Abduction feature relationship with logic

1 Problem

Input:

- Training data: $(\boldsymbol{x}_i, y_i)$, x is feature, $y = 1$ or 0 is label (maybe multiple labels)
- Background knowledge:
  - BK in domain, e.g.:
    - Domain has concepts `con1,con2,...`
    - A concept `con_i` only affect features of indices `[f_1,f_2,...]`.
    - some instances are constructed from some specific concepts: e.g. `inst_i` has concept `con_j`.
    - relations between sub-concepts, e.g.:
      exlcusiveness: `exclusive(con_i,con_j)`,
      cooccurance: `co_occur(con_i,con_j)`,
      spatial relationship: `left_of(con_i,con_j)`,
      ...
  - Meta knowledge:
    - template (meta-rules) for rule abduction; knowledge of constructing meta-rules. For example, `father(con_i,con_j)`→`metarule(con_i(X):-con_j(X),R(X))`, meaning if concept `con_i` is father of concept `con_j`, then every rule about `con_i` must include `con_j` in its body.
    - primitives of statistical models, e.g.:
      `cluster(Data,Num_Centeres,Partition)`,
      `sparse_coder(Data,Dict)`,
      `stat_classifier(Data,Labels,Model,Predictions)`,
      `data_feature_filter(Data,Feature_Idxs,New_Data)`,
      `data_instance_filter(Data,Data_Idxs,New_Data)`,
      ...

Output:

- A set of rules as hypothesis for target concept `label`.
- A set of rules to describe relations between concepts in domain (for transferability?)

2 Intuition

The number of relations is exponential to the number of concepts, each concept is affected by plenty of feature, so there will be a huge number of relations in feature space, and most of them are unimportant. **Using backgound knowledge to abduce feature relationships enables us to focus on relationships that only related to background knowledge, which are more important and comprehensible to the users**.

2.1 Main Idea

Example:

When learning the concept of automobile, we have background knowledge that there are 2 sub-concepts `A` and `B` belong to `automobile`, and relation that `A` always lies on the bottom of an `automobile` and just below (and connected with) `B`. After showing some examples of `automobile`, one can easily construct a valid hypothesis that `A` should be wheel, `B` should be car's body.

Moreover, this hypothesis will also be helpful when this person is trying to understand the concept of bicycle, since the knolwedge of `A` and `B` can be direclty transfered.

Background knowledge $\overset{\spadesuit}{\Longrightarrow}$ Meta-rules (template rules) $\Longrightarrow$ Statistical Models learning.

$\overset{\spadesuit}{\Longrightarrow}$ means "abduce"

- Incorporating first-order background knowledge by letting them to control the learning of statistical classifiers. The meta-rules act as templates and enables predicate invention, and the invented "predicates" are statistical classifiers.
- Statistical learning is directly controled by its training data, which is actually controlled by rule induction by variable bindings in primitives from background knowledge.
- Reuse the learned classifiers as logical predicates. The invented predicate (for exapmle the `R` in last paragraph) might be an useful mid-level concept that can help the learning of other labels.

2.2 Main Challenge

Semantic Gap: Relations in human knowledge are mostly defined in a semantic space (between high-level concepts). However, in statsitcal learning, most data are located in a primary low-level feature space, which is far from semantic meaningful. *Semantic gap* is the difference between human possessed knowledge space and statsitcal learning data's feature space. **For the above "automobile's wheels and body" example, the challenge is how to split "wheels" and "car body" in the feature space and understand they are two different objects that may have relationship.**

Typically, most datasets are collected in a statistical way, i.e., in each dimension of a feature vector, the values are statistics of each attribute. For example, TFIDF/bag of words features records the number of appearance of a word. This kind of features only preserve appearance of attributes, so only numerical and statistical relations can be defined, e.g. `>`, `<`, `co_occurance`, etc. Furthermore, extracting low-level features may cause loss of many semantic relations. Such as in computer vision domain, after feature extraction (HOG, SIFT, CNN, etc.), images are encoded into vectors, the loss of spatial information causes the incapability of spatial relations like `above` and `below`.

To solve the semantic gap problem and implant background knowledge in statistical machine learning process, we may have 2 kinds of solutions: 1) design background knowledge in mathematical & statistical language and make it applicable on statistical feature space; 2) induce high-level concepts from statistical feature space and direclty apply ordinary human knowledge.

2.3 Possible ideas

2.3.1 [Deprecated] symbols as graphical model variables (Generative)

The **structure of feature space** is assumed like this:

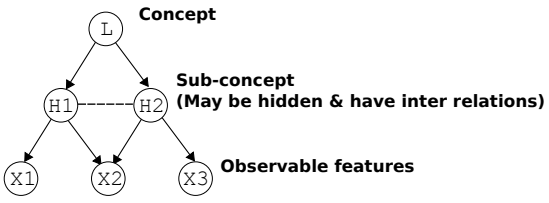

Figure 1.

Data $(\boldsymbol{x}_i, y_i)$ is generated by a probabilistic model, where $\boldsymbol{x} = x_1, ..., x_n$ is feature vector, $y \in \{0,1\}$ is label. For example in Figure 1, suppose there is only one *target concept* (label) `L`, it has 2 *sub-concept*s `H1` and `H2`. Each sub-concept can be regarded as a *latent label*, target/sub concepts controls the generation of each *feature* X_i. In the example above (if internal sub-concepts have no connections), the generative model is:

$$P(L, H1, H2, X1, X2, X3) = P(L)P(H1|L)P(H2|L)P(X1|H1)P(X2|H1, H2)P(X3|H2)$$

In order to learn this structure, certain background knowledge (even some annotations) is needed. For example, oracle can provide evidences of which features does the concept controls; or annotations that tells which concept does an instance belongs to.

The key idea of this model is "**target concept is a combination of its sub-concepts, and each (sub-)concept only affects a subspace in original feature space**". The knowledge of concepts and corresponding feature space are given in first-order logical form.

PROBLEM:

If a feature is generated from multiple concepts, the joint probability would be difficult to model (like the X_2 in the above example). **We need strong heuristics as prior to model the conditional probabilities.** For example mixture gaussian or additivity [A. Anandkumar et. al., 2013, *Learning Linear Bayesian Networks with Latent Variables*].

REASON OF GIVING UP THIS IDEA:

This algorithm doesn't work well because I hardly cand find a general way to model conditional probability well (even on artifitial data); The feature structure is restricted to `is_a(A,B)`, `exclusive(A,B)` kind of relationships.

2.3.2 [Undergoing] symbols as feature dictionary (Discriminative)

There is no specific assumption on the structure of domain. A concept is considered as a vector (or a set of vectors) in the original feature space. For example, handwritten characters and numbers are constructed by pen strokes, each stroke could be seen as a sub-concept of a character and represented by a vector of pixels. When writing a character, human is writing strokes follow a certain convention, e.g. a fixed orders and fixed patterns. The orders and patterns are structures in the feature space of pen strokes. Popular feature learning techniques do not consider this structure, most extracted features are scattered and meaningless, which makes the structure unclear.

A straight forward idea is using sparse coding to learn a dictionary from the original feature space, then use background knowledge to examine whether does the learned symbols (item in dictionary) satisfy the relational constraints in backgound knowledge. The main problem is that the dictionary learned by sparse coding strongly depends on its initialization (sampled data patches and initial values of dictionary), we cannot guarantee the learned dictionary contains the desired logical symbols in background knowledge. In short, the smantic gap will result in the difference between the learned dictionary and the logical symbols in background knowledge. (Although all sparse coding and autoencoder works claim that the learned dictionary presumes semantical meaningful concepts, but most of them are still too primary to form first-order logic relations, and the appearance of those concepts in learned dictionary cannot be guaranteed).

To solve this problem, we propose a repeated hypothesis then revise procedure. Main steps:

1. Sample a sub-part from each data;
2. Use the sub-parts as training data patches to train a sparse coder;
3. Use the trained sparse coder to reconstruct original data;
4. Get the difference between orignal data and reconstructed data;
5. Use the difference and certain background knowledge to resample sub-parts as data patches, goto step 2.

Conclusion:

This is a **unsupervised(?)** learning framework (no labels, but with other kind of backgound knolwdges), it can learn dictionary whose items follows a structure defined by logical language.

3 Language to describe relations in feature space

In this section we introduce the language for describing background knowledge in a binary classification problem.

- Primitives:
  - `concept(Con,Vector)`: A concept is a vector (basis) in feature space
  - `Relation(Con1,Con2)`: `Con1` and `Con2` has certain relations, e.g.,
    - `move_left(Con1, Con2)` means `Con2` can be obtained by moving `Con1` to the left (on a image).
    - `exclusive(Con1,Con2)`: `Con1` and `Con2` never appear together.
    - `co_occur(Con1,Con2)`: `Con1` and `Con2` always occur together.
  - ... ([Probabilistic] label relation constraints, to be added)
  - Statsitical models:
    - `sparse_coder(Data, Param, Model, Code)`: Learn a sparse coding dicitionary (stored in `Model`), and encode original data with the learned dictionary.
- Target Hypotheses: After learning, we want to obtain hypothesis like this:
  - A `Model` defining coding dictionary $d_1, d_2, ...d_D$
  - A set of facts that describing feature relationship: `rel(d_1, d_3),...`

4 MNIST example

Using 1×748 original pixel directly (not sampling 14×14 windows on original data, which can only learn small strokes having no semantic relations)

- If we do not include background knowledge, the learned dicts are whole picture of numbers themselves. [done]
- If we use the proposed method, we can use spatial background knowledge (a number could be split to "up and down" two parts, etc.), it should return a dictionary with number structures (e.g. "9" is composed by a small "o" on the up and a line or a curve down it) [doing]