APRENDIZADO DE MÁQUINA
e o
ATARI 2600

Murilo Saraiva de Queiroz

A revista **Nature** é uma das publicações científicas mais antigas e prestigiadas do mundo: foi nela em que foram divulgados pela primeira vez resultados como a fissão nuclear (1939), a estrutura do DNA (1953) e o clone do primeiro mamífero, a ovelha Dolly (em 1997). Em fevereiro de 2015 um artigo publicado na Nature chamou a atenção de entusiastas do mundo todo: em ***"Human-level control through deep reinforcement learning"*** ("Controle em nível humano através de aprendizado por reforço profundo", numa tradução livre) pesquisadores da empresa **Deep Mind**, recentemente adquirida pelo **Google**, apresentam um algoritmo capaz de aprender sozinho a jogar, com desempenho equivalente ou superior ao de um jogador humano, 49 títulos de **Atari 2600**! Neste artigo vamos explicar como esse algoritmo funciona, de onde veio e por que esse resultado é relevante ao ponto de aparecer numa publicação tão conceituada.

Inteligência Artificial e Aprendizado de Máquina

Quando falamos em inteligência artificial em videogames a primeira coisa que nos vem à cabeça é o comportamento dos inimigos nos jogos, controlados pelo computador. A campanha de lançamento do Atari 2600 no Brasil, de 1983, já enfatizava a importância disso, com o slogan clássico ***"Atari: O Melhor Inimigo do Homem"***. De certa forma o primeiro grande salto na história dos videogames foi, justamente, transformar o computador no adversário: **SpaceWar!** e **Pong** requerem duas pessoas e o jogo é meramente o "campo" em que elas competem entre si; é com **Breakout** (e, mais tarde, **Space Invaders**) que o desafio passa a ser entre jogador humano e a máquina. Mas o blocos de Breakout não reagem, e os alienígenas de Space Invaders se comportam de maneira metódica, mecânica e previsível. Só algum tempo depois é que o comportamento complexo dos inimigos no jogo ganhou destaque: as diferentes personalidades e comportamentos dos fantasmas de **Pac-Man** são uma das razões para o sucesso estrondoso do jogo.

Como qualquer jogador razoável de Pac-Man sabe, cada um dos fantasmas na versão de arcade segue regras bastante simples e muito específicas, mas que quando combinadas geram um comportamento rico e surpreendente (mais detalhes sobre isso podem ser encontrados no já clássico - e completíssimo - **Pac-Man Dossier**). Essa abordagem, a de programar um conjunto de regras relativamente simples que determinam as reações dos inimigos e que (quando funcionam direito!) dão a impressão de um comportamento "inteligente", é a mesma usada em quase todos os jogos de videogame, desde os clássicos até os mais recentes.

Regras assim são chamadas **heurísticas**, e é claro que, dependendo do jogo e do desempenho desejado, elas podem se tornar bastante complexas. Quando associadas a técnicas mais sofisticadas, como máquinas de estados finitos ou busca em árvore, o resultado obtido pode ser surpreendente. Em todos esses exemplos a "inteligência artificial" é parte do próprio jogo, desenvolvida cuidadosamente pelo programador para que os adversários se comportem de maneira desafiadora e divertida.

>

Como nem sempre os inimigos têm as mesmas capacidades e objetivos do jogador, nem sempre faz sentido em falar "desempenho similar ao humano" quando falamos de inteligência artificial nesse caso: como o jogador nunca assume o controle dos fantasmas, as heurísticas que os controlam têm quase nada a ver com controlar Pac-Man.

Em jogos em que há uma simetria - jogadores humanos e os controlados pelo computador têm capacidades e objetivos semelhantes, o desafio da inteligência artificial passa a ser emular o comportamento de uma pessoa: jogar de maneira inteligente, de maneira cooperativa ou competitiva, de forma satisfatória o suficiente para manter a diversão. É justamente isso que fazem os **bots** dos jogos multijogador em rede modernos, ou o modo "contra o computador" de jogos como Xadrez, Tênis, e outros.

Bots (no sentido de "software capaz de controlar o jogo assumindo o papel do jogador humano") podem ser implementados usando heurísticas cuidadosamente escritas, como as que mencionamos antes. Codificar e afinar a inteligência artificial dos bots de um jogo é um trabalho difícil e longo, que exige um conhecimento profundo do jogo, muita criatividade e testes quase intermináveis, e mesmo assim os resultados muitas vezes ficam aquém do esperado.

Mas e se fosse possível fazer com que o próprio bot aprendesse, sozinho, a jogar bem? E se o bot, imitando o que fazem os humanos, aprendesse com a prática, melhorasse a cada derrota, descobrisse sozinho estratégias e táticas capazes de melhorar seu desempenho e pontuação? Essa é justamente a proposta do **aprendizado de máquina**.

Ao invés de elaborar cuidadosamente um programa para jogar bem um título específico, muitas vezes incorporando o conhecimento do desenvolvedor (ou de jogadores experientes), ensinamos o computador a aprender sozinho. Nesse contexto existem duas formas de aprendizado de máquina que podem ser úteis: o **aprendizado supervisionado**, e o **aprendizado por reforço**. Uma outra modalidade, o aprendizado **não-supervisionado**, é menos comum no contexto de jogos e não será discutida aqui.

Aprendizado Supervisionado

O aprendizado supervisionado é baseado em uma enorme quantidade de **exemplos**. Se as soluções ou respostas para um determinado problema são conhecidas (por exemplo, se um especialista é capaz de resolver o problema ou jogar bem um determinado título), e temos uma enorme quantidade de exemplos da coisa certa a fazer, podemos usar um algoritmo que aprende a partir daqueles exemplos e imita o comportamento do especialista, inclusive sendo capaz de generalizar esse comportamento, aplicando-o em situações que não foram vistas durante o treinamento.

Um exemplo típico seria um jogo de **pôquer**: ao invés de cuidadosamente calcular as probabilidades de vencer com uma determinada mão, e escrever regras sofisticadas para decidir o que fazer em cada situação (ou seja, usar heurísticas), poderíamos usar um banco de dados com milhares de mãos de pôquer e as ações que um jogador experiente tomaria quando saísse com cada uma dessas mãos. Esse banco de dados seria usado como entrada para um algoritmo de aprendizado supervisionado, que aprenderia os padrões de comportamento

>

(muitas vezes intuitivos, e nem sempre consistentes) do especialista e os aplicaria em partidas reais. Se o algoritmo é bem sucedido ele teria desempenho razoável mesmo quando saísse com uma mão que não fosse idêntica àquelas com as quais ele foi treinado, por ser capaz de generalização.

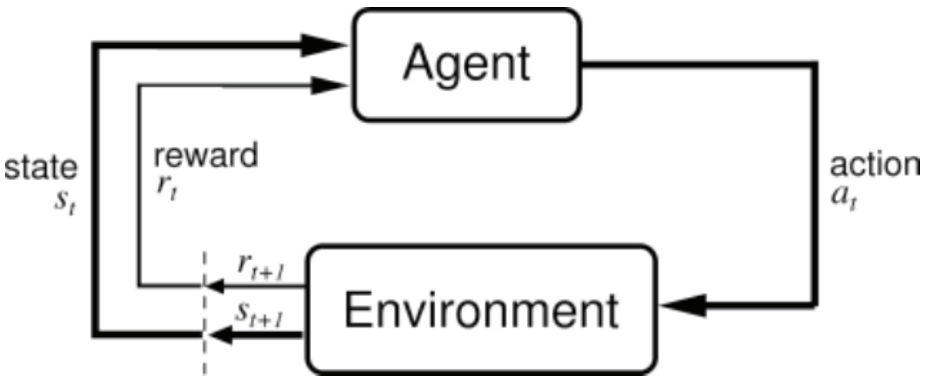
O problema do aprendizado supervisionado é que a quantidade de exemplos necessária para o aprendizado de comportamento que não seja trivial é gigantesca - no caso do pôquer, muito maior do que um jogador experiente conseguiria produzir mesmo jogando continuamente por vários meses. Uma abordagem assim funciona para problemas mais simples, ou problemas em que é possível coletar uma grande quantidade de exemplos, mas é inviável no caso da maioria dos videogames. Um outro problema é que, mesmo se tudo der certo, o desempenho do modelo gerado não vai ser radicalmente superior ou diferente do desempenho do especialista que forneceu os exemplos.

Aprendizado por Reforço

Como vimos antes, o aprendizado supervisionado funciona se temos um especialista capaz de produzir milhares ou milhões de exemplos de "como jogar bem". Mas normalmente não é assim que aprendemos a jogar (ou a fazer qualquer coisa)! Ao invés de aprender por imitação, um iniciante **"se vira" sozinho**: experimenta os controles, **explorando** a esmo o ambiente, testando como o jogo reage a determinado botão ou quando há alguma colisão. Mais cedo ou mais tarde uma de duas coisas acontece: o jogador descobre algo que é vantajoso, e ganha uma **recompensa** como pontos ou a passagem para uma nova fase por isso, ou ele é **punido** por ter feito algo ruim (perde uma vida, ou energia, por exemplo). O ciclo então recomeça: o jogador continua a explorar, dessa vez tentando evitar o que trouxe punições no passado e buscando o que trouxe benefícios. Com o tempo, o jogador começa a entender "o mundo do jogo", as regras que o governam e as estratégias que trazem mais recompensa a longo prazo.

É exatamente assim que o aprendizado por reforço funciona. A cada instante, o agente observa o **estado** do jogo (onde ele está, que objetos estão por perto, etc.). Baseado no estado, ele decide qual **ação** irá executar, que irá levá-lo a um **novo estado**. Inicialmente, por não saber absolutamente nada a respeito do jogo, o agente simplesmente sorteia uma ação qualquer que o levará ao próximo estado. Após uma série de ações quaisquer, escolhidas aleatoriamente, o agente chega em um estado em que há uma recompensa ou punição, ou seja, onde surge o sinal de **reforço**. Esse é o momento em que o aprendizado acontece: o estado em que ele recebeu uma recompensa passa a ter um valor maior, correspondente ao prêmio. **Todos os estados anteriores** pelos quais o agente passou até chegar o prêmio também aumentam de valor; quanto mais próximo do estado final, maior o aumento. O sucesso (ou o fracasso) "deixa um rastro" por todos os estados por onde o jogador esteve, e afeta todas as decisões que o levaram até o resultado final.

Ao reiniciar, na hora de decidir qual ação tomar, o agente já tem uma informação nova: ele ainda escolhe aleatoriamente a ação a ser tomada, mas dá uma **probabilidade** maior para a ação que leve para um estado de valor maior. Conforme mais e mais partidas são jogadas, mais informação é obtida sobre quais estados são vantajosos, quais são prejudiciais, e quais são os caminhos (ou seja, as sequências de estados) que levam a recompensas maiores.



Se o estado é a posição do personagem num labirinto, por exemplo, ao longo do tempo o agente acaba construindo um "mapa" que indica onde estão os tesouros e onde estão os perigos do jogo. Se é difícil saber para qual estado o agente vai ao realizar uma determinada ação, é mantida uma outra tabela que armazena as probabilidades de, partindo de um determinado estado s1, a ação a levar a um determinado estado s2.

Aprendizado por Reforço e Videogames

Existem diversos algoritmos de aprendizado por reforço, todos baseados nos princípios descritos anteriormente. Pesquisadores da área comparam esses diferentes algoritmos usando-os para resolver as mesmas tarefas, e comparando o tempo que os algoritmos levam para aprender a resolvê-la, ou quão bem (quanta recompensa, por exemplo) ela é atacada. O ideal para essas comparações é conseguir um grande número de tarefas em ambientes relativamente simples, que tenham objetivos claros, com os critérios de sucesso e falha bem definidos. Videogames, principalmente os clássicos, são excelentes candidatos para isso, e ainda oferecem outras vantagens: são facilmente compreendidos, despertam interesse do público e os resultados obtidos podem ser diretamente aplicados comercialmente com baixo investimento.

A ideia de usar videogames para avaliar algoritmos de aprendizado por reforço se tornou muito popular nos últimos dez anos. Nos primeiros trabalhos do gênero pesquisadores modificavam emuladores de código aberto para que funcionassem em conjunto com os algoritmos de aprendizado por reforço, uma tarefa bastante delicada e trabalhosa. Mais recentemente foram criados ambientes, como o **Arcade Learning Environment**, especificamente para esse fim, que já fazem toda a integração necessária, permitindo que os pesquisadores foquem apenas nos algoritmos de aprendizado propriamente ditos.

Para que um algoritmo de aprendizado por reforço aprenda a jogar um título de videogame ele precisa de acesso a três elementos: o **estado** atual do jogo, ou seja, a localização dos elementos na tela tais como inimigos, itens a serem coletados, o avatar do jogador, etc.; uma forma de realizar **ações** (a direção da alavanca e os botões do joystick); e o sinal de **reforço** (a pontuação obtida no jogo, por exemplo).

Tipicamente as ações disponíveis são sempre as mesmas (no caso do Atari, oito direções possíveis e o botão). O sinal de reforço varia de jogo para jogo, e é uma das poucas coisas que exigem uma atenção específica: o ambiente extrai a pontuação do jogo e fornece esse número, já processado, para o algoritmo.

Extrair e representar o estado do jogo é um problema mais complicado. Se fornecermos informações estruturadas cuidadosamente extraídas e organizadas para o algoritmo (por exemplo, localização e identificação precisa de cada um dos elementos - jogador, inimigos, prêmios, etc. - do jogo na tela), o aprendizado fica mais fácil, mas isso requer um esforço específico e não trivial para cada jogo. Quanto mais investimos na representação do estado do jogo, mais fácil o problema fica, porque estamos incorporando nosso entendimento do jogo à essa representação. Além disso, menos interessante o algoritmo de aprendizado se torna, já que ele se torna cada vez mais dependente dos mecanismos de um jogo específico.

A simplicidade do Atari 2600 traz uma solução interessante para esses problemas, possibilitando uma solução genérica - testar um mesmo algoritmo em dezenas ou centenas de jogos diferentes, com mecânicas distintas. A memória **RAM** do Atari é minúscula: meros 256 bytes de RAM. Junte-se a isso alguns outros poucos registradores do processador e do hardware de vídeo (TIA), e você tem uma representação completa do estado do hardware do 2600 - que pode ser usado como entrada para o algoritmo de aprendizado por reforço!

Usando essa abordagem (memória RAM como representação do estado do jogo, joystick e botão

>

como representação das ações possíveis, e pontuação ou tempo de duração da partida como sinal de reforço) vários autores obtiveram resultados muito interessantes em diversos jogos, usando tanto algoritmos simples (**Sarsa, Q-Learning**) quanto abordagens mais sofisticados tais como **NEAT** ("Neuro-evolution of Augmenting Topologies") e **CMA-ES** ("Covariance Matrix Adaptation Evolution Strategy").

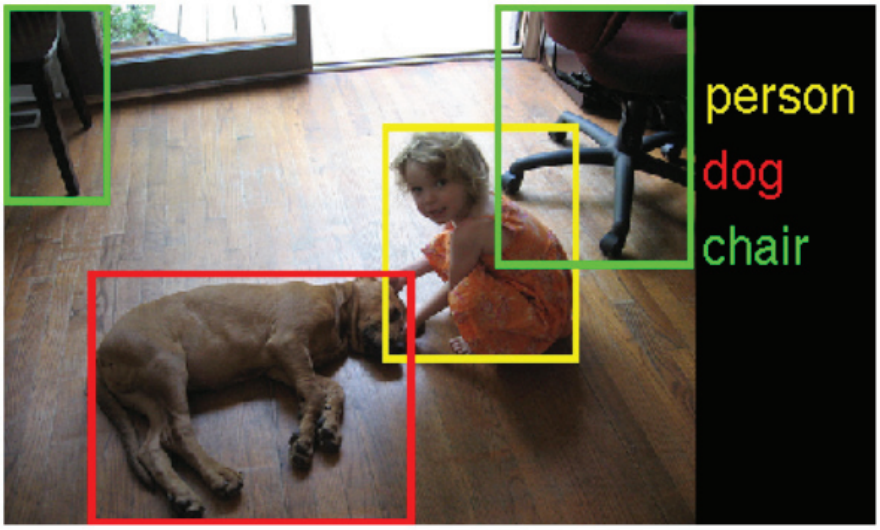
Mas se essa é uma abordagem conhecida, qual foi a grande inovação que fez a pesquisa do time do Google ser publicada na Nature? O grande diferencial deles é justamente na **representação do estado do jogo**. Pessoas não jogam videogame observando os valores armazenados na memória RAM de um videogame! Pessoas aprendem a jogar observando, obviamente, a tela do jogo... E é justamente isso que o time da Deep Mind fez: aprendizado por reforço baseado em visão computacional - **a entrada que o algoritmo enxerga é simplesmente a imagem que é exibida na tela**, e é isso que deixou muita gente na comunidade científica boquiaberta.

Visão Computacional e Aprendizado Profundo

Visão computacional, "fazer o computador entender imagens", é um problema extremamente difícil. Até poucos anos atrás, os resultados obtidos eram muito, muito ruins, e as técnicas usadas bastante limitadas: pesquisadores criavam algoritmos específicos para extrair características bem definidas da imagem - por exemplo identificar formas geométricas de determinada cor ou tamanho, e usavam regras heurísticas para, com base nessas características, identificar o que havia na imagem ou tomar uma decisão.

Essa estratégia de extração de características bem definidas também foi usada para gerar a entrada de algoritmos de aprendizado por reforço para videogames, juntamente com a ideia da RAM; se você além da RAM fornece apenas a localização, tamanho e cor dos objetos que aparecem na tela, o aprendizado fica bem mais fácil, e ainda assim genérico o suficiente (já que você não precisa incluir informação de um jogo específico, está só manipulando cores e formas geométricas). Os melhores resultados obtidos até pouco tempo atrás usavam esse tipo de abordagem (extração "manual" de características da imagem), que é suportada pelo Arcade Learning Environment.

Um dos problemas mais tradicionais de visão computacional é a classificação de imagens: dada uma imagem o computador deve dizer o que aparece nela, dizendo por exemplo "pastor alemão", "orquídea" ou "avião a jato"). A competição anual da ImageNet consiste em classificar 150.000 imagens em aproximadamente 1000 categorias diferentes.



A edição de 2010 da competição da ImageNet trouxe um resultado dramático: pesquisadores da Universidade de Toronto (Alex Krizhevsky, Ilya Sutskever e Geofrey Hinton) ganharam a competição com uma enorme vantagem, usando pela primeira vez uma **rede neuronal profunda** chamada **AlexNet** para resolver o problema.

>

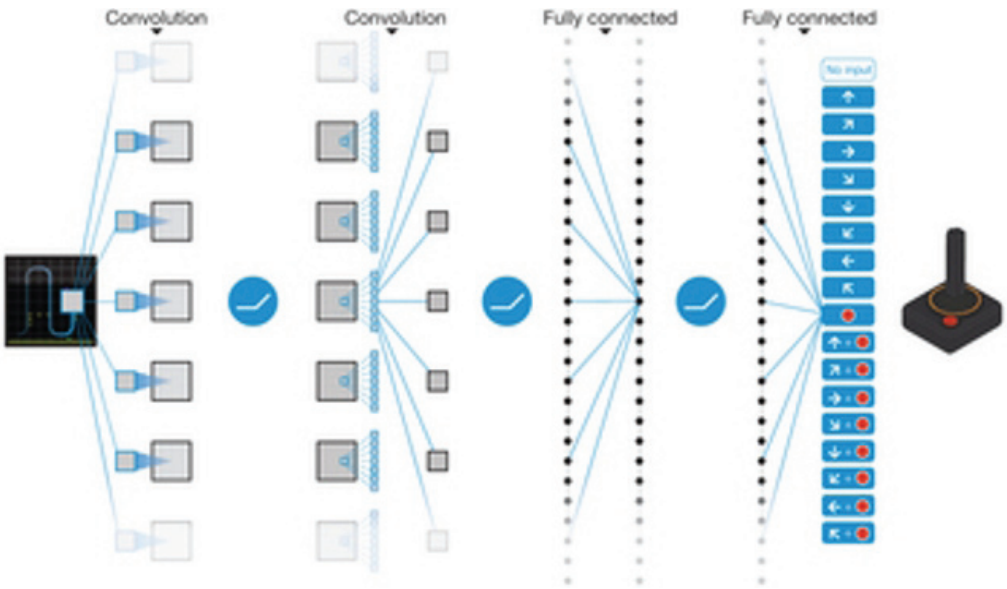
Redes neuronais tradicionais, de apenas duas camadas, nunca se mostraram viáveis para visão computacional, e foram praticamente abandonadas. Teoricamente as redes neuronais profundas, compostas por várias (10 ou mais) camadas empilhadas, vagamente inspiradas no **córtex visual dos mamíferos**, seriam úteis para esse tipo de problema. Mas ninguém sabia como treinar, usando aprendizado supervisionado, uma rede assim, e além disso seria necessária uma enorme quantidade de exemplos e um esforço computacional gigantesco para que ela aprendesse alguma coisa.

A AlexNet foi treinada com mais de um milhão de exemplos, gerados por centenas de pessoas através da Internet. Para reduzir drasticamente o tempo de treinamento o time usou placas de vídeo (GPUs) da **NVIDIA** para realizar todos cálculos, usando-as não para gerar gráficos 3D, como o usual, mas como supercomputadores. Mudanças nos algoritmos de treinamento e na arquitetura das redes resolveram as dificuldades que sempre impediram o uso de redes neuronais para classificação de imagens, e uma nova linha de pesquisa - "Aprendizado Profundo" (**Deep Learning**), numa alusão à profundidade / número de camadas da rede utilizada, surgiu.

Nos anos seguintes praticamente todos os competidores passaram a usar abordagens semelhantes, e subitamente o que antes era considerado um sonho distante - um computador **descrever uma imagem usando frases completas**, como "criança de vestido vermelho brincando num balanço", passou a algo viável. O grande desempenho das placas de vídeo nessa tarefa motivaram a criação de modelos dedicados para essa tarefa e novos algoritmos e frameworks de aprendizado começaram a ser desenvolvidos num ritmo muito rápido: o aprendizado profundo passou a ser usado para melhorias expressivas em outras tarefas, como **reconhecimento de voz**, tradução on-line e processamento de linguagem natural (empresas como Google, Facebook, Baidu, Microsoft e Apple hoje têm departamentos de pesquisa específicos para deep learning).

A grande contribuição do aprendizado profundo para o aprendizado supervisionado que já conhecemos é a **capacidade de abstração**: a primeira camada da rede recebe os pixels da imagem, e extrai automaticamente deles características muito simples: presenças de linhas horizontais e verticais, regiões de alto contraste, regiões de cores diferentes. As **camadas seguintes combinam esses elementos simples** para extrair novas informações mais genéricas e complexas, como formas geométricas e outros padrões. Prosseguindo nas diversas camadas as características encontradas são mais e mais sofisticadas (capazes de detectar olhos, rostos, rodas, objetos simples). O algoritmo de aprendizado supervisionado usado na última camada consegue aprender facilmente as categorias porque se baseia justamente nas **características de alto nível obtidas no final** (se foram detectados olhos, nariz, chupeta, tecido cor de rosa é fácil inferir que se trata de um bebê do sexo feminino).

Ao usar um **algoritmo de aprendizado por reforço na última camada**, e rodar o treinamento milhões de vezes, a rede neuronal profunda aprende a reconhecer todos os elementos visuais presentes no jogo (por exemplo, diferenciar inimigos de aliados, >



mesmo se tiverem a mesma cor), e o aprendizado por reforço faz com que o agente aprenda a reagir a esses elementos de forma a maximizar a recompensa obtida.

O resultado é literalmente um sistema que aprende a jogar Atari como eu e você: olhando para a tela, aprendendo o que cada um dos gráficos minimalistas significa, explorando o jogo para descobrir o que fazer para aumentar a sua pontuação, e usando esse conhecimento para jogar de forma competente. Quem já passou horas a fio aprendendo a jogar um novo título de Atari sabe que não é uma tarefa fácil!

J80

Referências:

O artigo publicado na Nature (2015):
Human-level control through deep reinforcement learning
http://www.nature.com/nature/journal/v518/n7540/full/nature14236.html

Uma descrição detalhada de Pac-Man, incluindo as regras que governam o comportamento dos fantasmas:
The Pac-Man Dossier
http://home.comcast.net/~jpittman2/pacman/pacmandossier.html

Dissertação de mestrado (2009) sobre aprendizado por reforço para jogos de Atari 2600:
Game-independent AI agents for playing Atari 2600 console games
https://era.library.ualberta.ca/public/view/item/uuid:12a46386-fcbb-4c18-9466-f8b3bb4baa1a

Ambiente para experimentos com aprendizado por reforço usando emuladores de videogame:
Arcade Learning Environment
http://www.arcadelearningenvironment.org/

Técnicas mais sofisticadas de aprendizado por reforço para jogos de Atari:
A Neuroevolution Approach to General Atari Game Playing (2013)
http://www.cs.utexas.edu/~ai-lab/?atari

Análise do algoritmo da DeepMind, reprodução da implementação em código aberto:
Artificial General Intelligence that plays Atari video games: How did DeepMind do it?
http://robohub.org/artificial-general-intelligence-that-plays-atari-video-games-how-did-deepmind-do-it

Artigo descrevendo a primeira rede neuronal profunda para classificação de imagens:
ImageNet Classification with Deep Convolutional Neural Networks
http://www.cs.toronto.edu/~fritz/absps/imagenet.pdf

Material de referência e tutoriais sobre aprendizado profundo na NVIDIA:
Deep Learning at NVIDIA
https://developer.nvidia.com/deep-learning

Página do concurso de reconhecimento de imagens da ImageNet:
ImageNet Large Scale Visual Recognition Challenge 2014
http://image-net.org/challenges/LSVRC/2014/index