1. 인공지능의 정의

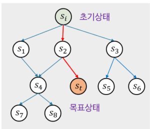
- 튜링테스트: 사람처럼 행동하는가?
- AI 파라독스: 어제의 AI로 간주되던 것이 오늘은 AI가 아님

2. 탐색

1. 맹목적 탐색(전체 탐색): 깊이 우선탐색, 너비 우선 탐색, 반복적 깊이 심화 탐색, 양방향 탐색

깊이 우선 탐색
- (단점) **최단 경로** 해 탐색 보장 불가
- (장점) 메모리 공간 사용 효율적

너비 우선 탐색
- (장점) 최단 경로 해 탐색 보장
- (단점) 메모리 공간 사용 비효율

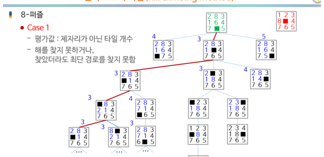


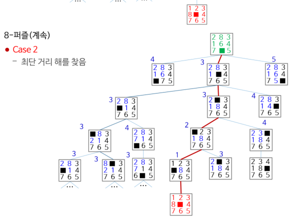
반복적 깊이심화 탐색
- 최단 경로 해 보장
- 깊이우선과 너비우선의 장점을 모두 가짐
- 반복적인 깊이 우선 탐색에 따른 비효율성은 낮음
  - 노드별10개의 자식노드를 가정할 때, 너비 우선 탐색 대비 약 11%의 추가 노드 생성

양방향 탐색
- 너비 우선 탐색 장단점 모두 유지
- 상대적으로 복잡

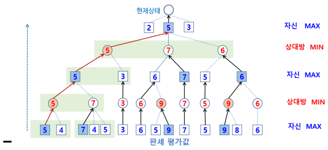
2. 정보 이용 탐색(부분 탐색, **휴리스틱 탐색**): 최상 우선 탐색, 빔 탐색, A^* 알고리즘
   - **언덕 오르기 탐색: 국소 최적점(local optima)에 빠질 위험 존재**

언덕 오르기 기법(hill climbing method)

1. 

2. 

3. 게임 탐색(상대적 탐색):
   - **mini-max 알고리즘: 지정된 깊이의 게임트리를 제작**, 단말 노드부터 위로 올라가면서 연산을 반복해 **판세 계산**



   - **α-β 가지치기**: 검토할 필요 없는 노드 탐색에서 제외
   - **몬테칼로 트리 탐색**

➡ 1) 반복적 깊이 심화 탐색은 탐색 트리의 레벨을 증가하면서 루트 노드부터 해당 레벨까지의 부분 트리(서브트리)에 대해 너비 우선 검색을 수행한다.

2) 탐색 공간 전체를 차례로 검사하는 방식이다.

3) 너비 우선 탐색 기법은 탐색공간 트리를 레벨별로 차례로 검사하므로 반드시 해를 찾을 수 있다.

4) 깊이 우선 탐색 기법은 한 노드를 검사한 이후에는 그 노드에 연결되어 있는 노드들 중에서 아직 검사하지 못한 노드를 선택하여 검사한다.

정보 이용 탐색에 관한 설명 중에서 맞지 않는 것은?

➡ 1) 탐색 공간 전체를 차례로 검사하는 방식으로 과정의 각 상태에서 추출할 수 있는 정보를 활용하여 적절한 검사 대상 노드를 선택한다.

2) A* 알고리즘은 휴리스틱 평가값 결정에 이미 투입된 비용과 남은 비용이 함께 고려된다.

3) 빔 탐색은 최상 우선 탐색을 기반으로 하되 각 레벨에 정해진 개수의 노드만 탐색한다.

4) 언덕 오르기 탐색은 전역 최적해(global optima)를 항상 찾을 수 있는 것은 아니다.

다음 중에서 게임 트리 탐색이 아닌 것은?

➡ 1) A* 알고리즘

2) mini-max 알고리즘

3) 몬테칼로 트리 탐색

4) α-β 가지치기

다음 중에서 게임 트리 탐색에 관한 설명 중에서 맞지 않는 것은?

➡ 1) 게임 트리 탐색에서 항상 '나' 또는 '상대방'은 자신이 승리하는 최종적인 최적해의 상태까지 미리 수를 읽어서 다음 액션(착점)을 취한다.

2) 게임 트리 탐색은 2명의 참여자가 서로 경쟁해서 승리를 쟁취하려는 게임에 대한 탐색 기법이다.

3) 게임 트리 탐색에서 '나'와 '상대방'이 교대로 상태 변경을 초래하는 액션을 취한다.

4) 게임 트리 탐색에서 '나'와 '상대방' 각자는 판세를 잘 못 읽어 잘못된 액션을 취하지는 않는다고 가정한다.

3. 지식 표현의 개요

- AI 초창기에는 탐색 중심의 GPS(General Problem Solver) 제작에 주안점을 둠. 지금은 실제적인 문제를 해결하기 위해 **지식 + 추론 패러다임**과 **특화**로 방향 전환
- 인공지능은 **형식지**에 주안점

종류	유형	비고
논리(logic)	선언적(declarative)	▪ 명제 논리, 술어 논리 ▪ 대수적 기호 연산을 통한 추론
규칙(rule)	절차적(procedural)	▪ 매칭-점화 기반 추론 기능 ▪ 규칙 기반 전문가 시스템(expert system)

논리기호	이름	결합 사례	의미
¬ ~ !	부정(negation)	$\neg P$	P 가 아님 ("not" "아닌")
∨ + ∥	논리합(disjunction)	$P \lor Q$	P 또는 Q ("or" "또는")
∧ & ·	논리곱(conjunction)	$P \land Q$	P 그리고 Q ("and" "그리고")
→ ⇒ ⊃	함의(implication)	$P \rightarrow Q$	P 이면 Q ("if-then" "[만약에] … 그렇다")
≡ ↔ ⇔	동치(equivalence)	$P \equiv Q$	$(P \rightarrow Q) \land (Q \rightarrow P)$ ("if and only if" "반드시 그럴 때만")

진리표(truth table)
- 참, 거짓과 측면에서의 **논리기호**의 역할

P	Q	$\neg P$	$P \lor Q$	$P \land Q$	$P \rightarrow Q$	$P \equiv Q$
F	F	T	F	F	T	T
F	T	T	T	F	T	F
T	F	F	T	F	F	F
T	T	F	T	T	T	T

$\neg P$ NOT p
$P \lor Q$ P OR Q
$P \land Q$ P AND Q

- 논리식의 진리값 **결정**(interpretation)
  P = T, Q = F, R = T 일 때,
  - $(P \lor \neg Q)$: T
  - $(Q \land \neg R)$: F
  - $(P \lor \neg Q) \land (Q \land \neg R)$: F
- 함의(implication) : $P \rightarrow Q$
  - P 이면, Q 이다.
  - $P \rightarrow Q = \neg P \lor Q$

- 정형식(wff)
- **리터럴**: 한 명제기호와 해당 명제 기호의 부정을 통칭
- **절**: 리터럴들이 논리합이나 논리곱으로만 연결된 논리식
- **논리곱 정규형(conjunctive normal form, CNF)**: 논리합 절들의 논리곱 형태의 논리식
- **논리합 정규형(disjunctive normal form, DNF)**: 논리곱 절들의 논리합 형태의 논리식
- **모델**: 논리식에 포함되어 있는 각 명제기호의 진리값 상태(2^n 개의 모델 존재)
- **항진식(tautology)**: 모든 모델에서 참인 논리식
- **모순적 또는 충족불가능한 논리식(contradiction or unsatisfiable)**: **항위식(contradiction)**: 모든 모델에서 거짓인 논리식
- **충족가능한(satisfiable)** 논리식: 적어도 하나의 모델에서 참인 논리식
- **연역적 추론**: 참인 사실들로부터 새로운 참인 명제를 도출하는 것
  - **추론의 정당성(soundness)**: 추론 규칙을 통해 도출된 논리식은 항상 논리적 귀결이 된다 $(\nabla \vdash \mathfrak{w} \text{ -> } \nabla \models \mathfrak{w})$
  - **추론 규칙의 완전성(completeness)**: 주어진 논리식들의 논리적 귀결은 항상 추론 규칙을 통해 찾아낼 수 있다.
  - **정리 증명(theorem proving)**: ∇가 공리(추론을 할 때 참인 것으로 주어지는 논리식), ⊠가 정리(공리들에 추론 규칙을 적용하여 얻어지는 논리식)

논리적 귀결(logical implication)

$\Delta \models \omega$

Δ : 정형식(wff)의 집합
ω : 정형식 (의 집합)

P	Q	Δ: $P \land Q$	Δ: $P \rightarrow Q$	ω: Q
F	F	F	T	F
F	T	F	T	T
T	F	F	F	F
T	T	T	T	T

사례 : $\Delta = \{P \land Q, P \rightarrow Q\}$, $\omega = \{Q\}$

- Δ에 있는 **모든** 정형식을 **참**(T)으로 만드는 **모델**은 ω를 **참**(T)으로 만들 경우
  - Δ가 참이면, ω도 참이다
  - ω는 Δ의 논리적 귀결(logical implication or consequence)
- 추론
  - Δ로부터, 알려지지 않은 ω를 찾는 것 (연역 과정)

추론규칙(inference rule)
- 추론 과정에 적용하는 규칙
- **함의**(p → q)의 논리적 관계를 이용하여 새로운 논리식을 **연역적**으로 유도
  - p : 전제(premise)
  - q : 결론(conclusion, consequence)

P : 새이다. $\neg P$: 새가 아니다.
Q : 날개가 있다. $\neg Q$: 날개가 없다.
R : 날 수 있다.

- 긍정 논법 (modus ponens) $P \rightarrow Q, P \vdash Q$
  전제 : 새이면 날개가 있다.
  사실 : 새이다.
  결론 : 날개가 있다.
- 부정 논법 (modus tollens) $P \rightarrow Q, \neg Q \vdash \neg P$
  전제 : 새이면 날개가 있다.
  사실 : 날개가 없다.
  결론 : 새가 아니다.
- 삼단 논법 (syllogism) $P \rightarrow Q, Q \rightarrow R \vdash P \rightarrow R$
  대전제 : 새이면 날개가 있다.
  소전제 : 날개가 있으면, 날 수 있다.
  결론 : 새면 날 수 있다.

- **장점**:
  - 대수학적 근거를 바탕으로 논리 개념을 자연스럽게 표현 가능
  - 지식의 첨가와 삭제가 용이
  - 정리증명 등 정형화된 지식 영역에 적합
- **단점**:
  - 절차적인 지식의 표현이 난해
  - 실세계의 복잡한 구조의 표현이 난해
- **술어 논리(predicate logic)**: 명제 논리의 한계를 극복하기 위해 고안된 논리 체계. **술어명(인수)** 형태로 표현
  - **항(term)**: 상수, 변수, 함수로 구성된 표현식. 그 개수는 제한이 없음
    - ex) John, x, father(John, legs(John))

전칭 한정사(universal quantifier) : ∀x
- "모든 x"
- "given any" "for each" "for every"

존재 한정사(existential quantifier) : ∃x
- "~한 x가 **적어도 하나** 존재한다."
- "there exist" "for some" "어떤 경우에"

- **일차 술어논리(first-order predicate logic)**: 변수에만 한정사를 쓸 수 있도록 한 술어논리

술어 논리를 이용한 지식 표현

likes(Tom, Math)
likes(Tom, Physics)
likes(Tom, Philosophy)

likes(Judy, Math)
likes(Judy, Physics)
¬likes(Judy, Philosophy)

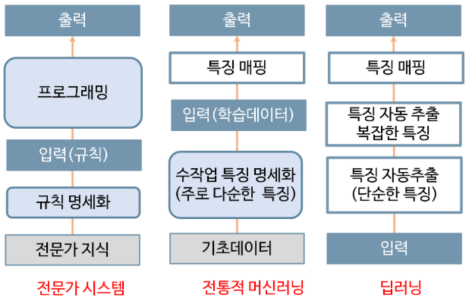
¬ likes(John, Math)
¬ likes(John, Physics)
likes(John, Philosophy)

- ∀x ∀y likes(x, y) : 모든 학생은 모든 과목을 좋아한다. (거짓)
- ∃x ∀y likes(x, y) : 모든 과목을 좋아하는 학생이 있다. (참), Tom
- ∀x { likes(x, Math) → likes(x, Physics) } : 수학을 좋아하면 물리도 좋아한다. (참), Tom & Judy
- ∃ x { likes(x, Math) ∧ likes(x, Physics) } : 수학과 물리를 모두 좋아하는 학생도 있다. (참), Tom & Judy

- **Horn 절**: 절의 한 형태로, **긍정 리터럴이 최대 하나인 절** ex) p -> Q(Horn 절이 아님), ¬p ∨ Q(Horn 절)
- 논리(logic) 기반 지식 표현 및 추론
  - 기호를 사용하여 문장들을 표현 & 기호의 조작을 통해 문장들의 참 또는 거짓을 판정하는 분야
  - 명제논리와 술어논리로 구분
    - 명제논리는 문장을 기호로 표현 하여 지식을 표현
    - 술어논리는 문장의 내용을 술어(predicate)과 객체(주체 또는 대상)으로 나눠 지식을 표현
  - 긍정논법, 부정 논법, 삼단논법, 도출(resolution) 등의 대수적 기법을 사용하여 추론 • 장단점
  - 선언적 지식 표현 및 지식의 추가 • 삭제가 용이
  - 실환경의 복잡한 지식의 표현에는 한계
    - 술어논리 기반의 PROLOG 언어가 한 때 주목을 받았지만 실패
- **규칙 기반 추론**:
  - 전문가시스템은 상대적으로 지식 표현이 수월하고, 불확실한 지식을 표현하는데 장점이 있어 규칙 기반으로 제작됨
  - 지식의 자동추출 및 축적(학습) 불가능

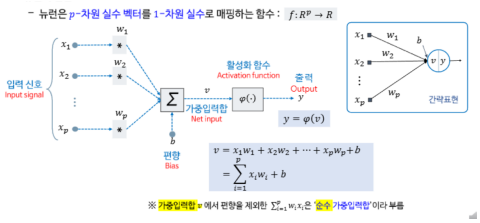
머신러닝의 개념

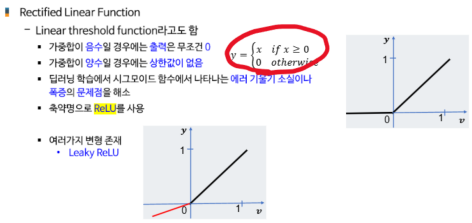
- 인공지능 > 머신러닝 > 딥러닝
- 프로그래밍은 해결할 수 없는 문제가 많았다: **특징점 자동 추출 학습, 새로운것 인식**(필기체, 얼굴 인식 등)
- **머신 러닝 태스크: 분류, 상관관계 파악(회귀), 군집화**
- **학습 방식**: 감독학습, 비감독학습 등
- **분류**: 분류 대상 카테고리(클래스)는 사전에 지정됨. 사물 인식, 손글씨 숫자 인식, 음성 인식 등
- **회귀(상관관계 분석)**: 출력은 실수. 상관관계 모델링을 통한 예측 및 추세 분석
- **로지스틱 회귀**: 종속변수가 범주형. 이항형, 다항형, 사실은 확률적 성격을 가미한 분류 기법.
- **군집화**: 클러스터의 개수는 사전에 정해져 있지 않음. 문서 분류, 영상의 색조에 따라 영상을 분류(위성 사진으로 곡물 수확량, 화재 상황 분석 등)
- **감독 학습(지도 학습)**: 데이터에 사람이 작업을 한 라벨을 다한 학습 데이터 구축. 분류, 회귀
  - 학습 샘플은 **샘플과 목표치 쌍**
- **비감독 학습(비지도 학습)**: 라벨이 없는 데이터로 학습. 군집화 등
- **강화 학습**: 게임 수행(아타리 알파고) 등
- **딥 러닝**: 학습 데이터(MNIST, CIFAR-10 등)에서 특징점 자동 추출 학습. **다층구조, 계층적 추상화 학습, End-to-End 학습**

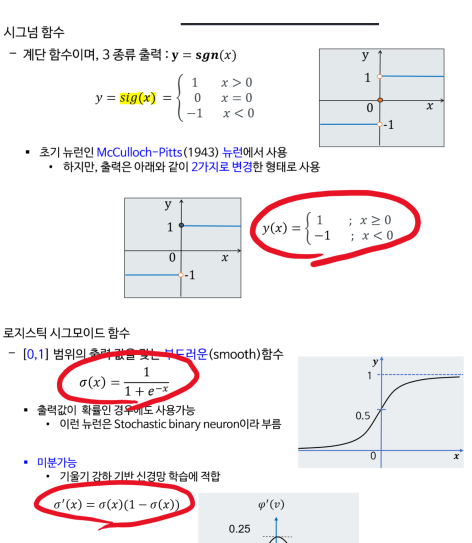


신경망

- **신경명의 학습: 가중치 변경**
- 생물학적 뉴련: 시냅스 연결 강도, 인공 뉴럴 네트워크: 연결선의 가중치 크기
- 학습 데이터를 입력으로 사용하여 연결 강도에 반영
- b = -θ



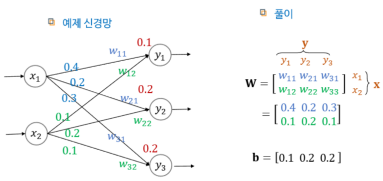




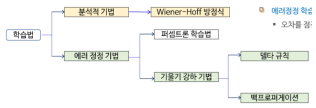
- **퍼셉트론**: 입력층과 출력층만으로 구성. 엄밀하게는 뉴런은 아니다. **시'그넘 함수의 수정본** 사용 (1 v >= 0)
  - **선형 분리 가능한 데이터에 대해서는 무조건 수렴**
  - 퍼셉트론의 한계: 선형분리성 (OR 연산은 가능하지만 XOR 연산은 불가능)

FNN

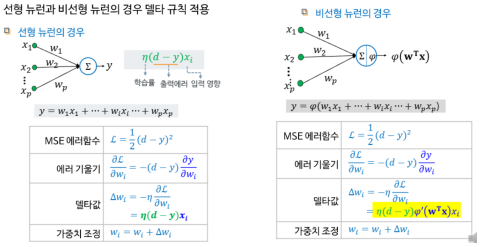
- **No cyclick link, 한 개 이상의 히든 계층, 다양한 활성화 함수 사용**
- **완전연결층**



- affine 계층: 선형 뉴런으로 구성된 계층. $xW + b -> y$
- **단일 모드**: 한번에 한개의 입력을 처리
- **배치 모드**: 여러 개의 입력을 한 번에 처리. 병렬처리 가능, 학습 속도 향상, 일반적으로 더 나은 성능
- 배치 처리를 하는 이유: **병렬 처리로 학습 속도 향상, 정확도는 떨어지나 일반화 좋다.**



- 평균제곱에러($\frac{1}{2N}\sum_{d=1}^{N}\sum_{i=1}^{q}(d_{di}-y_{di})^2$)
- 기울기 강하법: $x_{t+1} = x_t - \eta\nabla f(x_t)$



1. 신경망의 학습이 문제로 대두 (※ 델타규칙은 단층신경망에만 유효)
2. 백프로퍼게이션 알고리즘은 다층 신경망의 학습을 가능하게 함
3. 하지만, 기울기 소실 또는 폭증(error gradient vanishing or explosion) 현상으로 계층의 개수가 커지면 실용성이 떨어짐)다층 신경망이 소규모 응용에만 사용이 제한됨
4. 백프로퍼게이션 알고리즘의 여러 다른 이름: 오류 역전파 알고리즘, 역전파 알고리즘, 역전파 학습 알고리즘

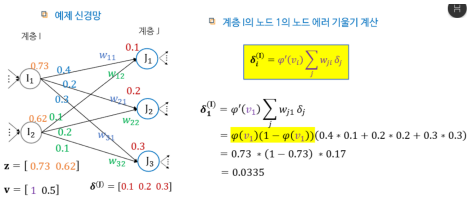


Figure 1: 노드 에러 기울기(활성화 함수의 미분값 x 후방계층 노드들의 가중 노드 에러 기울기의 합)

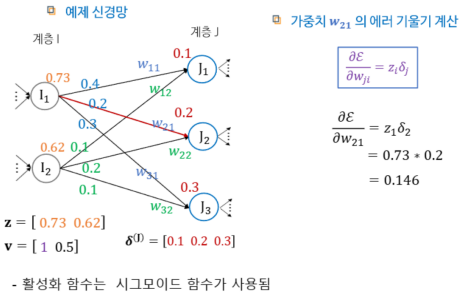


Figure 2: 가중치 에러 기울기(앞단 노드의 출력 x 뒷단 노드의 에러 기울기)

- SGD(stochastic gradient descent): 샘플 단위로 가중치 갱신.(패턴 모드, Online 모드) 전체 샘플 학습 처리를 Epoch라고 함. 미니배치의 크기가 1개 샘플인 경우
- 미니 배치 모드: 배치 모드와 온라인 모드의 절충안. Local minima에 빠져도 나올 가능성이 있음.
- 확률적 순서: 샘플의 학습 순서를 무작위로 선정