext/multimedia
  - – known as Web Content Management

A good KB should be expressive, concise, unambiguous, context-insensitive, effective, clear and correct

Knowledge engineering (expert systems, knowledge-based systems): the process of building a knowledge base

Knowledge engineering vs. software engineering#

The knowledge engineer or agent usually interviews the real experts or environments to become educated about the domain and to elicit required knowledge in a process called knowledge acquisition

	Knowledge engineering	Software engineering (Programming)
1.	Choosing a logic	Choosing a programming language
2.	Building a knowledge base	Writing a program
3.	Implementing the proof theory	Choosing or writing a compiler
4.	Inferring new facts	Running a program

Should be less work

Ontology

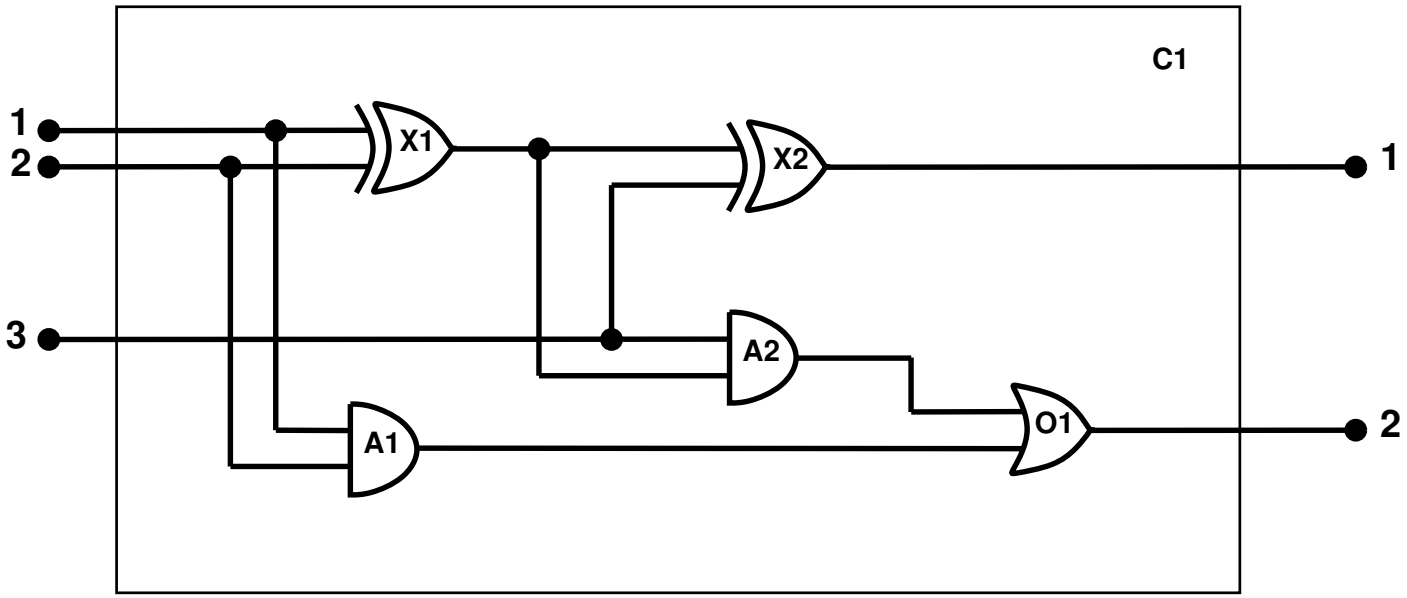
Ontology: a vocabulary for the domain knowledge

Ontological engineering: representing various ontology

The five-step methodology

1. Decide what to talk about
2. Decide on a vocabulary of predicates, functions and constants
3. Encode general knowledge about the domain
4. Encode a description of the specific problem instance
5. Pose queries to the inference procedure and get answers

Example: The electronic circuits domain



Ontological engineering

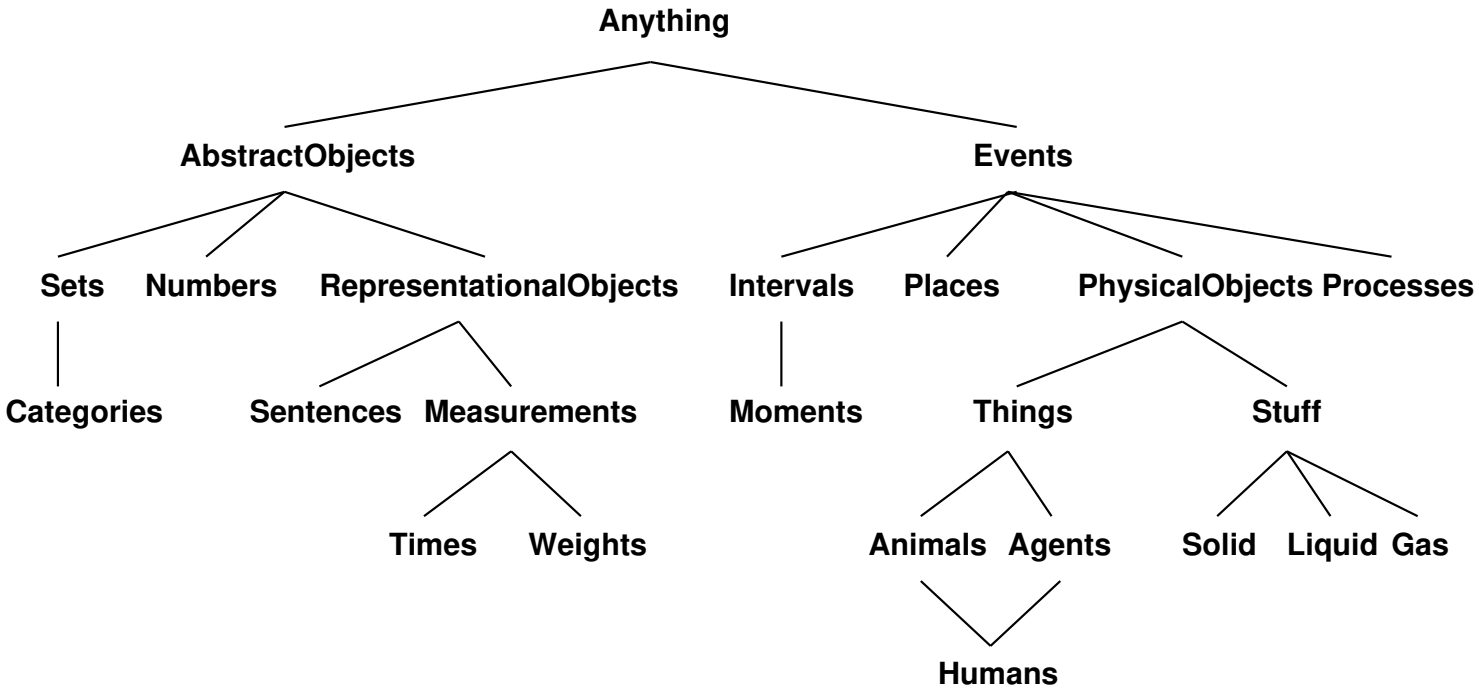
1. Decide what to talk about
   e.g., gates AND, OR, XOR and NOT
2. Decide on a vocabulary of predicates, functions and constants
   e.g., $Out(1, X_1)$
3. Encode general knowledge about the domain
   e.g., $\forall t_1 t_2 Connected(t_1, t_2) \Rightarrow Signal(t_1) = Signal(t_2)$
4. Encode a description of the specific problem instance
   e.g., $Type(X_1) = XOR$
5. Pose queries to the inference procedure and get answers
   e.g., what combinations of inputs would cause the first output of C_1 (the sum bit) to be off? The answer ...

General ontology

A general-purpose ontology has advantages over a special-purpose one

- Categories
- Measures
- Composite objects
- Time, Space, and Change
- Events and Processes
- Physical objects
- Substances
- Mental objects and belief

The world ontology



Hard to build a real-world ontology

Some KBs or ontologies#

There are some routes for building very larger KBs or ontologies:

– CYC: creating the ontology and writing axioms from cyclopedia (1990)

– DBpedia: importing categories, attributes and values from Wikipedia (2007)

– TextRunner: building by reading a large corpus of Web pages (2008)

– OpenMind: building by volunteers who proposed facts and commonsense knowledge in English (2002)

– Knowledge Graph (KG, previous Freebase): building by Google and holding more than 70 billion facts (2012)

– Wikidata, Linking Open Data (LOD), YAGO etc.

Other data sources (also known Deep Web): MusicBrainz, DrugBank etc.

Production systems

Production (rule-based) systems:

– *working memory*: a knowledge base

– *rule memory*: a set of inference rules with form

$p_1 \wedge \cdots \wedge p_n \Rightarrow act_1 \wedge \cdots \wedge act_m$

where p_i are literals, and act_j are actions to take when the p_i are all satisfied – forward chaining

– *match phase*: in each cycle, the system computes the subset of rules whose left-hand side is satisfied by the current contents of the working memory

– *conflict resolution phase*: the system decides which of the rules should be executed

– *act phase*: in each cycle, the system executes the action(s) in the chosen rule(s)

Production systems

Inefficient forward chaining unification match algorithm:

E.g., If there are $w = 100$ elements in working memory and $r = 200$ rules each with $n = 5$ elements in the left-hand side, and solving a problem requires $c = 1000$ cycles, then the naive match algorithm must perform $wrnc = 10^8$ unifications

Rete algorithm of OPS5

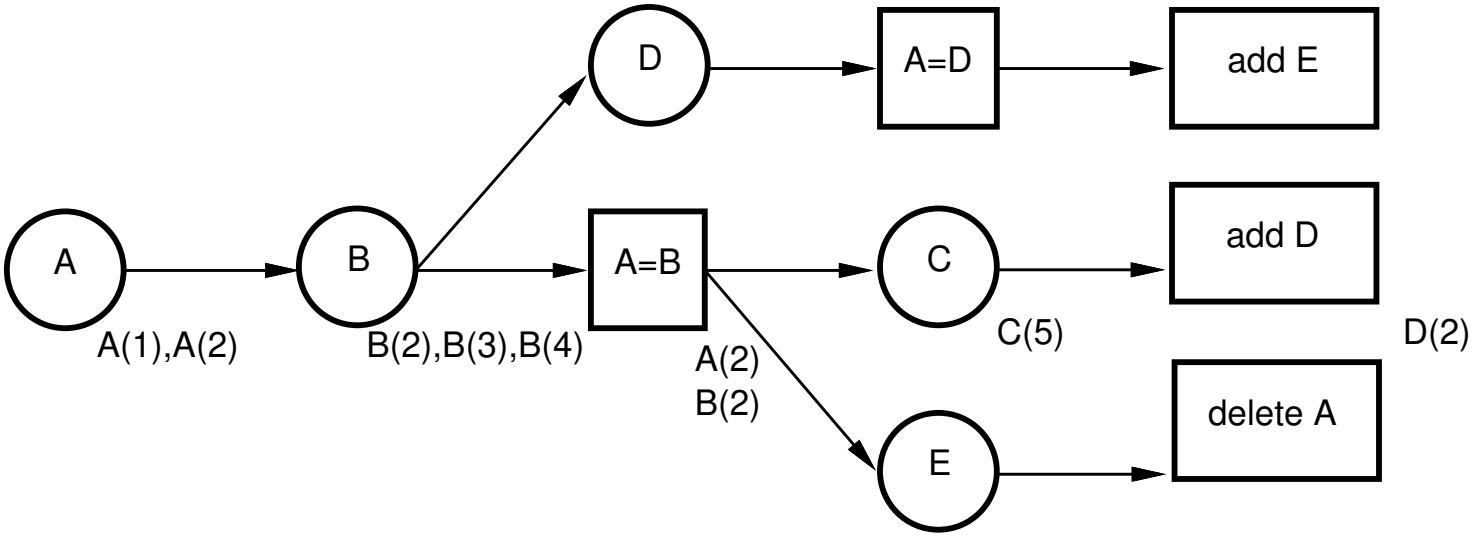
E.g., rule memory

$$A(x) \wedge B(x) \wedge C(y) \Rightarrow addD(x)$$
$$A(x) \wedge B(x) \wedge D(y) \Rightarrow addE(x)$$
$$A(x) \wedge B(x) \wedge E(y) \Rightarrow addDeleteA(x)$$

and working memory

$$\{A(1), A(2), B(2), B(3), B(4), C(5)\}$$

Production systems



Production systems#

Conflict resolution phase: some control strategy

- No duplication
- Recency
- Specificity
- Operation priority

OPS5

1. discard rule instances that have already been used
2. order remaining instances in terms of recency of working memory matching 1st condition (and then of 2nd condition, etc.)
3. if still no single rule, order rules by the number of conditions
4. select arbitrarily among those remaining

Production systems are essentially programming languages

Expert systems#

So-called expert systems are production systems

MYCIN: aided physicians in treating bacterial infections

– approximately 500 rules for recognizing about 100 causes of infection

E.g.

IF

the type of x is primary bacteremia

the suspected entry point of x is the gastrointestinal tract

the site of the culture of x is one of the sterile sites

$THEN$

there is evidence that x is bacteroides

– certainty factors: numbers from $[0, 1]$ attached to conclusions to rank order

Recently, IBM Watson Health systems, but replaced by LLMs-based medicine systems

Qualitative physics

Qualitative physics (qualitative reasoning) concerns specifically with constructing a logical, non-numeric theory of physical objects and processes

Measure: the values of the properties that we assign for objects

$$Price(tomato) = \$(0.3)$$

$$\forall d.d \in Days \Rightarrow Duration(d) = Hours(24)$$

Composite object

Composite object: any object that has parts
E.g., $PartOf(taiwan, china)$

Schema (script): structure description
$\forall a Biped(a) \Rightarrow$
$\exists l_1 l_2 b Leg(l_1) \wedge Leg(l_2) \wedge Body(b) \wedge$
$Attached(l_1, b) \wedge Attached(l_2, b) \wedge$
$l_1 \neq l_2 \wedge \forall l_3 Leg(l_3) \wedge PartOf(l_3, a) \Rightarrow (l_3 = l_1 \vee l_3 = l_2)$

Various common knowledge of physics processes
say, water cycle

Structured descriptions

In FOL, all categories and properties of objects are represented by atomic predicates

– correspond to simple nouns, e.g., $Person$

– seem to be more like noun phrases, e.g., $MarriedPerson$

– – have an internal structure and connections to other predicates

e.g., a married person must be a person

These connections hold by definition, not by virtue of the facts we believe about the world

Need a way to break apart a predicate to see how it is formed from other predicates

Tradeoff between expressiveness and tractability

Reasoning procedures required for more expressive languages
may not work very well in practice

Tradeoff: expressiveness vs. tractability
E.g., Description Logics (see later)
– limited languages: between propositional language and first-order language with decidability
– vivid reasoning: easy to implement

Categories

Category: include as members all objects having certain properties
E.g., An object (penguin) is a member of a category (birds)
$penguin \in birds$

Subclass relations organize categories into a taxonomy (hierarchy)
E.g., a category is a subclass of another category
$tomatoes \in fruit$

Inheritance: the individual inherits the property of the category from their membership
E.g., $Child(x, y) \wedge Familyname(john, y) \rightarrow Familyname(john, x)$

The problem: natural kind or inheritance with exception
E.g., $\forall x. x \in Typical(bird) \Rightarrow Flies(x)$

Semantic networks

Inheritance is the result of (path-based) transitivity reasoning over paths in a network

– inheritance in trees
– inheritance in DAGs

- if-then reasoning in graphical form

- "does a inherit from b?" is the same as "is b in the transitive closure of $:IS\text{-}A$ (or subsumption) from a?"

Inheritance reasoning

Inheritance with exceptions

$$\forall rxb.Holds(r,x,b) \Leftrightarrow$$
$$Val(r,x,b) \vee (\exists px \in p \wedge Rel(r,p,b) \wedge \neg InterveningRel(x,p,r))$$

$$\forall xpr.InterveningRel(x,p,r) \Leftrightarrow$$
$$\exists i Intervening(x,i,p) \wedge \exists b' Rel(r,i,b')$$

$$\forall aip.Intervening(x,i,p) \Leftrightarrow (x \in i) \wedge (i \subset p)$$

Multiple inheritance

- credulous accounts choose arbitrarily
- skeptical accounts are more conservative

Description logics

Description logics (DLs): a family of logics with notations designed to describe the definitions and properties of categories

Subsumption: checking if one category is a subset of another based on their definitions

Classification: checking if an object belongs to a category

DLs focus on the tractability of inference and serve as theoretical foundation for ontology

Concepts, roles, constants

In DL, there are sentences that will be true or false (as in FOL)

In addition, there are three sorts of expressions that act like nouns and noun phrases in natural language

- concepts are like category nouns, e.g., $GraduateStudent$
- roles are like relational nouns, e.g., $:AreaOfStudy$
  (note ":" at start)
- constants are like proper nouns, e.g., $johnSmith$

These correspond to unary predicates, binary predicates and constants respectively in FOL

However, unlike in FOL, concepts need not be atomic and can have semantic relationships with each other

A description logic: language#

Symbols of DL:

- atomic concepts, e.g., $GraduateStudent$
- roles: (all are atomic), e.g., $AreaOfStudy$
- constants

Four types of logical symbols:

- punctuation: $[$, $]$, $($, $)$
- positive integers: $1, 2, 3, \cdots$
- concept-forming operators: ALL, $EXISTS$, $FILLS$, AND
- connectives: $\sqsubseteq$, $\doteq$, and $\rightarrow$

A description logic: syntax#

The set of concepts is the least set satisfying

– Every atomic concept is a concept

– If r is a role and d is a concept, then $[ALL\ r\ d]$ is a concept

– If r is a role and n is an integer, then $[EXISTS\ n\ r]$ is a concept

– If r is a role and c is a constant, then $[FILLS\ r\ c]$ is a concept

– If $d_1, \cdots, d_k$ are concepts, then so is $[AND\ d_1, \cdots, d_k]$

Three types of sentences

– If d and e are concepts, then $(d \sqsubseteq e)$ is a sentence

– if d and e are concepts, then $(d \doteq e)$ is a sentence

– If d is a concept and c is a constant, then $(c \rightarrow d)$ is a sentence

A description logic: semantics#

Constants stand for individuals, concepts for sets of individuals, and roles for binary relations

The meaning of a complex concept is derived from the meaning of its parts the same way a noun phrases is

– $[EXISTS\ n\ r]$ describes those individuals that stand in relation r to at least n other individuals

– $[FILLS\ r\ c]$ describes those individuals that stand in the relation r to the individual denoted by c

– $[ALL\ r\ d]$ describes those individuals that stand in relation r only to individuals that are described by d

– $[AND\ d_1, \cdots, d_k]$ describes those individuals that are described by all of the d_i

Formal semantics can be defined

A description logic: example#

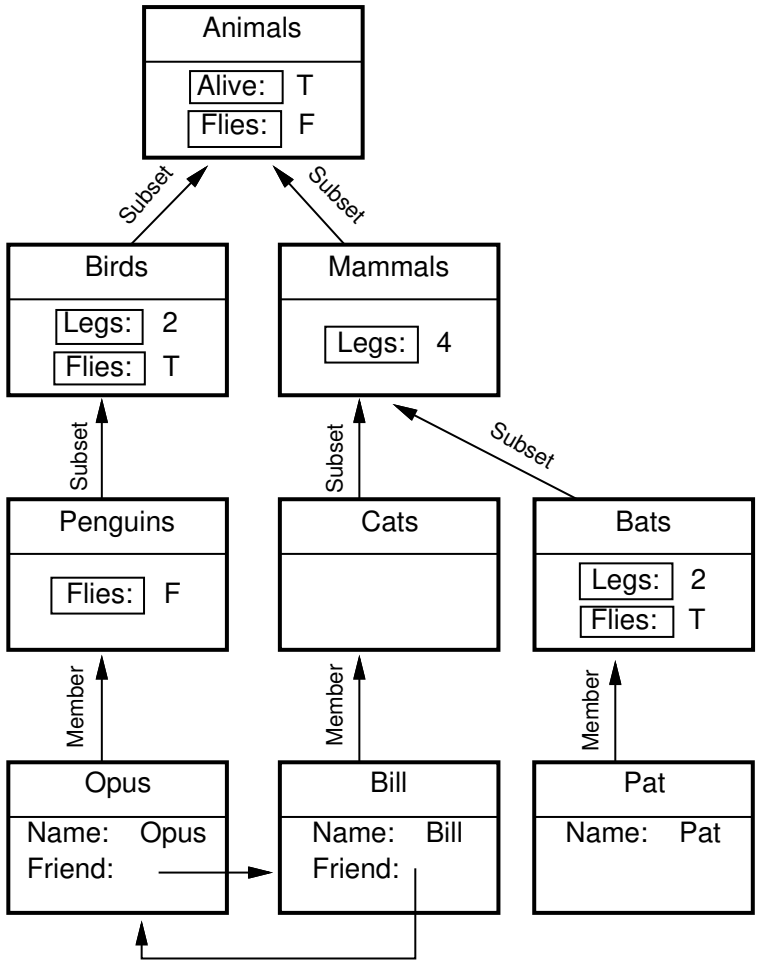
$$[AND\ \ Company$$
$$[EXISTS\ \ 7 : Director]$$
$$[ALL : Manager[AND\ \ Woman$$
$$[FILLS : Degree\ phD]]]$$
$$[FILLS : MinSalary\ \ \$24.00/hour]]$$

"a company with at least 7 directors, whose managers are all women with PhDs, and whose min salary is $24/hr"

A DL knowledge base is a set of DL sentences serving mainly to

- give names to definitions
- give names to partial definitions
- assert properties of individuals

Frame



(a) A frame–based knowledge base

Rel(Alive,Animals,T)
Rel(Flies,Animals,F)

Birds $\subset$ Animals
Mammals $\subset$ Animals

Rel(Flies,Birds,T)
Rel(Legs,Birds,2)
Rel(Legs,Mammals,4)

Penguins $\subset$ Birds
Cats $\subset$ Mammals
Bats $\subset$ Mammals
Rel(Flies,Penguins,F)
Rel(Legs,Bats,2)
Rel(Flies,Bats,T)

Opus $\in$ Penguins
Bill $\in$ Cats
Pat $\in$ Bats
Name(Opus,"Opus")
Name(Bill,"Bill")
Friend(Opus,Bill)
Friend(Bill,Opus)
Name(Pat,"Pat")

(b) Translation into first–order logic

Frame vs. logic

Link Type	Semantics	Example
$A \xrightarrow{Subset} B$	$A \subset B$	$Cats \subset Mammals$
$A \xrightarrow{Member} B$	$A \in B$	$Bill \in Cats$
$A \xrightarrow{R} B$	$R(A,B)$	$Bill \xrightarrow{Age} 12$
$A \xrightarrow{\boxed{R}} B$	$\forall x \;\; x \in A \;\Rightarrow\; R(x,B)$	$Birds \xrightarrow{\boxed{Legs}} 2$
$A \xrightarrow{\boxed{\boxed{R}}} B$	$\forall x \;\; \exists y \;\; x \in A \;\Rightarrow\; y \in B \wedge R(x,y)$	$Birds \xrightarrow{\boxed{\boxed{Parent}}} Birds$

Frame and semantic networks can be formalized by FOL, and hence can be thought of applications of FOL

Procedural knowledge#

Organizing procedural knowledge

knowing facts by executing code

Object-oriented (OO) representation

– with enough procedures/sentences in a KB, need to organize them

– in terms of objects

– – clustering procedures for determining properties, identifying parts, interacting with parts, as well as constraints between parts, all of objects

to make some things easier to find

Basic frame language

Frames are object structures

– individual frames: represent a single object, e.g., $person$

– generic frames: represent categories of objects, e.g., $students$

An individual frame is a named list of buckets called slots. What goes in the bucket is called a filler of the slot

$(frame\text{-}name$

$< slot\text{-}name1 \;\; filler1 >$

$< slot\text{-}name2 \;\; filler2 > \cdots)$

where frame names and slot names are atomic, and fillers are either numbers, strings or the names of other individual frames

Notation: attribute-value pair (AVP)

individual frames: $birds$

slot names: $: Fly$ (note ":" at start)

generic frames: $Animals$

IS-A and inheritance

Individual frames have a special slot: $INSTANCE\text{-}OF$ ($IS\text{-}A$)
whose filler is the name of a generic frame

$(birds$
$<: INSTANCE\text{-}OF \;\; Animals >$
$<: Fly \;\; T > \cdots)$

Similarly, generic frames have a slot: $IS\text{-}A$
whose filler is the name of another generic frame

Slots in generic frames can have attached procedures
– computing a filler (when no slot filler is given)
– propagating constraints (when a slot filler is given)

IS-A and inheritance#

Specialization relationships imply that procedures and fillers of a more general frame apply to more specific frame $\Rightarrow$ inheritance

Basic (local) reasoning with frames

– user instantiates a frame, i.e., declares that an object or situation exists

– slot fillers are inherited where possible

– causing more frames to be instantiated and slots to be filled

Object-oriented programming

specifying problems with frames is a style of **programming** rather than declarative object-oriented modeling of the world

Semantic Web

Semantic Web is the next generation of Web led by W3C (World Wide Web Consortium, `http://www.w3c.org`) that makes the Web pages understandable to machine

- RDF (Resource Description Framework) as underlying meta-data representational language is a language of categories (ontology)
- The "semantics" of Web (data and facts) is realized by ontology, OWL (Web Ontology Language) is an ontology representation language
- Various Knowledge Graphs are implementations of KB
- Description logics are theoretical foundation of ontology and the standards of ontology language
- The spirit of the semantic Web came from AI and can be viewed as an application of AI (so-called Internet + AI)

Resource description framework

RDF: W3C specifications as a metadata data model
- based on description logics
- syntax in XML (eXtensible Markup Language)

Triples: $subject$-$predicate$-$object$, or
(h,r,t) ($head$-$relation$-$tail$), entity-relationship (ER model)
- $subject, object$ (entities): the (web) resources
- $predicate$: the relationship between the subject and the object

RDF graph: a collection of RDF triples represents a labeled, directed multi-graph

SPARQL: query language for RDF graphs
- an SQL-like language

Web ontology language

OWL: a family of ontology representation languages
- based on description logics
- RDF/OWL syntax in XML (eXtensible Markup Language)

OWL extends RDF Schema
- Class equivalent
  Property
  sameIndividualAs
  ...
- RDFS
  subClassOf
  resource
  ID
  ...

Example: OWL

Define the terms "Camera" and "SLR", state that SLRs are a type of Camera

$$< owl : Class rdf : ID = \text{“}Camera\text{”} / >$$

$$< owl : Class rdf : ID = \text{“}SLR\text{”} >$$
$$< rdfs : subClassOf rdf : resource = \text{“}\#Camera\text{”} / >$$
$$< / owl : Class >$$

Knowledge graph

Knowledge graph (KG, or linked data, multi-relational data): representing entities and relations

Triples (h,r,t): $head$-$relation$-$tail$

– based on RDF and description logics

E.g. KG services to enhance search engine results with information gathered from a variety of sources

– Google knowledge panel, but replaced by LLMs

Change

Time:

E.g., $At(evening, sleep)$

Event:

E.g., $worldWarII, SubEvent(battleOfBritain, worldWarII)$

An event that includes as subevents all events occurring in a given time period is called interval

Space:

E.g., $In(beijing, china)$

$$\forall xl.Location(x) = l \Leftrightarrow At(x, l) \wedge \forall l_1 At(x, l_1) \Rightarrow In(l, l_1)$$

Process: liquid event

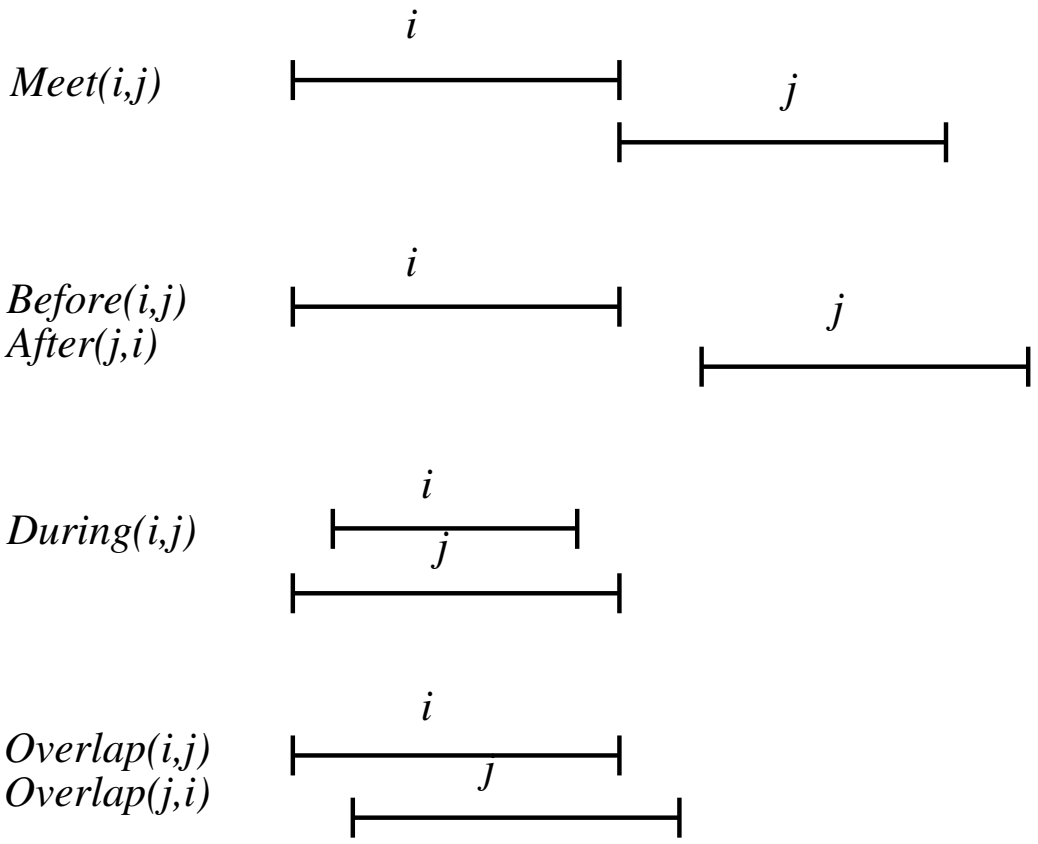
E.g., $T(working(teacher), todayLessonHours)$

$T(c, i)$ means that some event of type c occurred over exactly the interval i

Reasoning about change

Time interval

E.g., $\forall ij.Meet(i,j) \Leftrightarrow Time(End(i)) = Time(Start(j))$



Reasoning about change

Action

E.g., $\forall xyi_0.T(engaged(x,y),i_0) \Rightarrow$
$\exists i_1(Meet(i_0,i_1) \vee After(i_1,i_0)) \wedge$
$T(Marry(x,y) \vee BreakEngagement(x,y),i_1)$

Fluent: something that changes across situations

E.g., $President(USA)$
$T(democrat(president(USA)),AD2003)$

Context

E.g., $President(USA,45th) = DonaldTrump$

Explanation and diagnosis

Reasoning

– deduction, such as if α, $\alpha \Rightarrow \beta$, then β

– induction, such as α, β, then $\alpha \Rightarrow \beta$

– abduction, such as β, $\alpha \Rightarrow \beta$, then α

Abductive reasoning

– given $\alpha \Rightarrow \beta$, from β, abduce α

α is sufficient for β or

one way for β to be true is for α to be true

Can be used for causal reasoning: $(cause \Rightarrow effect)$

Diagnosis

One simple diagnosis uses abductive reasoning

KB has facts about symptoms and diseases

including: $(Disease \wedge Hedges \Rightarrow Symptoms)$

Goal: find disease(s) that best explain observed symptoms

Observe: we typically do not have knowledge of the form

$(Symptom \wedge \cdots \Rightarrow Disease)$

so the reasoning is not deductive

Non-uniqueness: multiple equally good explanations

Explanation#

Given KB, and β to be explained, we want an α s.t.

1. α is sufficient to account for β
   $KB \cup \{\alpha\} \models \beta$
2. α is not ruled out by KB
   $KB \not\models \neg\alpha$, i.e., $KB \cup \{\alpha\}$ is consistent
   otherwise, $(p \wedge \neg p)$ would count as an explanation
3. α is as simple as possible
4. α is in the appropriate vocabulary

Call such α an explanation of β w.r.t. KB

The simplest explanation is the negation of a clause with a minimal set of literal

Mental states

Propositional attitudes (modalities): e.g., know, believe, want, expect, etc.

Multi-agents: e.g., an agent reasons about the mental processes of the other agents

Formalizing reasoning about mental states

-syntactic theory

-possible worlds (modal logic)

Modal operators: B, K

$B(a, \psi)$ or $B_a(\psi)$: agent a believes that sentence ψ is true

$K(a, \psi)$ or $K_a(\psi)$: agent a knows that sentence ψ is true

$B(A, \psi), A = \{a_1, \cdots, a_n\}$: every agent of A believes that sentence ψ is true

A formal theory of belief#

Extending first-order language L:

Belief formulas: $Believes(Agent, fluent)$

Strings: $Flies(Clark)$ represented as $[F, l, i, e, s, (, C, l, a, r, k,),]$

– referential opaque: an equal term cannot be substituted for the one (mental object) in the scope of belief, e.g., $\text{“}Clark\text{”} \neq \text{“}Superman\text{”}$

Den function: mapping a string to the object that it denotes

$Name$ function: mapping an object to a string that is the name of a constant that denotes the object
E.g.,

$$Den(\text{”}Clark\text{”}) = ManOfSteel \wedge Den(\text{”}Superman\text{”}) = ManOfSteel$$

$$Name(ManOfSteel) = K_{11}$$

Belief theory#

Inference rules, e.g., Modus Ponens

$\forall apq.LogicalAgent(a) \wedge Believes(a,p) \wedge Believes(a, Concat(p, " \Rightarrow ", q) \Rightarrow Believes(a,q)$

where $Concat$ is a function on strings that concatenates their elements together, abbreviate $Concat(p, " \Rightarrow ", q)$ as $"p \Rightarrow q"$

E.g., belief rules: if a logical agent believes something, then it believes that it believes it

$$\forall ap.LogicalAgent(a) \wedge Believes(a,p) \Rightarrow Believes(a, "Believes(Name(a), p)")$$

Knowledge and belief

Logical omniscience:

$Believes(a, \phi), Believes(a, \phi \Rightarrow \psi) \models Believes(a, \psi)$

– So we need a limited rational agent

Belief and knowledge: knowledge is justified true belief

$\forall ap.Knows(a, p) \Leftrightarrow Believes(a, p) \wedge T(Den(p) \wedge T(Den(KB(a)) \Rightarrow Den(p))$

Belief and Time: $Believes(agent, string, interval)$

Knowledge and action: knowledge producing actions

Commonsense

KB

$\forall x Bird(x) \Rightarrow Flies(x)$

$Bird(Tweety)$

$KB \vdash Flies(Tweety)$??

With exceptions

$\forall x Bird(x) \wedge x \neq Penguin \wedge \cdots \Rightarrow Flies(x)$

$\forall x Bird(x) \wedge \neg Abnormal(x) \Rightarrow Flies(x)$

Commonsense reasoning: nonmonotonicity

Monotonicity of FOL

if $KB \vdash P$ then $(KB \wedge S) \vdash P$

i.e., if P follows from KB, then it still follows when KB is augmented by $TELL(KB, S)$

Nonmonotonicity: $KB \subset KB', \exists P, KB \vdash P$ but $KB' \nvdash P$

Nonmonotonic logics are the formalization of reasoning with incomplete knowledge

A solution to the frame problem and related problems

Commonsense reasoning: paraconsistency

Triviality of FOL

$\alpha \wedge \neg\alpha \vdash \beta$

i.e., everything follows from a single contradiction

Paraconsistency: $\{\alpha, \neg\alpha\} \subset KB, \exists\beta, KB \nvdash \beta$

Paraconsistent logics are the formalization of reasoning with inconsistent knowledge

Reasoning with incomplete knowledge

Closed World Assumption (CWA)

Let KB be a (finite) set of sentences (belief set), $T(KB)$ theory of KB is $T(KB) = \{\phi | KB \models \phi\}$

The CWA of KB, written as $CWA(KB) = KB \cup KB_{asm}$, defined as follows

1. $\phi \in T(KB)$ iff $KB \models \phi, \phi$ is a sentence
2. $\neg p \in KB_{asm}$ iff $p \notin T(KB), p$ is a ground atom
3. $\phi \in CWA(KB)$ iff $\{KB \cup KB_{asm}\} \models \phi$

Reasoning with incomplete knowledge

CWA

$KB = \{p(A), p(A) \Rightarrow q(A), p(B)\}$

$T(KB) \not\models q(B), T(KB) \not\models \neg q(B)$

$CWA(KB) \models \neg q(B)$

Problems

$KB = \{p(A) \vee p(B)\}$

$CWA(KB) \models \neg p(A) \wedge \neg p(B)$

Reasoning with incomplete knowledge#

Predicate Completion (COMP) (Negation-as-failure in Prolog)

$KB = \{p(A)\} \Leftrightarrow \forall x.x = A \Rightarrow P(x)$

I.e., "if" half of a *definition* for P

$\forall x.P(x) \Rightarrow x = A$

I.e., the *completion* formula for P

The completion of P in KB, written as $PC(KB; P)$, defined as follows

$COMP(KB; P) \equiv KB \wedge (\forall x.P(x) \Rightarrow x = A)$
$\forall x.P(x) \Leftrightarrow x = A$

Reasoning with incomplete knowledge#

Circumscription (CIRC)

Idea (Occam principle): the only objects satisfying the property P are those that must, given KB

Preferential semantics: minimality and minimal entailment $\models_m$

Let M_1, M_2 be two models. M_1 is less (preferential) than M_2, written as $M_1 \prec_P M_2$, if

1. $|M_1| = |M_2|$
2. $|M_1|_P \subset |M_2|_P$

Reasoning with incomplete knowledge#

CIRC

Let M be a model of KB. M is said minimal (preferential) iff there is no other models M' of KB such that $M' \prec_P M$

Define $KB \models_m \psi$ iff ψ is true in all minimal models of KB

Reasoning with incomplete knowledge#

Example KB

$\forall x Bird(x) \wedge \neg ab(x) \Rightarrow Flies(x)$
$Bird(Tweety)$
$Penguin(Tweety) \Rightarrow \neg Flies(Tweety)$

Set $P = \{ab, Penguin, Bird\}$. Then

$KB \models_m Flies(Tweety)$

Reasoning with incomplete knowledge#

Default logic (DL)

Default rule: an inference rule (at meta level)

$\frac{\alpha(x):\beta(x)}{\gamma(x)}$

E.g., $\frac{Bird(x):Flies(x)}{Flies(x)}$

Default theory $KB = (W, D)$: D is a set of default rules, W is a set of sentences

Extension of KB??

Reasoning with inconsistent knowledge

Maximal consistent subsets (MCS)

$MCS(KB)$ is the set of maximal consistent subset of KB

Reasoning with incomplete and inconsistent knowledge??

In what extent commonsense reasoning can be formalized??

The commonsense problem

Recall: What is common sense??

Commonsense is not explained, but

rely on our routines of behavior that we have learned over time

act in situations that are sufficiently unlike the routines we have seen before

Common sense is critical to human-level intelligence and AI

AI $\approx$ Commonsense

Neural representation

- Embedding representation
- Knowledge graph embedding
- Compression representation

Embedding representation

Recall: Word embedding $\Rightarrow$ Token embedding

The meaning of a token is defined as a vector in low dimensional space

– embedding: assign the vectors for representing tokens in a vector space

Vector semantics (distributional semantics) instantiates the DH (distributional hypothesis) by automatically learning **representations** of the meaning of tokens directly from their distributions in data in unsupervised ways

Embedding representation

Embedding representation: representing knowledge (say words) into low-dimensional (continuous) vector spaces

Contextualized embeddings: a model is pretrained to generate contextual **representations** of each token in a sequence, instead of just learning a token-to-embedding table

– mapping both a token and the surrounding context of tokens into a token embedding vector

LLMs for embedding representation

Knowledge embedding

Knowledge embedding representation (KER): representing knowledge into low-dimensional (continuous) vector spaces, so as to simplify computations preserving the inherent structure of the knowledge

E.g. fit any dataset with a (continuous, differentiable) scalar function with a single real-valued parameter

$$f_\alpha(x) = \sin^2\left(2^{x\tau} \arcsin\sqrt{\alpha}\right)$$

any dataset can be viewed as a list of numerical values $\mathcal{X} = [x_0, \cdots, x_n]$ describing the data content regardless of the underlying modality (time-series, images, sound etc.)

– say, animal shapes obtained with the different values of α

Ref. Boué L, Real numbers, data science and chaos: How to fit any dataset with a single parameter, arXiv, 2019.

Knowledge graph embedding

KG embedding (KGE): embedding entities and relations of a KG into low-dimensional vector spaces, to simplify computations preserving the inherent structure of the KG

Machine learning, especially deep learning

- Representing entities and relations.
  Entities are represented as vectors (deterministic points in the vector space)
- Defining a **scoring function**
  Defined each fact to measure its plausibility (facts observed in the KG have higher scores).
- Learning entity and relation representations (i.e., embeddings)
  Optimization problem that maximizes the total plausibility of observed facts

Distance-based embedding#

The relationships are represented as **translations** in the embedding space: if (h, r, t) holds, then the embedding of t should be close to the embedding of h plus some vector that depends on l

– i.e., the functional relation induced by the l-labeled edges corresponds to a translation of the embeddings

I.e., we want that $\overrightarrow{h} + \overrightarrow{l} \approx \overrightarrow{t}$ when(h, r, t) holds, while $\overrightarrow{h} + \overrightarrow{l}$ should be far away from $\overrightarrow{t}$ otherwise

There are various ways to embed knowledge into mathematical structures

Compression representation

Compression representation: the format used to store or transmit data in a more compact form than the original, for saving space, reducing transmission time, and improving efficiency

E.g., Lossless compression, JPEG, MP3/MP4, Zip, etc.

Write streams of data $x_{1:n} := x_1 x_2 \dots x_n \in \mathcal{X}^n$ of length n from a finite set of symbols $\mathcal{X}$

- $x_{\leq j} = x_{<j+1} := x_{1:j}$ for $j \leq n$
- the empty string as ϵ
- the concatenation of two strings s and r by sr

Coding distribution

A coding distribution P is a sequence of probability mass functions $P_n : \mathcal{X}^n \mapsto (0, 1]$

- for all $n \in N$ satisfy the constraint that $P_n(x_{1:n}) = \Sigma_{y \in \mathcal{X}} P_{n+1}(x_{1:n}y)$

for all $x_{1:n} \in \mathcal{X}^n$, with the base case $P_0(\epsilon) := 1$ (omit the subscript on P if no confusion)

The conditional probability of a symbol x_n given previous data $x_{<n}$ is defined as $P(x_n \mid x_{<n}) := P(x_{1:n}) / P(x_{<n})$

with the familiar chain rules

$P(x_{1:n}) = \Pi_{i=1}^{n} P(x_i \mid x_{<i})$ and

$P(x_{j:k} \mid x_{<j}) = \Pi_{i=j}^{k} P(x_i \mid x_{<i})$

Lossless compression

Lossless compression is to encode a stream of symbols $x_{1:n}$ sampled from a coding distribution P into a bitstream of minimal (expected) length while ensuring that the original data sequence is recoverable from the bitstream

Consider a binary source code $c : \mathcal{X}^* \mapsto \{0,1\}^*$

assigns to each possible data sequence $x_{1:n}$ a binary code word $c(x_{1:n})$ of length $\ell_c(x_{1:n})$ (in **bits**)

To minimize the expected bits per sequence $L := E_{x\sim P}[\ell_c(x)]$

i.e., encoding rare sequences with more bits and frequent sequences with fewer bits

Theorem (Shannon's source coding theorem, fundamental theorem of information theory): There is the limit on possible data compression as $L \geq H(P)$ for any possible code, where $H(P) := E_{x\sim P}[-\log_2 P(x)]$ is the entropy

Arithmetic coding

Arithmetic coding constructs a code with almost optimal length, given a coding distribution P and a sequence $x_{1:n}$

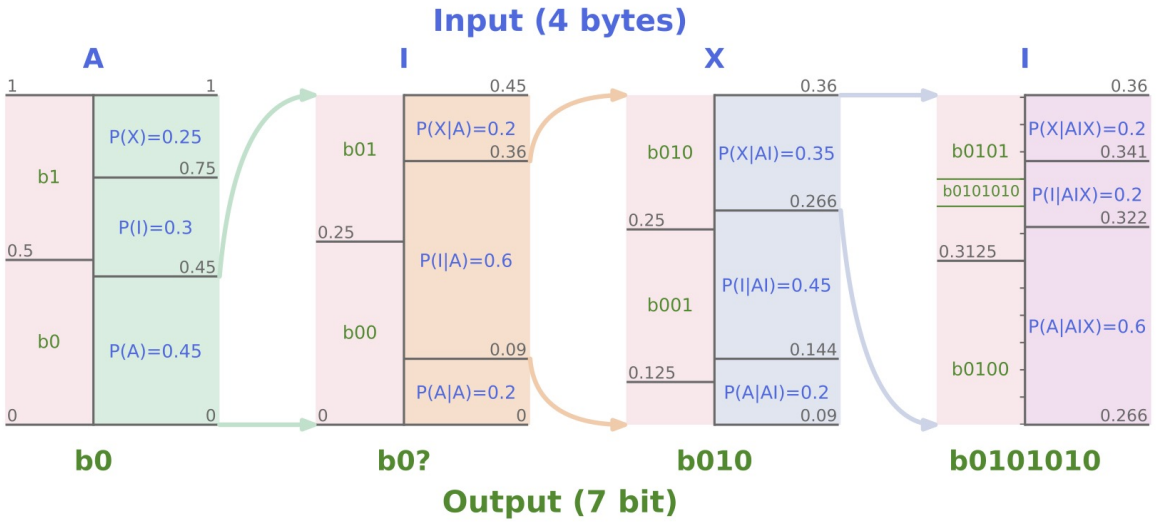
The connection between coding and compression with prediction and modeling

– Compressing well means modeling well in a log loss sense, and vice-versa

- Assuming infinite precision for the arithmetic operations involved, the arithmetic code has length $-\lceil \log P(x_{1:n}) \rceil + 1$ bits, whereas the optimal code length is $-\log P(x_{1:n})$ bits
- A practical implementation that is subject to B bit precision adds further $O\left(n2^{-B}\right)$ bits

– negligible for 32- or 64-bit arithmetic

Example: Arithmetic coding



Arithmetic encoding of the sequence 'AIXI' with a probabilistic model P (both in blue) resulting in the binary code '0101001' (in green)

Arithmetic codec#

Arithmetic encoder: The arithmetic code of a sequence $x_{1:n}$ is the binary representation of a number $\lambda \in [0, 1)$

- λ by narrowing down an interval that encloses λ step by step
  - Initially, this interval is $I_0 = [0, 1)$
  - In step $k > 0$ (i.e., encoding x_k), partition the previous interval $I_{k-1} = [l_{k-1}, u_{k-1})$ into N sub-intervals $\tilde{I}_k(x_1), \tilde{I}_k(x_2), \ldots$, one for each letter from $\mathcal{X} = \{x_1, x_2, \ldots, x_N\}$
  - The size of sub-interval $\tilde{I}_k(y)$ that represents letter y is $(u_{k-1} - l_{k-1}) \cdot P(y \mid x_{<k})$

To encode x_k, proceed with its corresponding interval, i.e., $I_k = \tilde{I}_k(x_k)$

Choose $\lambda \in I_n$ with the shortest binary representation in the terminating interval I_n and use that binary representation to encode $x_{1:n}$

Arithmetic codec#

Arithmetic decoder: Given λ and P decoding the k-th letter

- Starting with $I_0 = [0, 1)$
- Finding y s.t. $\lambda \in \tilde{I}_k(y)$ to decode $x_k = y$
- Set $I_k = \tilde{I}_k(x_k)$ and proceed with the k+1-st letter

Prediction-compression equivalence

Theorem: Minimizing the log-loss is equivalent to minimizing the compression rate of that model used as a lossless compressor with arithmetic coding

$\Rightarrow$ Predictive models (language models) can be transformed into lossless compressors and vice versa

Note: current language model training protocols use a maximum-compression objective

Prediction-compression equivalence

Compression-based sequence prediction: any compressor can be employed for sequence prediction

- Define $P(x_{1:n})$ as the coding distribution $2^{-\ell_c(\cdot)}$, where $\ell_c(x_{1:n})$ is the length of sequence $x_{1:n}$ when encoded with compressor c

- Recover the conditional distribution $P(x_i \mid x_{<i})$ by computing $2^{\ell_c(x_{<i})-\ell_c(x_{<i}x_i)}$, for all x_i

$\Rightarrow$ Any compressor (like gzip) can be used to build a conditional generative model

Hint: Information theory and machine learning are linked as "two sides of the same coin"

Universal coding#

Universal coding: Universal (optimal) source coding with respect to all computable sampling distributions can be achieved by choosing $\ell_c(x_{1:n})$ as the Kolmogorov complexity of $x_{1:n}$

- the conditional distribution described is universally optimal over $x_{<i}$, recovering the Solomonoff predictor (Bayesian mixture of all predictors that can be programmed in a chosen Turing-complete programming language)

Theorem: Compressing optimally is equivalent to predicting optimally and vice versa

LLMs as compressors

Prediction (LLMs) through the lens of compression as it encompasses generalization: a model that compresses well generalizes well

LLMs are general-purpose compressors
due to their in-context learning abilities

E.g., Chinchilla 70B achieves compression rates of 43.4% on ImageNet patches and 16.4% on LibriSpeech samples, beating domain-specific compressors like PNG (58.5%) or FLAC (30.3%), respectively

Compression and intelligence

Being able to compress well is closely related to intelligence

While intelligence is a slippery concept, file sizes are hard numbers

reducing the slippery concept of intelligence to hard file-size numbers

If you can compress the first 1GB of Wikipedia better than your predecessors, your (de)compressor likely has to be smart(er).

Hutter Prize (500'000€ prize for compressing human knowledge)
http://prize.hutter1.net/

to encourage the development of intelligent compressors/programs as a path to AGI

Compression≈intelligence??

Neurosymbolic representation

Neuro-symbolic AI (NSAI): Integration of neural and symbolic AI architectures to enhance reasoning and learning capabilities

Neurosymbolic representation aims to combine the strengths of neural representation and symbolic representation by integrating them into a unified representation

to leverage the learning capabilities of neural networks while also incorporating the knowledge, reasoning, and explainability of symbolic systems

- Limitations of neural networks: black-box nature, lack of interpretability, and difficulty in handling symbolic reasoning

- Limitations of symbolic reasoning: difficulties in handling uncertainty, scalability issues, and challenges in learning from data

Neuro-symbols

Neuro-symbols are the basic information processing units

A neuro-symbol is a structured representation that can encode symbolic knowledge and connect to other neuro-symbols, similar to how neurons are interconnected in neural networks

Neuro-symbols allow for the integration of symbolic and sub-symbolic information processing

Neuro-symbols can be hierarchically organized, with higher-level neuro-symbols representing more abstract concepts built from lower-level feature symbols extracted from raw data

Learning and reasoning

Neuro-symbolic networks can learn associations between sensory data and symbolic representations through a combination of neural network learning and symbolic reasoning

The learned neuro-symbolic networks can then be used for tasks such as perception, reasoning, and decision-making, combining the strengths of neural networks and symbolic systems

The symbolic component provides explainability and interpretability, while the neural component enables robust learning and generalization

E.g., KGE

Neuro-symbolics $\approx$ intelligence??