Language of Measurable Spaces for Natural Language Semantics

Jean-Philippe Bernardy, Rasmus Blanck and Aleksandre Maskharashvili

Oct 2019

Outline

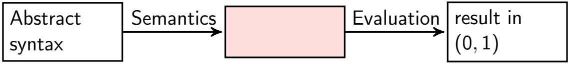
Motivation: Probabilistic Semantics

- Probabilistic reasoning has proven useful to model various linguistic phenomena (graded adjectives, pragmatics, etc.)
- Some believe it to occur in everyday life, events.
- Classical bayesian reasoning and vector models can be combined. Deep learning models have shown that individuals/situations and even predicates can be represented as points in a large-dimensional euclidean space (e.g. cosine distance). Hypothesis: Bayesian models can model such spaces.
- Intuitive probabilistic syllogisms can be accurately modeled.

Probabilistic Syllogisms

- Example 1
  - If you regularly eat humus, then you also enjoy tabouli.
  - Most people that enjoy tabouli insist on having mint tea with food.
  - If you eat humus, then you insist on having mint tea with food.
- Example 2
  - John is always as punctual as Mary.
  - Sam is usually more punctual than John.
  - Sam is more punctual than Mary.

Goal



- Solve a language-design problem
- Construct a (logic-style) language which is
  - sufficiently powerful to express probabilistic problems
  - convenient enough to support probabilistic syllogisms
  - has reasonable interpretations (models)
  - simpler than a full-fledged probabilistic programming language (which some authors advocate)

Example

- If you eat humus, then you also enjoy tabouli.
- Most people that enjoy tabouli insist on having mint tea with food.
- If you eat humus, then you insist on having mint tea with food.

Possible interpretation:

$$
\begin{aligned}
\Omega = [&eatHumus : Predicate;\\
&enjoyTabouli : Predicate;\\
&haveMintTea : Predicate;\\
&p1 : Most(z : [x : Ind; eatHumus(x)])enjoyTabouli(z.x);\\
&p2 : Most(z : [x : Ind; enjoyTabouli(x)])haveMintTea(z.x)]\\
X = &P_{\omega:\Omega}[Most(z : [x : Ind; \omega.eatHumus(x)])\omega.haveMintTea(z.x)]
\end{aligned}
$$

Main Idea

- Combine:
  - Rich types (functions, Σ-types)
  - Probability distributions
- A (measurable) space A is a type (or set, written $Set(A)$) equipped with an *integrator*.
- The integrator generalises the notion of integral/sum:
  $Integrate(x : A)t[x]$
  Integrates the expression $t[x]$ over the space A.

Base cases

Probability distributions can be interpreted as spaces.

1. Assume a discrete probability distribution P over a set S. We construct the space $\mathit{Discr}(P)$ as follows:
   - $\mathit{Set}(\mathit{Discr}(P)) = S$
   - $\mathit{Integrate}(x : \mathit{Discr}(P))t = \sum_{(x:S)} P(x) \cdot t$
2. Assume a continuous probability distribution over $\mathbb{R}$, with density function f. We construct the space $\mathit{Cont}(f)$ with:
   - $\mathit{Set}(\mathit{Cont}(f)) = \mathbb{R}$
   - $\mathit{Integrate}(x : \mathit{Cont}(f))t = \int_{(x:\mathbb{R})} f(x) \cdot t \cdot dx$

   Note that in our integrators, the bound variable is not repeated in the form of "dx".

Cartesian products and Σ spaces

1. If A and B are spaces, then $A \times B$ is a space.
   - $Set(A \times B) = Set(A) \times Set(B)$
   - $Integrate(z : A \times B)t =$
     - $Integrate(x : A) Integrate(y : B) t[(x, y)/z]$
2. If A is a space and B is a space, Σ(x:A)B is a space. Additionally, the variable x can occur in B.
   - $Set(\Sigma(x : A)B[x]) = \{(x, y) \mid x \in A, y \in B[x]\}$
   - $Integrate(z : \Sigma(x : A)B[x])t =$
     - $Integrate(x : A) Integrate(y : B[x]) t[(x, y)/z]$

Example:

- $A = \Sigma(\alpha : Uniform(2, 5))\Sigma(\beta : Uniform(2, 5))Beta(\alpha, \beta)$

It is convenient to use record notation for Σ types. The space below is isomorphic to the above example:

$A = [\alpha : Uniform(2, 5); \beta : Uniform(2, 5); x : Beta(\alpha, \beta)]$

Filtering: IsTrue(ϕ)

To represent evidence, we introduce the space $\mathit{IsTrue}(\phi)$, where ϕ is a Boolean-valued expression. $\mathit{IsTrue}(\phi)$ has a single element, which we will call $\diamond$, by convention.

- $\mathit{Set}(\mathit{IsTrue}(\phi)) = \diamond$

The density depends on the truth of ϕ:

- $\mathit{Integrate}(x : \mathit{IsTrue}(\phi))t = 0$ if ϕ is false
- $\mathit{Integrate}(x : \mathit{IsTrue}(\phi))t = t[\diamond/x]$ if ϕ is true

Filtering: IsTrue(ϕ), cont'd

Example:

- $A = \Sigma(lo : Uniform[0,1])\Sigma(hi : Uniform[0,1]) IsTrue(lo < hi) \times Uniform[lo, hi]$

We may sometimes omit *IsTrue* altogether and simply write the following for the same space:

- $A = \Sigma(lo : Uniform[0,1])\Sigma(hi : Uniform[0,1])(lo < hi) \times Uniform[lo, hi]$

Or, In record notation:

$$
\begin{aligned}
A = [lo &: Uniform[0,1]; \\
hi &: Uniform[0,1]; \\
p_1 &: lo < hi; \\
x &: Uniform[lo, hi]]
\end{aligned}
$$

Lemma: integrators are linear operators

Lemma:

- $Integrate(x : A)(k \cdot t) = k \cdot Integrate(x : A)t$
- $Integrate(x : A)(t + u) = Integrate(x : A)t + Integrate(x : A)u$

Proof: By induction on A, relying on the linearity of sums and integrals for base cases.
[pedantic: the underlying vector space is that of functions over Set(A) — the dimension of this space is #Set(A)]

Definitions: measure and expected value

The measure of a space (its total volume) is given by

- $\mathrm{measure}(A) = \mathit{Integrate}(x : A)1$

The expected value of $t[x]$ over $x : A$ is given by:

- $E_{x:A}[t[x]] = \frac{\mathit{Integrate}(x:A)t[x]}{\mathrm{measure}(A)}$

(One can say that x is a random variable sampled in A.)

Expected truth value

The number that we will be mostly interested in is the expected truth value of a formula $\phi[\omega]$, where ω is a world ranging in a space Ω. It is given by:

- $P_{\omega:\Omega}(\phi) = E_{\omega:\Omega}[\mathit{Indicator}(\phi)]$

If Ω is the space of possible worlds, then $P_{\omega:\Omega}(\phi)$ is the probability of ϕ. We have also:

- $P_{\omega:\Omega}(\phi) = \frac{\text{measure}(\Sigma(\omega:\Omega)\phi)}{\text{measure}(\Omega)}$

Back to example

$$
\begin{aligned}
\Omega = [&eatHumus : Predicate;\\
&enjoyTabouli : Predicate;\\
&haveMintTea : Predicate;\\
&p1 : Most(z : [x : Ind; eatHumus(x)])enjoyTabouli(z.x);\\
&p2 : Most(z : [x : Ind; enjoyTabouli(x)])haveMintTea(z.x)]\\
X =&P_{\omega:\Omega}[Most(z : [x : Ind; \omega.eatHumus(x)])\omega.haveMintTea(z.x)]
\end{aligned}
$$

What remains to do:

- define the space of Individuals and Predicates
- give a suitable definition for the "Most" quantifier

Individuals

Fortunately we have ways to interpret individuals as elements in a space, borrowed from machine-learning methods. The idea is simply to use a large dimensional vector space:
$Ind = Normal(0,1)^n$
With n sufficently big, depending on the complexity of the problem at hand.

Space of predicates: example

If an individual is represented by a vector x and a vector p represents a predicate, then x is said to satisfy the predicate if $p \cdot x > 0$. (Ie, both vector are oriented in the same direction in the underlying euclidean space.)

- $Predicate = \{\lambda x.p \cdot x > 0 \,|\, p : Normal(0,1)^n\}$
- Note: $Set(Predicate) = Ind \to Bool$

We deliberately restrict the space of possible predicates to make ranging over it meaningful. (There are too many functions to pick a meaningful "random" one).
If words can be represented by a vector, then so can predicates (hopefully). Again this idea comes from machine-learning methods.

Most

Thanks to the probabilistic setting, we can interpret generalized quantifiers. (Most, Few, etc.). We define:

- $\mathit{AtLeast}\,\theta(x : A).\phi \triangleq \text{measure}(\Sigma(x : A)\phi) > \theta\, \text{measure}(A)$

Then we can interpret "Most cn vp" as $\mathit{AtLeast}\,\theta(x : [\![cn]\!])([\![vp]\!]x)$
This is possible be cause measures are internalised in the language of propositions.

Conclusion

- Supports many phenomena
- Probabilistic reasoning works
- Arguably more convenient than probabilistic programming
  - Better match with logic/type theories
  - More straightforward semantics
  - Formally more powerful than probabilistic programming (by internalising the notion of measure/expected value/probability)

For completeness: Morphing spaces

The idea is to map the space of vectors to a (sub)space of predicates. How to do this? We need to extend our language of spaces with the construction $\{e \,|\, x : A\}$, for any space A, with the semantics:

1. $\mathit{Itegrate}(z : \{e[x] \,|\, x : A\})t[z] = \mathit{Integrate}(x : A)(t[e[x]])$
2. $\mathit{Set}(\{e[x] \,|\, x : A\}) = \{e[x] \,|\, x : \mathit{Set}(A)\}$

Note that we do not change the density when integrating – there is no need to compensate for a non-uniformity in e.

Universal Quantifiers

It is natural to add the construction $\forall x : A.\phi$ to propositions, with the following definitions:
$\forall x : A.\phi \triangleq \mathit{AtLeast}\, 1(x : A)\phi \triangleq \text{measure}(A) \leq \text{measure}(\Sigma(x : A)\phi)$

Pitfalls

Assume

- $A = [-1..1]$ and
- $\phi = (x \neq 0)$

We then have:

$$\text{measure}(A) = 2$$
$$\text{measure}(\Sigma(x : A)\phi) = 2$$

And according to the above definition:
$\forall x : A.\phi = \mathit{true}$
(So this operator really means "for almost all" in probabilistic logic)

Dealing with this pitfall

- define a more precise measure that counts single elements
  - not computable, because HOL is undecidable
- use "soft transitions" (continuous interpretations of propositions)
  - still does not make $\forall x : A.\phi$ coincide with the usual definition (but can help with the approximation algorithms in many cases.)
- do not use problematic domains
  - this is what we do.
  - (for example use dirac delta for equalities)

probability density/mass functions

We can define a generic notion of probability distribution over the spaces defined as above.
Let's first define $G[A](x, y)$ with the idea that $G[A](x, y) = 1$ if $x = y$, 0 otherwise.
By induction:

$$\begin{aligned} G[\mathrm{Distr}(d)](x, y) &= \delta(x - y) \\ G[\mathrm{Distr}(d)](x, y) &= \mathit{Indicator}(x = y) \\ G[\mathrm{IsTrue}(\phi)](x, y) &= 1 \\ G[\Sigma(z : A)B]((x, y), (x', y')) &= G[A](x, x') \cdot G[B](y, y') \end{aligned}$$

Then the Probability (mass) distribution over A is given by:

- $P_A(x) = E_{y:A} G[A](x, y)$

Note that if A is continuous, the argument of $G[A](x, y)$ is integrated, so δ always occurs under an integral.