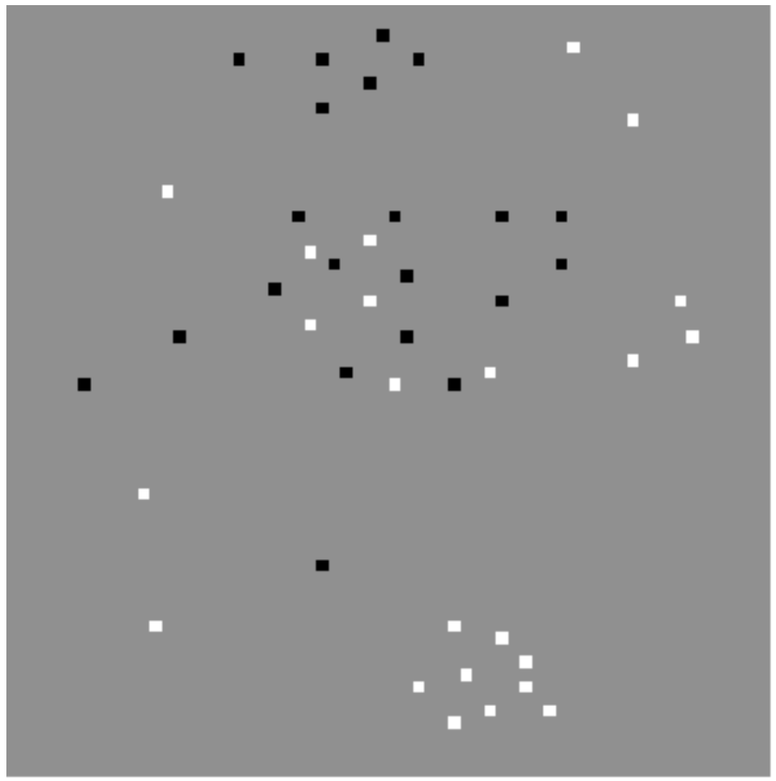
Influence maps

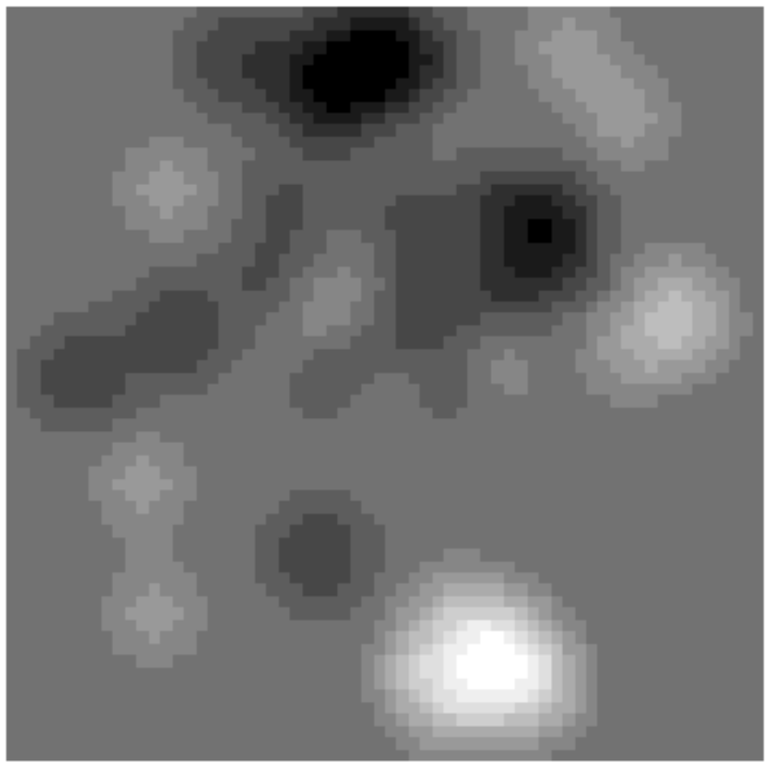
Kraftfull representation av information om spelsituation!

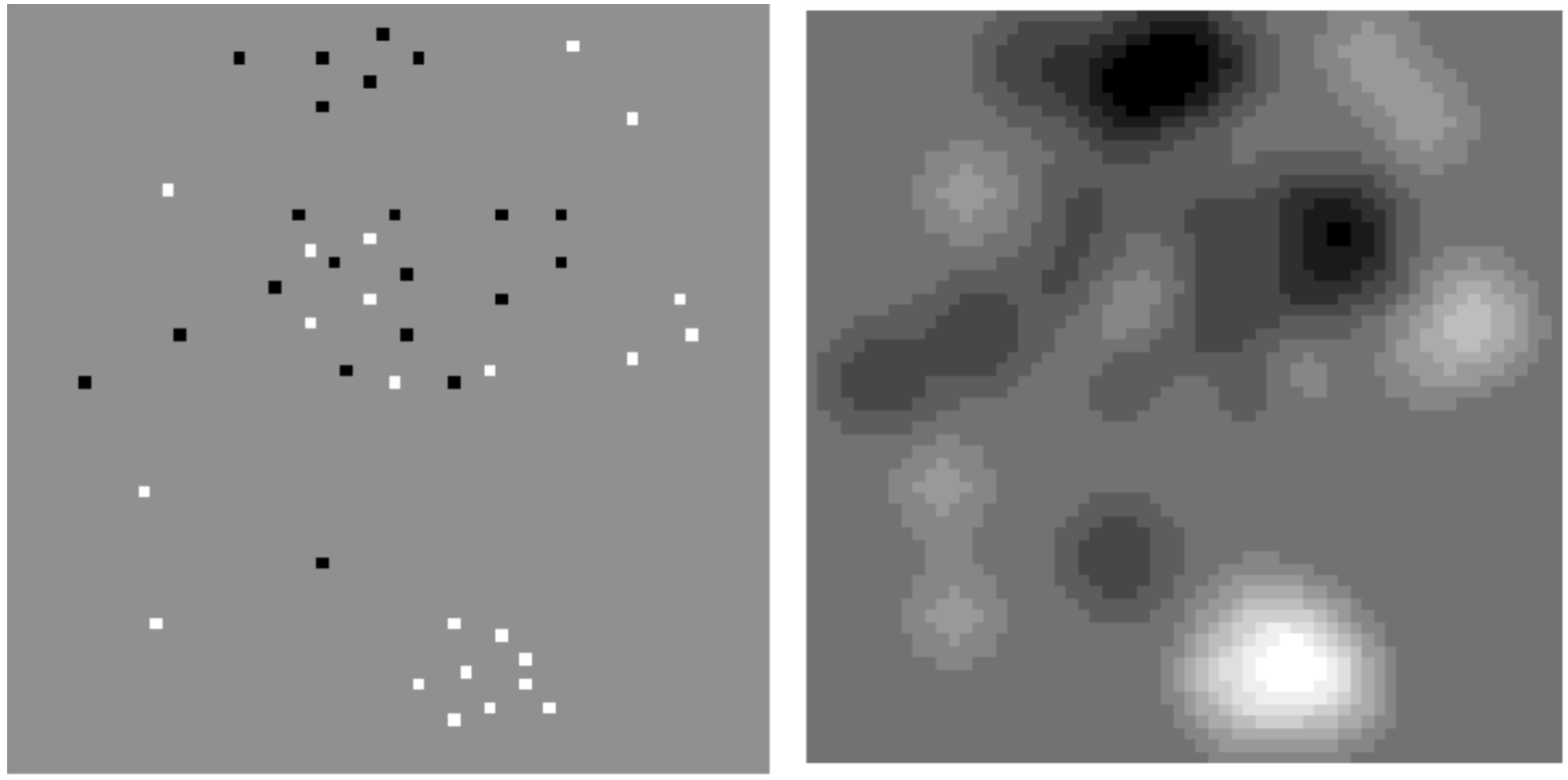
Lågpassfilter av karta av spelagenter förenklar analysen.

- Vem “äger” rutan?
- Är området säkert, inga fiender nära?
- Vilken annan medspelare står fri?
- Var är motståndaren som svagast?

Influence maps, exempel







Fatta beslut beroende på influensvärdet

- Anfall svaga punkter
- Passa till obevakade medspelare

Varianter

- Terrängberoende filtrering, långsammare spridning i svår terräng
- Flera influence maps för olika data
- Olika värde i kartan för olika pjästyper

Varianterna styrs i stort av speltypen, men grundprincipen är generellt användbar till många problem.

Exempel

En influence map för egen styrka. Vad äger jag?

En influence map för motståndaren.

En influence map för värdefulla punkter (resurser, vinstpoäng, baser...)

En karta över terrängen. Låt filtrering eller vägsökning påverkas av denna.

Angrip punkt med högt värde/(motstånd-styrka)

Exempel

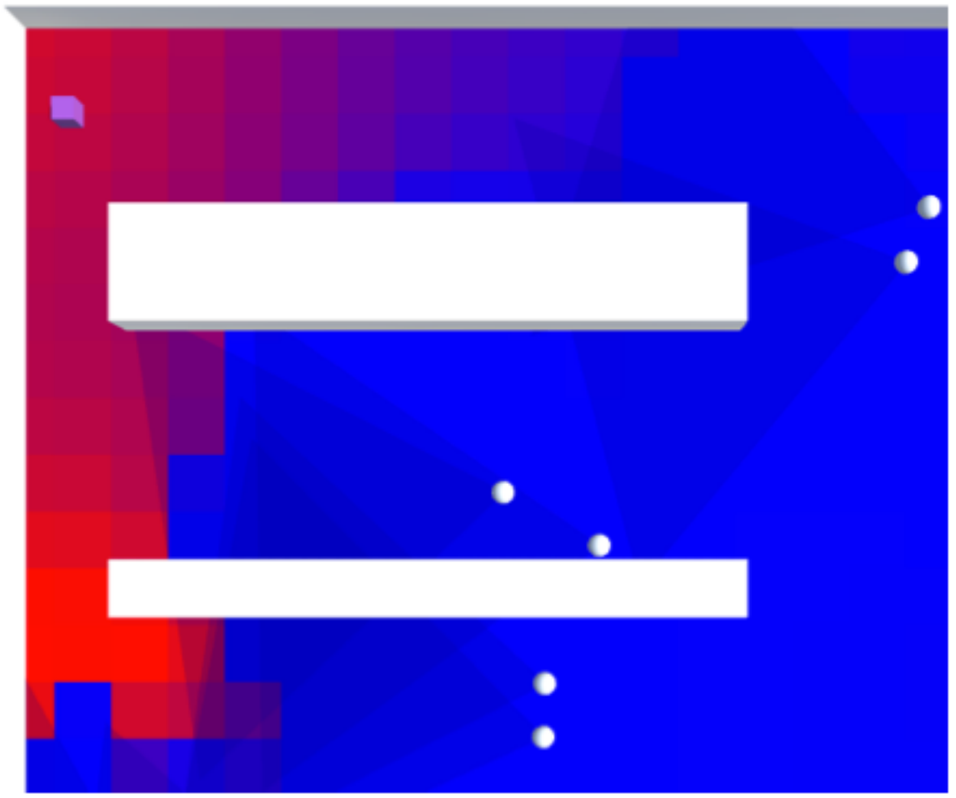
Projekt 2016 (Dennis Ljung & Sebastian Sundin): AI som söker efter spelaren.

Influence map som "minns" var den sökt!

Om AI-agenten ser spelaren: Stark ökning kring den platsen.

Dessutom modifierad flocking för att sprida AI-agenterna. (Kunde också varit influence map!)

Rött: Områden där AI-agenterna vet att spelaren är eller varit relativt nyligen.



Influence maps och bildanalys

En influence map är en bild av den spatiella situationen.

Bildanalys är ett välutforskat ämne - många metoder finns för att analysera innehållet.

- Segmentering, watersheds
- Linjära och olinjära filter
- Formanalys, skelettering

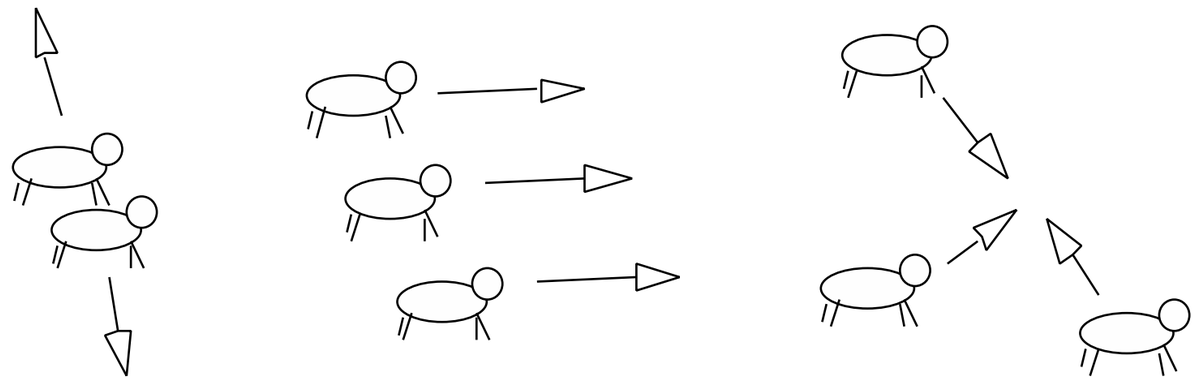
Analysen kan lätt snabbas upp på GPUn med vanliga grafikmetoder

Flocking - Boids

Gruppbeteenden kan simuleras med enkla medel.

Craig W Reynolds “boids”, baserade på tre enkla regler:

- Separation - undvik att gå för nära varandra
- Alignment - sträva åt samma håll
- Cohesion - sträva mot mitten av gruppen



Boids, implementation

Partikelsystem

Separation: Kraft bort från närmaste boid

Alignment: Beräkna medelhastighet, drag egen hastighet mot denna

Cohesion: Beräkna medelposition, kraft mot denna

Boids (naiv N^2)

```
for all boids i
  for all boids j except i
    if dist(i, j) < kMaxDistance
      count += 1
      accumulate speed difference
      accumulate average position
      accumulate avoidance vector

for all boids i
  if count > 0 then
    divide accumulated data by count
  affect speed by
     speed diff (alignment)
      average pos (cohesion)
      avoidance vectors (avoidance)
  position += speed
```

Viktiga parametrar att ”skruva på”

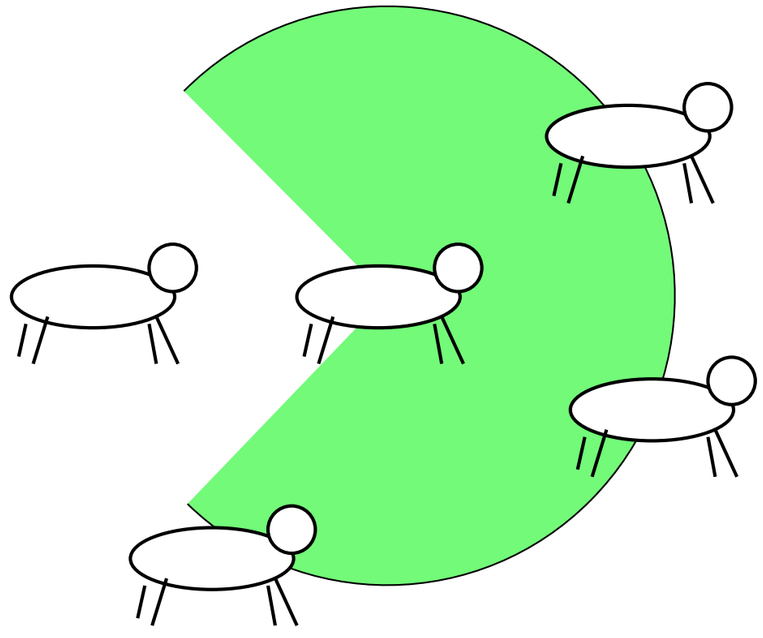
Vilken vikt på varje komponent?

Hur stor omgivning skall räknas?

Avståndsberoende funktioner för separation och cohesion?

Boids, synfält

En boid kanske inte bör ha ”ögon i nacken”

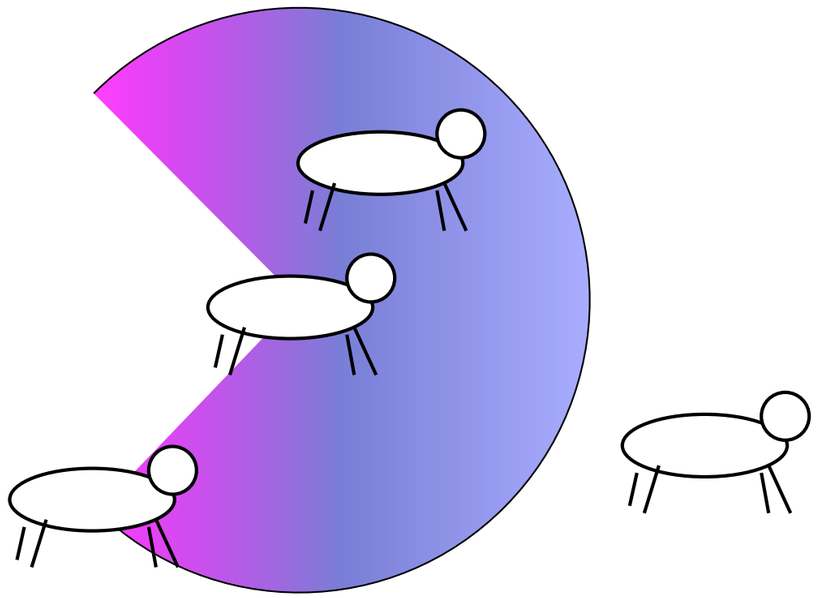


Vilka andra boids skall räknas?

Begränsa på vinkel såväl som avstånd.

Undvik diskontinuiteter

En enkel tröskelfunktion kan ge ryckig animation



Mjukt avtagande funktion?

För stark separation kan ge våldsamma krafter om funktionen blir för stor nära

Boids är exempel på A-life

Imitation av verkliga beteenden

Optimala lösningar är inte målet!

En A-life-vägsökning får gärna vela och tramsa, om det stämmer med agentens typ.

Optimal lösning för ett djur på vägen vs verkligt beteende?

Boids är också exempel på *gruppbeteenden*

Gruppbeteenden är ofta icketriviala, intressanta problem.

Hur delar man upp grupper för bästa resultat?

Hur undviker man att två grupper hindrar vägen för varandra?

Skall individer tilldelas sin "plats i ledet" eller hitta dem själva?

Flockinglabben på torsdag

Baseras på enkel 2D-spritemotor

Implementera boids-algoritmen

Något tillägg för att få ett "rikare" system.

Lärande system

System som lär sig är ett intressant ämne! Mest kända metoderna/koncepten:

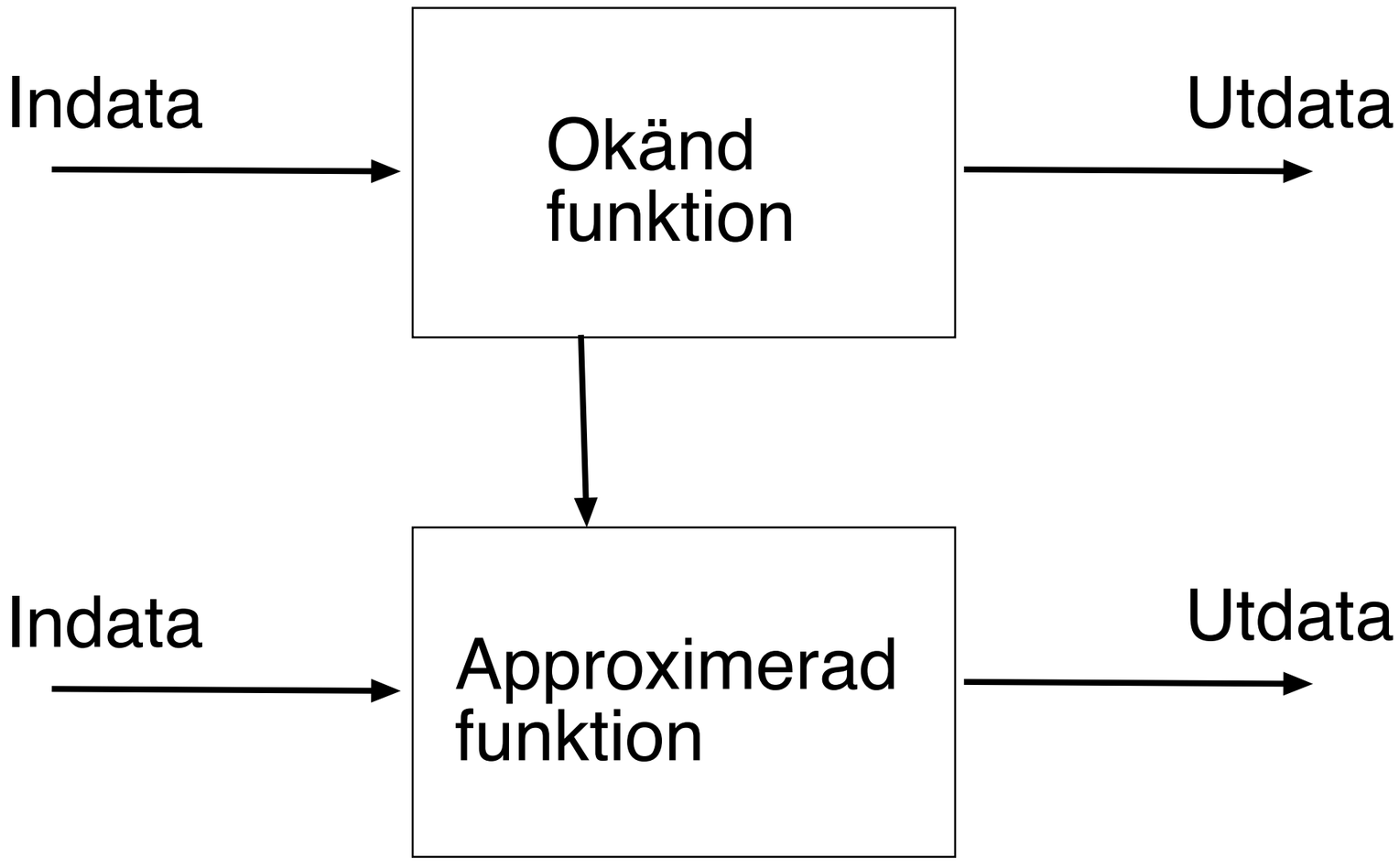
- Neurala nät

Generativ AI:

- LLM
- Diffusion

Men observera: Allt är i grund och botten *optimering*!

Optimering?


Indata
Okänd funktion
Utdata
Indata
Approximerad funktion
Utdata

Grundmetoder för lärande system/funktionsoptimering

Hill climbing

Simulated annealing

Genetic algorithms

Generativ AI

Användbart för spel?

Generera geometri

Generera grafik, texturer

Generera kod

Allt måste kontrolleras, modereras

Neurala nät, Neural networks

En representation av systemet, av den önskade funktionen.

Kan med tidigare nämnda metoder modellera komplexa funktioner.

Mycket hett ämne idag, "deep learning", tack vare att vi nu kan hantera stora nät.

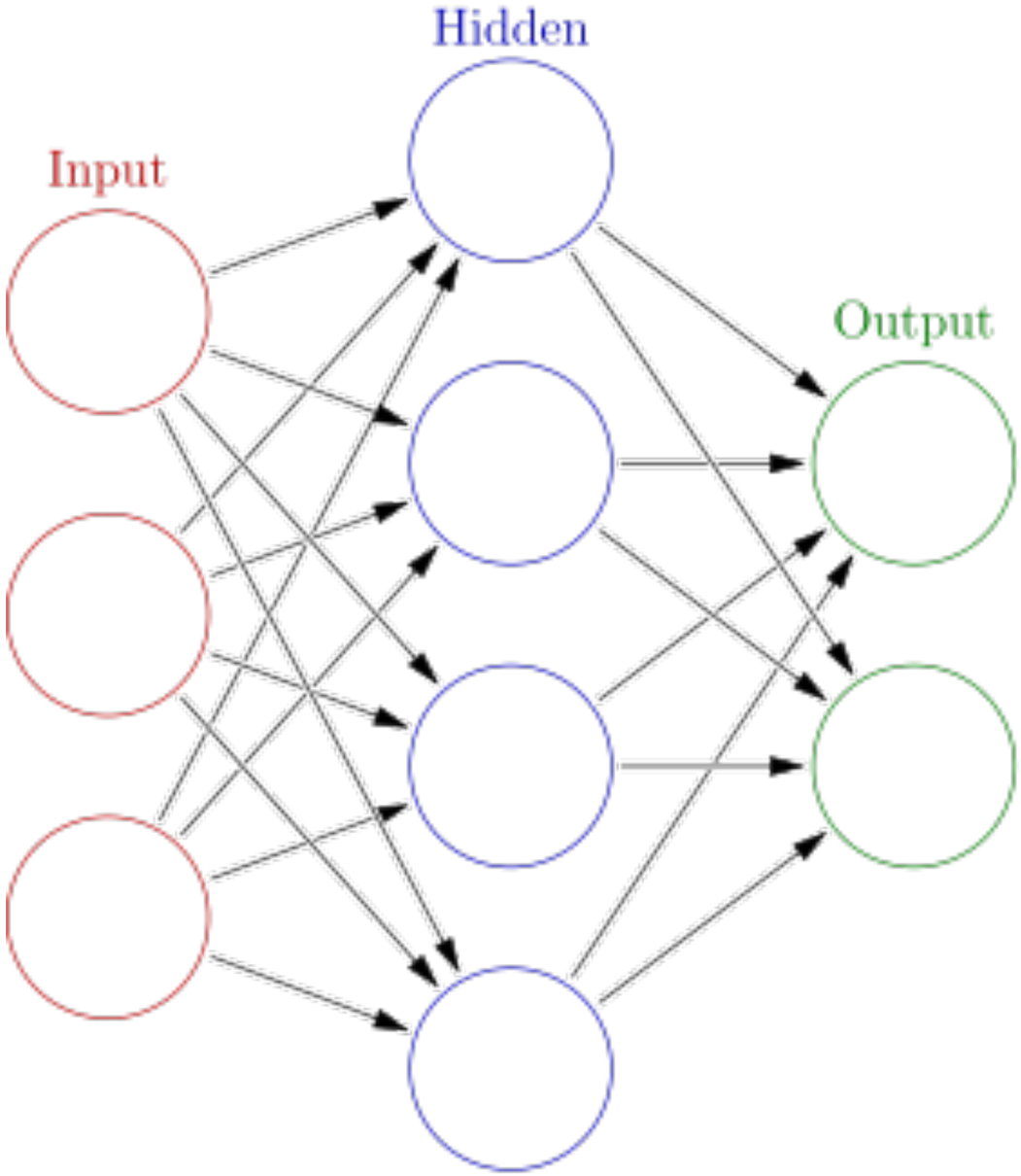


Bild från Wikipedia

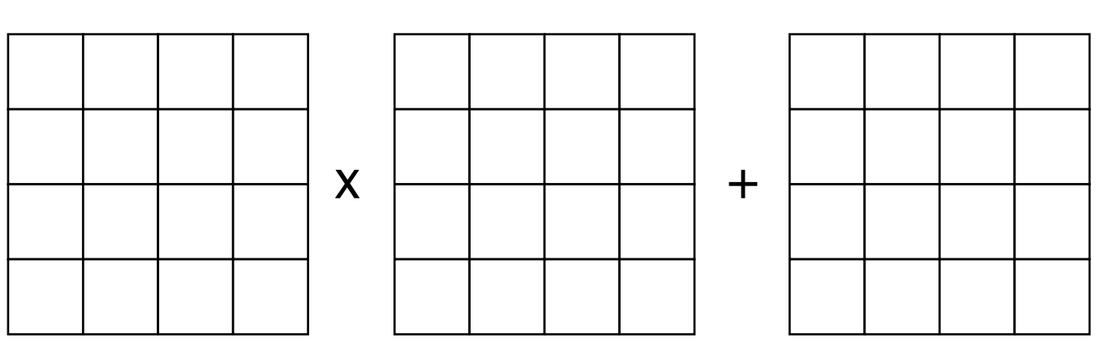
Tensor cores

GPU-hårdvara för lärande system.

Accelerar matrismultikation och ackumulering

4x4 matrismultiplikation

Matrismultiplikation i låg precision (!)



Bygg större multiplikationer med 4x4 som byggstenar

Låg precision för snabbare beräkningar

FP16 in, FP32 ut

Är detta relevant för spel?

På vad sätt?

Inlärning under gång?
Off-line för träning eller generering?

Tidiga spel med inlärning eller liknande inslag:

SimLife

Viss simulering av genetik, men

- urval av population är inte (exakt) samma sak som inlärning och upplevs inte som det
- Spelet var inte särskilt underhållande och simuleringen var dålig

Black&White

Inlärning av beteende på “varelsen”

Intressant idé, mindre lyckat resultat

Men är AI viktigt för spel?

Javisst!

- Beteenden och vägplanering
- Världsrepresentation
- Analys av speldata i denna representation
- Komplex agentrepresentation för styrning av beteenden

Därmot är lärande mer marginellt (men bevisa gärna att jag har fel)

AI-projekt?

Många möjligheter.

De bra baserar sig ofta på influence maps och annan analys av världen.

A* räcker inte som huvudpoäng.

Att använda generativ AI är inte ett projekt.

Våga gå lite bortom grunderna.