ISSN 1984-5693
Vol.13, 2021
http://dx.doi.org/dx.doi.org/10.22407/1984-5693.2021.v13.p.27-34

SIMGPI 4º SIMPÓSIO GESTÃO DA PRODUÇÃO INDUSTRIAL PRODUÇÃO E MANUFATURA AVANÇADA

Perspectivas da Ciência e Tecnologia

Artigos Científicos

A IMPORTÂNCIA DOS ACELERADORES DE HARDWARE EM PROJETOS QUE USAM INTELIGÊNCIA ARTIFICIAL

The importance of hardware accelerators in projects using artificial intelligence

Gabriel Simas Gomes Barata[1], Ingrid Teixeira do Nascimento[1*], Filipe Pereira Mesquita dos Santos[1], Ana Carla de Souza Gomes dos Santos[1,2], Genildo Nonato Santos[1]

[1] Instituto Federal de Educação, Ciência e Tecnologia do Rio de Janeiro (IFRJ) *campus* Nilópolis, RJ. Brasil.

[2] Centro Federal de Educação Tecnológica Celso Suckow da Fonseca (CEFET) *campus* Maracanã, RJ. Brasil.

Submetido em: 18-01-2021. Aceito em: 21-01-2021. Publicado em: 13-04-2021.

***Autor para correspondência:** ingridteixeira22@gmail.com

Resumo: A aplicação da inteligência artificial permite automatizar processos de uma forma nunca vista antes e criar soluções muito mais operacionais. Essa automatização é desejada em diversos projetos com aplicações em muitas áreas. Os projetos de pesquisa do Instituto Federal do Rio de Janeiro poderiam usar essa inteligência artificial em suas metodologias para automatizar os procedimentos, o que permitiria mais eficiência e alcançando assim melhores resultados. Contudo a falta de maquinário adequado para trabalhar com inteligência artificial compromete tal abordagem de operação. Esse trabalho mostra uma comprovação de como o maquinário pode inviabilizar operações com inteligência artificial por apresentar dados gerados em um estudo de caso. No estudo, a falta de maquinário adequado limitou a eficiência da inteligência artificial em 62% de acurácia.

Palavras-chave: aceleradores de hardware; inteligência artificial; longo tempo de treino.

Abstract: The application of artificial intelligence allows us to automate processes in a way never seen before and to create much more operational solutions. This automation is desired in several projects with applications in many areas. Research projects at the Federal Institute of Rio de Janeiro could use this artificial intelligence in their methodologies to automate procedures, which would allow for more efficiency and thus achieve better results. However, the lack of adequate machinery to work with artificial intelligence undermines such an approach to operation. This work shows proof of how the machinery can make artificial intelligence operations unfeasible by presenting data generated in a case study. In the study, the lack of adequate machinery limited the efficiency of artificial intelligence to 62% accuracy.

Keywords: hardware accelerators; artificial intelligence; long training time.

ISSN 1984-5693
Vol.13, 2021
http://dx.doi.org/dx.doi.org/10.22407/1984-5693.2021.v13.p.27-34

INTRODUÇÃO

As redes neurais, uma classe de algoritmos de inteligência artificial, em geral as do tipo *deep learning* que são as mais funcionais dentro dessas classes, apresentam custos computacionais pesados em sua fase de desenvolvimento, o que impõe aos desenvolvedores restrições elevadas quanto às especificações de suas plataformas computacionais (JAY LEE *et al*., 2018).

O modelo mais básico de rede neural vem do conceito de neurônio artificial, conhecido como *perceptron, que* foi inventada em 1958 por Frank Rosenblatt no Cornell *Aeronautical Laboratory*. Já o mais complexo, as *deep learnings* ou redes convolucionais profundas, o modelo em si é quase que completamente inspirado no funcionamento do cérebro biológico. Nas *deep learnings* existe um compromisso entre a eficiência de se realizar uma tarefa e a dimensão da rede, quanto maior a rede mais eficiente ela é. Além disto, existe também uma relação direta entre custo computacional e as dimensões da rede. Assim, para desenvolver uma inteligência artificial (com redes neurais) que realize tarefas com elevada eficácia, é preciso um hardware de desenvolvimento de alta capacidade computacional (PITUKH *et al*., 2004).

No caso da maioria dos campi dos Institutos Federais de Educação, Ciência e Tecnologia do Rio de Janeiro (IFRJ), como é o caso do campus Nilópolis, esse tipo de hardware não se encontra disponível, o que compromete o desenvolvimento de projetos na área de inteligência artificial aplicando algoritmos do tipo *deep learning*. Neste artigo, mostraremos um caso prático de como a falta de hardware necessário compromete o treinamento de uma rede neural do tipo *deep learning*.

O objetivo do trabalho é o de apresentar as diversas fases de desenvolvimento de uma inteligência artificial do tipo *deep learning*, de dimensões médias, capaz de reconhecer e classificar lixo reciclável em imagens capturadas por câmeras de vídeo. E durante esse processo, demonstrar que é inviável a utilização de computadores sem modernos recursos de aceleração de gráficos por meio de uma caracterização de um computador que é compatível aos que estão disponíveis no *campus* Nilópolis do IFRJ.

Fundamentação teórica

Redes neurais são algoritmos da classe das inteligências artificiais e em geral possuem uma entrada de dados, um conjunto intermediário de camadas de processamento, que realizam operações matemáticas baseadas em um conjunto de pesos pré-definidos, e uma saída onde a rede entrega os dados já manipulados. Uma rede neural é um aproximador (computador) universal de qualquer função matemática, assim, dada uma determinada entrada de M dimensões, a rede apresenta uma determinada previsão em sua saída de N dimensões, seguindo um mapeamento M para N já pré-determinado pelo conjunto de pesos (BUCK *et al.*, 2007). O treino de uma rede neural consiste em encontrar um conjunto de pesos específico para que o mapeamento M para N ocorra de forma aproximada e que envolva um erro aceitável. O procedimento de treino consiste em usar um algoritmo otimizador (de pesquisa operacional) para procurar o conjunto de pesos que minimiza o erro. O otimizador é guiado no sentido contrário ao gradiente do erro entre a previsão feita na saída da rede para uma determinada entrada e a saída do mapa M para N previamente conhecido. Os mapas M para N são chamados de bases de treino e podem ser encontrados já montados na internet ou podem ser customizados para uma aplicação específica. Em uma rede neural é possível afirmar que a quantidade de pesos é diretamente proporcional ao tempo necessário, chamado de tempo de treino, da busca para encontrar o conjunto de pesos adequados para aproximar o mapeamento M para N. Para avaliar a fase de treino, são

ISSN 1984-5693
Vol.13, 2021
http://dx.doi.org/dx.doi.org/10.22407/1984-5693.2021.v13.p.27-34

de treino são usadas métricas que controlam o erro associado às estimativas feitas pela rede neural em relação a base de treino, chamada de acurácia (uma medida complementar ao erro), o tamanho dos subconjuntos da base de treino que serão avaliados em cada rodada de processamento, o batch, e o erro propriamente dito (ZHAI *et a*l., 2017).

Para facilitar no desenvolvimento de inteligências artificiais com redes neurais costuma-se usar bibliotecas de programação específicas para esse fim. Essas bibliotecas são instaladas em ambientes de programação genéricos, tornando todo o conjunto um ambiente de desenvolvimento integrado para trabalhar com redes neurais. Em redes neurais, existem algumas bibliotecas muito conhecidas, como por exemplo, o Tensor Flow, NLTK e Scikit-Learn. Dentre estas, a que tem mais destaque é o Tensor Flow, que se encontra disponível para o ambiente de programação Python 3 (GU *et al.*, 2013).

O trabalho com redes convolucionais profundas em geral requer muito poder computacional devido à quantidade elevada de pesos nas camadas de processamento e também pela quantidade de dados da base de treino. Por exemplo, uma rede convolucional profunda de arquitetura AlexNet costuma ter 5 milhões de parâmetros e uma base de treino pequena 2 mil imagens. Os computadores de desenvolvimento costumam ter potentes sistemas aceleradores de hardware do tipo GPU (Unidade de Processamento Gráfica). A GPU possui um melhor desempenho que os clássicos processadores (CPU) por sua capacidade de realizar processamento em paralelo e distribuído de maneira massiva. Não entraremos nessa discussão nesse trabalho, mas quem quiser se aprofundar nesse tema pode acompanhar uma discussão específica sobre esse assunto em (SHAAFIEE *et al.*, 2017).

METODOLOGIA

A presente metodologia foi dividida em duas partes para permitir facilitar a didática de apresentação que visa resolver as questões objetivas deste trabalho. Em primeiro lugar foi levantada a possibilidade de se desenvolver uma inteligência artificial capaz de reconhecer e classificar imagens de lixo reciclável. Nessa parte foi feito um estudo para selecionar ferramentas de software que seriam necessárias para tal tarefa. O algoritmo escolhido para a inteligência artificial foi a de estrutura de rede neural convolucional. O primeiro passo da criação da inteligência artificial foi a escolha da linguagem de programação, o *python*. A escolha se deu devido à versatilidade na manipulação de dados de imagem e de mídia no geral e também pela variedade de ferramentas computacionais disponíveis para essa linguagem. Como exemplo, podemos citar a biblioteca *Tensorflow* que é usada para trabalhar com redes neurais no geral. O segundo passo foi o de escolher um ambiente de desenvolvimento integrado. O ambiente escolhido foi o jupyter notebook devido o seu design simplificado, que é baseado em células onde rodam trechos de código. Esse design permite mudar a ordem das células, alterando o trecho de código que irá rodar no momento ou mesmo executar um mesmo trecho mais de uma vez, se necessário. Outra grande vantagem desse ambiente é a não necessidade de executar o código inteiro caso um certo trecho seja alterado. Por último foi definido a base de dados de treino. Dentre as bases de dados que se encontram disponíveis na internet, foi encontrada uma chamada *Trashnet*, que consistia em um conjunto de imagens de lixo reciclável e não reciclável, organizadas por classes. As classes foram denominadas originalmente como "*cardboard", "glass", "metal", "paper", "plastic*" e "*trash*", que continham imagens de, respectivamente, 403 de caixas de papelão, 501 de garrafas e copos de vidro, 410 de latas de metal, 594 de pedaços de papel, 482 de garrafas e embalagens de plástico. Por fim, 137 de lixo não reciclável. Dessa forma, todas as etapas necessárias para desenvolver a rede neural foram descritas e detalhadas na Figura 1.

ISSN 1984-5693
Vol.13, 2021
http://dx.doi.org/dx.doi.org/10.22407/1984-5693.2021.v13.p.27-34

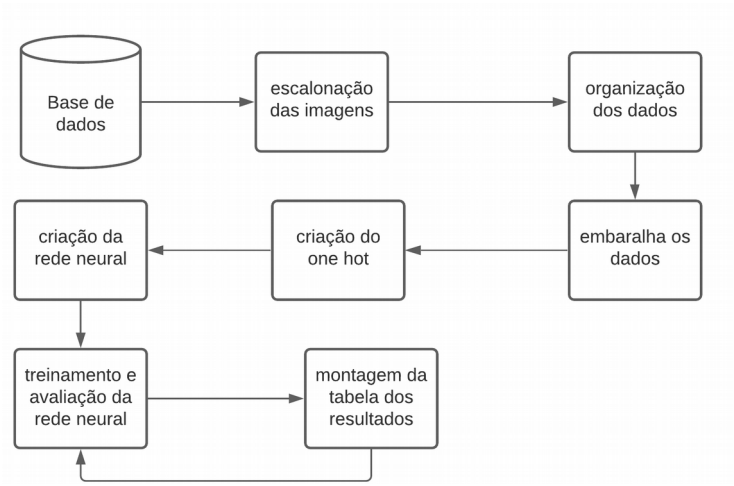

Figura 1. Fluxograma do algoritmo desenvolvido para a avaliação.

A outra parte do trabalho consistiu em responder a questão de se é viável treinar uma inteligência artificial que classifica lixo reciclável usando computadores pessoais sem aceleradores de hardware modernos, como aqueles disponíveis no IFRJ - Nilópolis. Assim, algumas métricas para medir o tempo de treino e a acurácia de treino foram adicionadas ao algoritmo apresentado. Para medir o desempenho do computador que é relacionado a quantidade de parâmetros máximos computados por segundo, foram feitas 3 cópias do algoritmo escolhido para a rede neural e cada um deles foi alterado de maneira que o primeiro tivesse 30.000 parâmetros, o segundo 300.000 parâmetros e o terceiro 3.000.000 de parâmetros. Isso possibilitou fazer estimativas mais precisas sobre a capacidade de computar parâmetros do computador usado.

Na presente metodologia, usando a biblioteca *Tensorflow*, os 3 algoritmos desenvolvidos foram executados, o tempo de treino e a acurácia foram medidos para cada uma das redes e para uma combinação de épocas e *batches* (que serão melhor descritos no capítulo Resultados). Os resultados foram armazenados após a execução dos algoritmos e esses dados foram usados para construir uma tabela e um gráfico de estimativas de desempenho que também serão posteriormente apresentados.

RESULTADOS E DISCUSSÃO

Em nosso experimento foram desenvolvidas três redes neurais convolucionais a fim de medir o limite da capacidade de processamento do computador utilizado. A primeira com cerca de 30.000 parâmetros, a segunda com cerca de 300.000 parâmetros e a terceira com cerca de 3.000.000 parâmetros. Uma base de treino contendo cerca de 2.200 imagens foi usada. A base foi baixada do repositório github.com e para o treino foi dividida em duas, 70% para o treinamento (base de treino) e 30% para a validação (base de validação). O teste consistiu de variar o número de épocas em 5, 10 e 20 e o batch de 300, 100 e 30 em cada uma das redes neurais. Aplicando cada uma dessas configurações, foram medidos o tempo necessário para o computador terminar a fase de treino, a acurácia considerando a base de treino e de validação e o erro considerando a base de treino e de validação. Os resultados da primeira e da segunda rede podem ser vistos nas Tabelas 1 e 2. O computador se mostrou inadequado para operar com a última rede e esses resultados não aparecem. O processo de treino inteiro ocorreu utilizando o CPU do computador (DELL INSPIRON, modelo 14 2640 com uma CPU i5 de terceira geração e memória RAM de 6 GB).

ISSN 1984-5693
Vol.13, 2021
http://dx.doi.org/dx.doi.org/10.22407/1984-5693.2021.v13.p.27-34

Tabela 1. Resultados do teste de treino da rede neural de 30.880 parâmetros

Épocas	Batch	Tempo (s)	Acurácia	Acurácia de validação	Erro	Erro de validação
5	300	156,00	0,23535156	0,26220703	0,44042968	0,43505859
10	300	296,00	0,28857421	0,29785156	0,42382810	0,42041015
20	300	601,50	0,36132812	0,34130859	0,40844726	0,40454101
5	100	154,00	0,43725585	0,41381835	0,38281250	0,38281250
10	100	305,75	0,49780273	0,44262695	0,35571289	0,36767578
20	100	606,50	0,57373046	0,49536132	0,31054687	0,34814453
5	30	167,50	0,52294921	0,48364257	0,33422851	0,34472656
10	30	339,25	0,53710937	0,46630859	0,32299804	0,35595703
20	30	670,50	0,62011718	0,51269531	0,28515625	0,33886718

Tabela 2. Resultados do teste de treino da rede neural de 292476 parâmetros

Épocas	Batch	Tempo (s)	Acurácia	Acurácia de validação	Erro	Erro de validação
5	300	1112	0,22961425	0,24768066	0,4702148	0,45410156
10	300	2182	0,22961425	0,24768066	0,4384765	0,43530273
20	300	4340	0,22961425	0,24768066	0,4387207	0,43505859
5	100	1079	0,22961425	0,24768066	0,4392089	0,43505859
10	100	2116	0,22961425	0,24768066	0,4392089	0,43481445
20	100	4428	0,22961425	0,24768066	0,4389648	0,43554687
5	30	1114	0,22448730	0,24768066	0,4394531	0,43603515
10	30	2222	0,22961425	0,24768066	0,4392089	0,43505859
20	30	4496	0,22961425	0,24768066	0,4392089	0,43579101

Ambas as redes foram inicializadas a partir de parâmetros aleatórios. Tendo em vista a máxima conservação da informação de entrada da rede, optou-se por utilizar imagens coloridas e no formato 50 x 50 nas bases de treino. Os erros foram calculados usando a metodologia de erro médio quadrático.

ISSN 1984-5693
Vol.13, 2021
http://dx.doi.org/dx.doi.org/10.22407/1984-5693.2021.v13.p.27-34

Analisando os resultados apresentados na Tabela 1, pôde-se perceber que houve um declínio no erro de treino e de validação ao longo das épocas e uma ascensão na acurácia e na acurácia de validação, tal qual era o objetivo do treino. Na Tabela 2, onde era esperado que houvesse uma redução nos erros superior ao dos dados da Tabela 1, devido ao aumento no número de parâmetros. Porém, as tendências de redução dos erros não foram mantidas. Em relação ao tempo de treino fica claro que reduzindo o batch, aumentando a época e a quantidade de parâmetros, o tempo de treino aumenta também. Como exemplo, na Tabela 2, em batch 30 e época 20 a rede de cerca de 300.000 levou quase 1 hora e 30 minutos para concluir o treino, atingindo acurácia de cerca de 20%. Em estimativa feita, considerando que o tempo de treino é diretamente proporcional ao número de parâmetros, a terceira rede neural demoraria por volta de 74 horas (3 dias) para concluir seu treino. O teste apresentado mostra que pode ser muito complicado operar com computadores contando apenas com o recurso de CPUs quando se está trabalhando com inteligência artificial do tipo Deep Learning.

A máquina utilizada para o treinamento possui uma placa de computação gráfica Geforce GT620M com 1 GB de memória (considerada antiga). Contudo, essa placa de vídeo é incompatível com o software CUDA, e por isso, inacessível a plataforma *Tensor Flow*. Mas acredita-se que com acesso a uma placa gráfica mais atual os tempos de treino apresentados cairiam drasticamente.

CONCLUSÕES

Neste trabalho foi apresentado um estudo de caso que exemplificou o desenvolvimento de uma inteligência artificial capaz de identificar itens recicláveis (lixo) em imagens. O desenvolvimento consistiu da instalação da biblioteca *Tensor Flow*, capaz de desenvolver, treinar e testar redes neurais do tipo *Deep Learning*. Para possibilitar a utilização da biblioteca foi também instalada a plataforma Anaconda que disponibilizou os itens Python3 e Jupyter Notebook. Foram usadas duas diferentes arquiteturas para o modelo de rede neural, uma com 30.000 parâmetros de treino e outra com 300.000 parâmetros de treino. Ambos os modelos foram treinados com uma combinação de 6 diferentes combinações de épocas e *batches*. Para o treino foi usado a base de dados *Trash-Net* da aluna Mindy Yang da "*Stanford's CS 229: Machine Learning Class*", disponível no site de programadores independentes **github**. Um modelo com cerca de 3.000.000 de parâmetros foi testado. Contudo, o tempo de treino se mostrou proibitivo (74 horas) para a configuração de hardware disponível no momento.

De acordo com os dados mostrados nos resultados do presente trabalho, o melhor desempenho encontrado foi a do modelo de 30.000 parâmetros que alcançou cerca de 62% de acurácia em uma configuração de treino com épocas de 20 e *batches* de 5. Mais detalhes podem ser encontrados no texto.

Era esperado que o modelo com mais parâmetros atingisse o melhor desempenho, contudo isso não ocorreu. Existem diversas hipóteses que foram formuladas para que isso não ocorresse. Por exemplo, a quantidade de épocas para o treino do modelo maior não foi suficiente e que esse quantitativo poderia ter sido estendido mantendo a quantidade de batches em seu menor patamar (5). Contudo, tais verificações não foram feitas devido ao fato que o problema mais relevante da discussão já havia sido exposto. Os tempos elevados de treino que foram mostrados refletem muito a precariedade de equipamentos que se tem acesso para trabalhar com inteligência artificial no IFRJ/CNIL. Como no estudo de caso demonstrado, aumentar a quantidade de épocas significaria aumentar os tempos de treino dos modelos e assim submeter as máquinas disponíveis para esse propósito a um superaquecimento e risco de avaria.

ISSN 1984-5693
Vol.13, 2021
http://dx.doi.org/dx.doi.org/10.22407/1984-5693.2021.v13.p.27-34

Dessa maneira concluímos aqui que máquinas como a que temos não são adequadas para trabalhar com esse tipo de modelo computacional e quando os IFs estão discutindo a montagem de laboratórios de inovação e de tecnologia, equipamentos adequados para esse fim, como computadores que tenham em suas configurações processadores gráficos de alta capacidade (GPU's e TPU's) são essenciais assim como cortadoras a laser e impressoras de 3 dimensões. Como trabalho futuro, é pretendida a obtenção de placas de processamento gráfico de alta capacidade para permitir o treino dos modelos de muitos parâmetros (acima de 300.000) da maneira adequada e assim concluir o presente estudo.

REFERÊNCIAS

BUCK, I. GPU Computing: Programming a Massively Parallel Processor. **International Symposium on Code Generation and Optimization (CGO'07).** San Jose: CA 2007, p. 17-17. doi: 10.1109/CGO.2007.13 Disponível em:< http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=4145100&isnumber=4145090>.

CADAMBI, S; DURDANOVIC, I *et al.*. A Massively Parallel FPGA-Based Coprocessor for Support Vector Machines. **17th IEEE Symposium on Field Programmable Custom Computing Machines.** Napa: CA. 2009, 115-122. doi: 10.1109/FCCM.2009.34. Disponível em: <http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=5290941&isnumber=5290880>.

FRENKEL, C; LEFEBVRE, M; LEGAT J; BOL, D. A 0.086-mm^2 12.7-pJ/SOP 64k-Synapse 256-Neuron Online-Learning Digital Spiking Neuromorphic Processor in 28-nm CMOS. **IEEE Transactions on Biomedical Circuits and Systems 13**(1), 2019, 145-158. doi: 10.1109/TBCAS.2018.2880425 Disponível em:< http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=8528875&isnumber=8626744>.

GU, R; SHEN, F; HUANG, Y. A parallel computing platform for training large scale neural networks", 2013 **IEEE International Conference on Big Data.** Silicon Valley: CA. 2013, 376-384. doi: 10.1109/BigData.2013.6691598 Disponível em: <http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=6691598&isnumber=6690588>.

HENNESSY, JL; PATTERSON, DA. **Computer Architecture: A Quantitative Approach**. 3rd Edition, Morgan Kaufmann Publishers (Elsevier). Burlington, 2002. ISBN:1-55860-596-7.

LEE, J; HD; DAVARI, H; SINGH, J; PANDHARE V. Industrial Artificial Intelligence for industry 4.0 - based manufacturing systems. **Manufacturing Letters 18**, 2018, 20-23. https://doi.org/10.1016/j.mfglet.2018.09.002. Disponível em:<http://www.sciencedirect.com/science/article/pii/S2213846318301081>.

OH, HJ; MUELLER, SM *et al.* A fully pipelined single-precision floating-point unit in the synergistic processor element of a CELL processor. **IEEE Journal of Solid-State Circuits 41**(4), 2006, 759-771. doi: 10.1109/JSSC.2006.870924. Disponível em:< http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=1610620&isnumber=33823>.

ISSN 1984-5693
Vol.13, 2021
http://dx.doi.org/dx.doi.org/10.22407/1984-5693.2021.v13.p.27-34

PITUKH, I; NIKOLAYCHUK Y; VOZNA, N. Principles of computer networks construction with deep paralleling of information flows on the basis of matrix models of data movement," Proceedings of the International **Conference Modern Problems of Radio Engineering, Telecommunications and Computer Science**, Lviv-Slavsko: Ukraine. 2004, pp. 417-419. Disponível em: <http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=1366011&isnumber=29914>.

SHAAFIEE, M; LOGESWARAN, R; SEDDON, A. Overcoming the limitations of von Neumann architecture in big data systems. 7th International Conference on Cloud Computing, Data Science & Engineering. **Confluence**, 2017, 199-203. doi: 10.1109/CONFLUENCE.2017.7943149. Disponível em: <http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=7943149&isnumber=7943112>.

ZHAI, X; JELFS, B; CHAN, RHM; TIN, C. Self-Recalibrating Surface EMG Pattern Recognition for Neuroprosthesis Control Based on Convolutional Neural Network. **Frontiers in Neuroscience 11.** 2017. 10.3389/fnins.2017.00379.

ZHONG, RY; XU X; KLOTZ, E; NEWMAN, ST. Intelligent Manufacturing in the Context of Industry 4.0: A Review. **Engineering 3**(5), 2017, 616-630. https://doi.org/10.1016/J.ENG.2017.05.015.