

Machine learning sangat erat kaitannya ke kecerdasan (intelligence) adalah belajar(learning). Pada kenyataannya, kecerdasan tidak ada tanpa kemampuan untuk belajar karena belajar ini berarti kebutuhan akan pengetahuan baru. Learning memperbolehkan untuk beradaptasi ke dan menggunakan keuntungan berbagai situasi dan kejadian sehingga kemampuan learning menjadi suatu perangkat atau alat yang sangat berguna.

Ada sebuah pendapat yang saling kontradiksi di mana penerapan komputer untuk dapat belajar adalah: sangat mudah dan sangat sulit. Pendapat tersebut berdasarkan adanya dua perbedaan yang mencolok dari learning: 1) rote learning, 2) cognitive learning.

Dari dua perbedaan di atas, kemungkinan besar timbul beberapa tipe atau jenis lain dari learning seperti: 1) learning by analogy, 2) learning by example, 3) learning by observation.

Namun demikian, seluruh tipe learning dapat disimpulkan bahwa pengetahuan-pengetahuan yang ada dapat dibutuhkan atau tidak ke mekanisme sesungguhnya di mana memperbolehkan terjadinya learning.

Learning by Rote, Banyak yang telah kita pelajari membentuk kenyataan di mana kita harus mengulang atau mengingatnya. Proses ini disebut: Learning by Rote.

Rote learning tidak hanya terbatas pada kenyataan (fact), tetapi dapat diterapkan ke rangkaian kegiatan-kegiatan.

Rangkaian yang diingat adalah: 1) mengambil kotak-kotak merah dan kuning, 2) meletakkan ke suatu tempat.

Inti dari rote learning adalah: spesialisasi di mana segala sesuatu yang dihafal adalah spesifik dan rangkaian langkah-langkah yang membentuk proses tidak dapat digeneralisasi. Rangkaian penghafalan ini disebut: prosedur. Sebuah komputer dapat dengan mudah mempelajari sesuatu yang kita hafalkan, karena komputer diprogramkan.

Rote Learning melibatkan: pengingatan baik kenyataan atau prosedur-prosedur dan tidak membutuhkan sesuatu generalisasi yang dirinci atau pemikiran tingkat tinggi.

Cognitive Learning, Suatu cara yang paling penting dalam belajar adalah: cognitive learning di mana cara ini paling sulit diterapkan. Dalam bentuk learning di sini, kita harus menggunakan alasan untuk: menganalisa, mengorganisasi, dan menghubungkan bagian-bagian tertentu pengetahuan. Cognitive learning sangat menakjubkan di mana kita dapat belajar sebuah generalisasi.

Kita dapat membentuk deskripsi kelas untuk prosedur-prosedur, kemudian kita dapat beradaptasi terhadap prosedur-prosedur yang tergeneralisasi ke berbagai situasi sejenis. Proses ini adalah kemampuan untuk menggeneralisasi prosedur yang membedakan manusia dari robot.

Sebuah robot hanya dapat mengenal proses tertentu, tetapi tidak dapat menggeneralisasi. Kemampuan kita untuk belajar sebuah klasifikasi tidak terbatas pada obyek atau prosedur, tetapi dapat diterapkan pada ide dan konsep.

Kemampuan untuk mempelajari deskripsi kelas adalah dasar pembentukan sebuah komputer di mana berfikir seperti cara manusia lakukan.

Kesulitan terbesar untuk menangani bila mencoba untuk membentuk sebuah intelligent computer, di mana kenyataannya bahwa kita sendiri memiliki sedikit pengertian tentang proses pemikiran manusia. Pada umumnya, kita tidak memiliki ide bagaimana kita dapat membentuk deskripsi kelas tergeneralisasi dari obyek tertentu. Namun demikian tentunya percobaan untuk menemukan sebuah cara untuk menggantikan tipe learning ini dalam sebuah komputer, kadang melihat ke dalam pemrosesan pemikiran manusia.

Prof. Patrick Henry Winston, direktur laboratorium AI di MIT membuat solusi umum yang mengarah ke pengertian cognitive learning. Metoda learningnya tentang deskripsi kelas disebut: "hit-and-near-miss".

Deskripsi kelas gapura:

1) harus memiliki sebuah blok atau cilinder pada atapnya dari dua blok lain yang tidak bersentuhan.

2) blok pendukung kemungkinan berdiri tegak atau tergeletak.

Procedure 'hit-and-near-miss' memiliki 2 asumsi:

1) Terdapat petunjuk yang menampilkan salah satu contoh bagian dari kelas atau 'near-misses', dan tidak pernah salah tentang contoh-contoh tersebut.

2) Contoh peralatan harus contoh valid karena contoh tersebut satu dari bentuk-bentuk model awal.

Dua prinsip penting yang harus dilibatkan dalam procedure 'hit-and-nearmiss' untuk meningkatkan efisiensi dan keandalannya, yaitu:

1) Disebut 'no-guessing' dilibatkan, ketika komputer tidak dapat melihat perbedaan antara model yang benar dengan model yang tidak benar, yaitu bila terjadi suatu pengakuan terhadap obyek yang tidak, maka sesungguhnya komputer tidak mempelajari sesuatu.

2) Disebut: 'no-altering', jika petunjuk mendukung obyek yang benar, tetapi gagal untuk memadankan ke definisi yang ada, maka komputer membentuk secara terpisah klasifikasi kasus khusus dari pada mencoba untuk memperbesar klasifikasi yang ada untuk mencakup obyek tersebut.

Masalah sekunder yang juga menjadi bahan pertimbangan adalah: bagaimana caranya untuk menyimpan pengetahuan yang dibutuhkan ke dalam komputer baik pengetahuan alami dan pemilihan pemrograman menentukan cara di mana pengetahuan dapat ditunjukkan.

Teknik-teknik untuk mengembangkan lapangan knowledge representation: Trees, Lists, Networks.

Trees, Bila kita berpikir kembali pada metoda backward chaining dalam expert system, dapat dilihat dengan jelas bahwa cara paling efisien untuk menunjukkan pengetahuan untuk tipe expert system adalah tree. Kemajuan expert system melalui tree, akan menyingkat bagian besar dan menemukan goal secara cepat.

Expert system yang menggunakan tree pengetahuan akan selalu menanyakan user-nya untuk menjawab pertanyaan yang relevan. Secara alami, tree adalah hirarki, sehingga kemungkinan hanya digunakan untuk menyimpan pengetahuan hirarki, sehingga tree bukan merupakan batasan yang baik, karena banyak pengetahuan gagal ke dalam kategori yang berada dalam tree. Gambar berikut ini, tree digunakan untuk menyimpan informasi tentang persegi-empat, segitiga, dan trapesium.

Kerugian terbesar dari penggunaan tree: Kesulitan pembentukan dan pemeliharaan tree sehingga menurunkan efisiensi.

List penting untuk keberhasilan pemrograman Turbo Prolog, dimana list selalu menggunakan metoda tradisional AI untuk knowledge representation.

Contoh: Bahasa AI pertama LISP (LISt Processing) dirancang untuk menangani list secara efisien. Fungsi dari list, sehingga sangat menarik untuk knowledge representation adalah: mudah dilakukan (dikerjakan) dalam prolog. Untuk mengerti bagaimana knowledge ditunjukkan dalam list dapat diberikan.

Walaupun list hanya dapat diproses secara sequential, tetapi list sangat penting karena kita dapat menggunakannya untuk menunjukkan sesuatu dan seluruh tipe knowledge. List sangat fleksibel di mana membuat list sangat penting untuk AI sebagai alternatif utama. Dengan menggunakan berbagai skema index, kita dapat membuat list hampir seefisien tree.

Network, Representasi knowledge sebagai sebuah network adalah menggunakan dan sangat berguna, juga bentuk network sangat kompleks. Representasi knowledge dimungkinkan di mana representasi network dari knowledge akan distandarisasi.

Network Knowledge didasari pada 2 kondisi:

- 1) Knowledge dalam network ditunjukkan oleh node-node dalam sebuah graph non hirarki, tidak seperti tree, semua node dalam network memiliki kepentingan yang sama dan salah satu dari node-node tersebut dapat digunakan sebagai posisi awal.
- 2) Node-node diatur sehingga tipe knowledge sejenis dikelompokkan dekat satu sama lainnya, yaitu adjacent node memiliki hubungan 'near-miss'.

Kita mengakses model network dengan memasukkan network pada sebuah posisi yang tepat dan kemudian memproses sepanjang sampai node sesungguhnya dicapai.

Secara teori, metoda ini seharusnya sangat efisien, karena setiap perpindahan ke node baru dibuat karena transisi(perpindahan) berada pada arah pengembangan yang sejenis, di mana merupakan sebuah bentuk hill climbing. Walaupun prosedur ini akan bekerja tanpa masalah di mana kita masuk ke dalam network, tetapi ada baiknya untuk masuk pada sebuah node yang agak mendekati goal. Kebanyakan model-

model network juga berisi list index yang membantu memilih node masukkan untuk setiap situasi.

Kejadian terburuk: Jika tiba-tiba kita memilih node yang jauh dari sejenis, maka network mengacak ke dalam list. Keuntungan metoda network dari knowledge representation adalah metoda ini dapat dengan mudah dan efisien menangani baik knowledge hirarki dan nonhirarki. Knowledge nonhirarki akan mencoba dari sebelah luar batasan network, dengan knowledge hirarki yang dilokasikan lebih dekat dengan pusat.

Representasi Knowledge (Pengetahuan), Knowledge adalah segala sesuatu yang digunakan oleh pemecah persoalan untuk memecahkan persoalan. Understanding: kemampuan menggunakan knowledge(pengetahuan) kita untuk memproduksi (meramal) efek-efek dari suatu kegiatan (aksi) di dunia atau untuk memproduksi hasil dari suatu observasi.

Hal-hal yang harus dipertimbangkan atau ada dalam knowledge: Objek (deskripsi, hak-hak/kewajiban, operasi-operasi), Kejadian-kejadian, Meta-knowledge: knowledge about knowledge (mengetahui apa yang anda tahu atau tidak tahu).

Prosedur-prosedur knowledge (procedural Knowledge) Prosedur-prosedur tentang bagaimana menyelesaikan persoalan dan bagaimana melaksanakan pekerjaan tersebut. (misal: bagaimana naik sepeda, bagaimana menerbangkan pesawat, dan sebagainya). Proses-proses Knowledge: Acquisition, mendapatkan dan mengintegrasikan knowledge; retrieval, mencari knowledge-knowledge yang relevan; inference/reasoning, mencari knowledge-knowledge baru dari knowledge ke dalam knowledge.

Tipe-Tipe Representasi:

Analogue representations (representasi-analog) representasi internal secara struktur sama dengan sesuatu yang sedang direpresentasikan;

Table-table data: Relational Data Base, Samuel Rate Learning Experience;

Logic; Production Rule; Procedural Representations; Semantik Network: Network Data Base; Frame, Scripts.