Universidade Federal de Santa Catarina
Centro Tecnológico - CTC
Departamento de Informática e Estatística - INE
Curso de Sistemas de Informação
Disciplina: Programação Paralela e Distribuída
Professor: Odorico Machado Mendizabal

Trabalho Final

Tensor Flow

Alunos:
Bruno Daniel Elias
Diovana Rodrigues Valim
Felipe Capistrano Maes
Gabriel Aristeu Cabral
Mariany Ferreira

06 de setembro de 2021

I. Introdução

Quando falamos em inteligência artificial, a primeira coisa que vem à cabeça são robôs conscientes tais quais o exterminador do futuro. Entretanto, o conceito de inteligência artificial pode ser trazido para uma realidade mais próxima da que vivemos não como uma fantasia futurista, mas aplicada a cenários onde seu uso gera benefícios significativos. Inseridos em uma sociedade globalizada que ainda desconhece os impactos concretos que a avalanche de informação pode causar, a inteligência artificial pode ser aplicada para análises de dados preditivas baseadas no aprendizado de máquina.

Inteligência artificial, um termo amplamente utilizado hoje em dia, esconde até certo ponto outros dois conceitos extremamente importantes para o desenvolvimento da espinha dorsal de um sistema inteligente. Existem diferenças entre os conceitos de IA, machine learning e deep learning, embora elas não sejam excludentes: muito pelo contrário, elas complementam umas às outras. Eles tem a ver com Inteligência Artificial, porque a IA em si pode ser considerada um termo genérico. Derivado da computação cognitiva, trata-se de um "guarda-chuva" sobre abordagens diferentes para criar um sistema autônomo, capaz de gerir as suas próprias regras e agir de acordo com elas.

Assim, pode-se dizer que todo Machine Learning é um tipo de IA. Mas nem toda IA é baseada em Machine Learning. Já o Deep Learning é uma das técnicas utilizadas para que a máquina consiga interpretar dados e aprender com eles.

As fronteiras do aprendizado de máquina, conceito fundamental a ser definido quando falamos de inteligência artificial, entretanto, ainda são um pouco vagas. Dizer que uma máquina é dotada de inteligência, isto é, capaz de aprender algo, é antes de tudo, compreender que computadores podem ser programados para aprender e extrair informações de dados. Segundo a definição de 1959 de Arthur Samuel, o aprendizado de máquina é o campo de estudo que dá aos computadores a habilidade de aprender sem ser explicitamente programado. Inteligências artificiais estruturadas sobre redes neurais profundas e algoritmos de aprendizagem são capazes de analisar e descobrir padrões ocultos em conjuntos de dados não estruturados, revelando informações antes desconhecidas.

Uma máquina dotada de inteligência é, antes de tudo, um sistema matemático extremamente preciso que trabalha analisando dados para gerar conclusões. Se você treina um modelo de aprendizado de máquina dizendo que os emails que contêm, no corpo ou no título, as palavras "Para você", são geralmente marcados como spam, facilmente ele será capaz de reconhecer as variações desta abordagem bem como os padrões implícitos que dificilmente seriam abordados como regra de negócio em um algoritmo de decisão manual.

Tratando de inteligência artificial, muito dificilmente podemos prosseguir no tema sem falar sobre os conceitos de dados e informação. Um conjunto de dados é a reunião de valores que, a princípio, não possuem significado relevante ou sentido. Portanto, não tem valor algum para embasar conclusões, muito menos respaldar decisões. Entretanto, quando analisamos um conjunto de dados, é possível dele extrair algo extremamente valioso: a informação, que é a ordenação e organização dos dados de forma a transmitir significado e compreensão dentro de um determinado contexto. A abundância de dados que coletamos fornece às inteligências artificiais os exemplos de que eles precisam para identificar diferenças, aumentar suas capacidades de reconhecimento de padrões e identificar mesmo os menores detalhes dentro destes padrões encontrados.

Até pouco tempo atrás, armazenar e processar dados era uma tarefa difícil e extremamente custosa; entretanto, a modernização dos bancos de dados, que cada vez mais tornam-se poderosos e versáteis mecanismos de controle do fluxo de dados, proporcionou uma revolução nunca antes vista para a área de análise de dados. Se o armazenamento e o processamento de dados não caracteriza mais uma barreira, é completamente plausível trabalhar modelos probabilísticos extremamente precisos, alimentados com enormes conjuntos de dados. E é exatamente do treinamento de um modelo como este que derivam todas as conclusões que as grandes corporações do vale do silício tomam a respeito do conteúdo que elas mostram para nós; a estimativa é que, durante sua vida, um ser humano gere em torno de incríveis 1.7 megabytes de dados por segundo. A análise minuciosa que as máquinas são capazes de fazer sobre os dados do seu perfil do Facebook gera conclusões a partir das quais é possível definir com precisão qual matéria do Buzzfeed deve aparecer primeiro no seu feed de notícias.

II. A estrutura de um modelo de *Machine Learning* baseado em algoritmos de *Deep Learning*

Criar um sistema computacional voltado ao aprendizado de máquina é estruturar um modelo de decisão baseado em um algoritmo que pode ser melhorado via dados de treinamento, sem programação explícita. De acordo com a literatura envolvida nesta área da ciência da computação, existem inúmeras classificações aplicáveis a um sistema de aprendizado de máquina, das quais é possível pontuar três em particular, segundo o artigo *Princípios do aprendizado de máquina e da aprendizagem profunda,* da companhia que provê soluções quanto à análise de dados em grandes empresas, *PureStorage.*

- Aprendizagem supervisionada: São feitas entradas de rótulos e exemplos atuais a respectiva saída desejada e isso permite ao algoritmo aprender as regras que mapeiam entradas e saídas.
- Aprendizagem não supervisionada: Os rótulos não são fornecidos e, portanto, o algoritmo pode encontrar sua própria estrutura de

processamento das entradas (por exemplo, encontrando padrões ocultos nos dados).

- Aprendizagem por reforço: O algoritmo interage repetidamente com um ambiente dinâmico com um objetivo específico como, por exemplo, ganhar um jogo ou dirigir um carro. O algoritmo chega à solução mais otimizada para o problema por meio de repetidos erros e tentativas.

De alguma maneira, baseando-se em um algoritmo, uma máquina recebe grandes conjuntos de dados, a partir dos quais ela gera análises e respostas, independentes de fatores externos à ela. Fundamentando-se na ideia de aprendizado humano, constrói-se sistemas computacionais que partem de funções de probabilidade ou redes neurais para discernir sobre os dados a ele apresentados.

Por exemplo: Apresenta-se *n* fotos de gatos e *não* gatos a um sistema de aprendizado de máquina durante um dado intervalo de tempo. Depois de um período médio-longo, o algoritmo, baseando-se nos dados que ele recebeu como treinamento, consegue decidir o que *é gato* e aquilo que *não é gato*. Esta abordagem, apesar de semelhante ao aprendizado dos seres-humanos, visto que, só conseguimos identificar um gato porque já vimos muitos deles antes, ainda não representa a capacidade de assimilação e conexão que uma rede neural possui. O conceito de uma rede neural segue o mesmo caminho do compartilhamento de informações pelos neurônios do cérebro, isto é, é dotada da capacidade de adaptação que os seres humanos possuem.

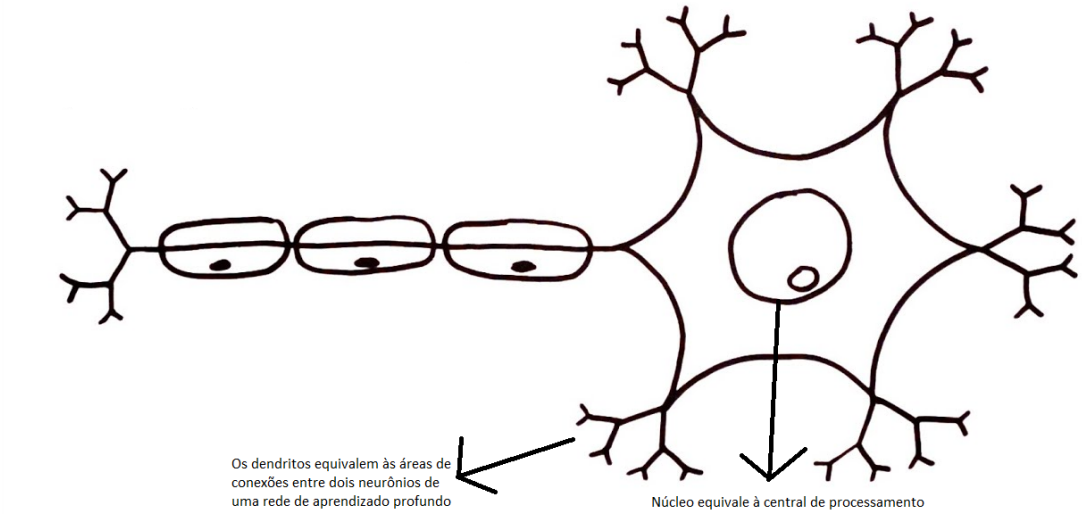

Imagem 01: Assimilação entre um neurônio real e o conceito de neurônio aplicado em redes de aprendizado profundo.

Uma rede neural, utilizada em sistemas de aprendizado de máquina baseados em aprendizagem não supervisionada, isto é, estruturados sobre algoritmos de *Deep Learning,* normalmente podem ser representadas por um processador paralelamente distribuído, constituído por inúmeras unidades mais simples de processamento. Nestas

unidades, está armazenado o conhecimento experimental. A função dos algoritmos de aprendizado profundo é modificar os pesos e bias, estruturas fundamentais que compõem um sistema de *Machine Learning,* de cada neurônio que estrutura a rede para alcançar um objetivo comum final.

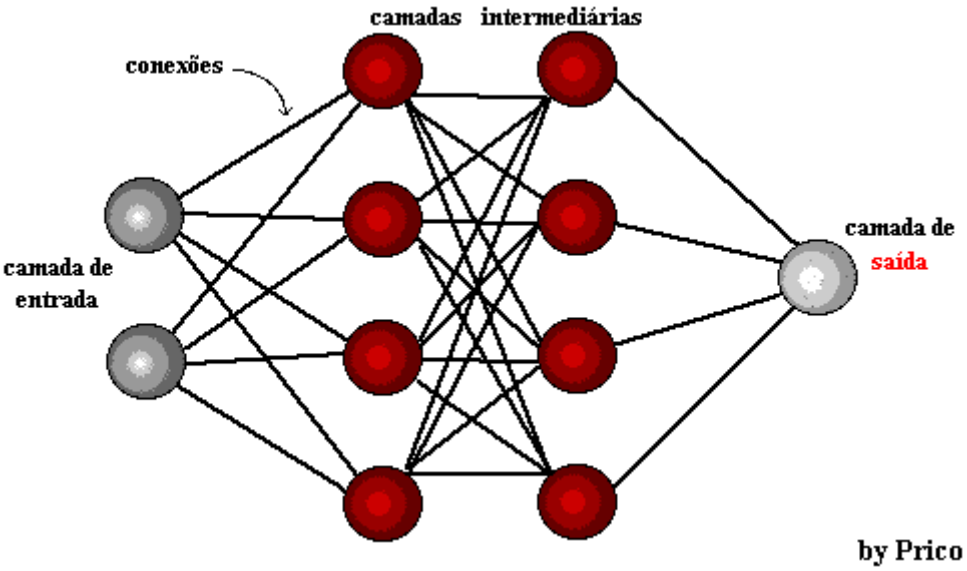

Imagem 03: Exemplo de uma arquitetura *Perceptron* multicamadas.

A função mais importante de uma rede neural é reconhecer os padrões de um modelo inserido no ambiente no qual ela está trabalhando. Dentro deste conceito, possuímos duas estruturas fundamentais na espinha dorsal de uma rede neural: a função de custo, e a função de ativação.

Uma função de custo está diretamente relacionada com a taxa de erro dentro da estrutura da rede neural. Procura-se sempre obter baixos valores de função de custo, e para isso, usa-se algoritmos de otimização.

Já a função de ativação é baseada no conceito de que redes neurais são, em sua essência, modelos matemáticos. A partir disso, podemos tomar como exemplo, para duas camadas ocultas:

$$y = \oslash(\oslash(X \times W1)W2)_{w}$$

Onde X é a matriz de dados, W1 e W2 são os pesos das camadas ocultas, W é o peso da camada de saída e $\oslash$ é uma função não linear qualquer, conhecida como *função de ativação.* Se retirarmos $\oslash$ da nossa equação, teremos apenas um modelo de regressão linear normal. Sendo assim, as funções de ativação servem para dar capacidade representativa às redes neurais artificiais, introduzindo ao modelo um componente de não linearidade, sendo aplicada na saída de cada nó da rede neural.

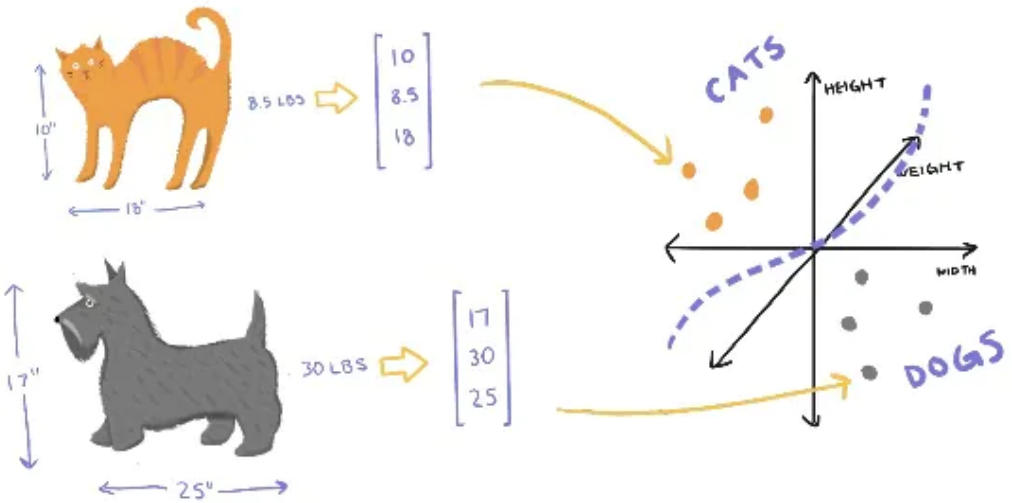

Imagem 05: Avaliação dos padrões de medidas de um gato e de um cachorro. Desenho feito por @dalequark, no *Twitter*.

III. O que é Tensor Flow?

O *Tensor Flow* é uma biblioteca poderosa especialmente utilizada para cálculo numérico, caracterizando-se como uma ferramenta adequada para o aprendizado de máquina em larga escala. Além disso, seu código-fonte é *open source,* isto é, pode ser livremente utilizado e visualizado por qualquer pessoa. A princípio, a regra de funcionamento do *Tensor Flow* é bastante simplista. Segundo o livro *Mãos à Obra: Aprendizado de Máquina com Scikit-Learn & Tensor Flow,* primeiro define-se um grafo de cálculos para ser executado em *Python*, e, em seguida, o *Tensor Flow* pega esse grafo e o executa de forma eficiente utilizando um código C++ otimizado[1].

Entretanto, a parte vital do funcionamento do *Tensor Flow* reside na possibilidade de dividir este grafo em *n* partes que poderão ser executadas de forma paralela em CPU e/ou GPU, acelerando o processamento do *dataset* analisado. A quantidade de dados utilizados na construção de um modelo de aprendizado de máquina é um fator extremamente importante para a corretude dos resultados, isto é, o nível de confiança que podemos ter neste modelo. Nas sociedades da atualidade, em que o acesso à internet torna-se quase uma necessidade, a quantidade de dados disponíveis para análise compõe valores colossais, cujo processamento sequencial derivaria custos altíssimos. Todavia, o *Tensor Flow* também suporta abordagens de computação distribuída, nas quais é possível treinar redes neurais profundas, dividindo a carga de processamento em inúmeras máquinas, desde que os resultados possam ser obtidos em um tempo relativamente razoável. A escalabilidade e a flexibilidade que o *framework* disponibiliza aos seus utilizadores são características essenciais na ferramenta da Google que garantiram sua rápida popularização desde que foi liberada ao código aberto. Tecnicamente, o uso do *Tensor Flow* não se limita às aplicações de *machine learning* estruturado sobre redes neurais profundas. “O *Tensor Flow* pode treinar uma rede com *milhões* de parâmetros em um conjunto de

treinamento composto por *bilhões* de instâncias, com *milhões* de características cada."[2]

É possível pensar no *Tensor Flow* sendo, de certa forma, uma linguagem de programação. Entretanto, é necessário destacar que ele é um *framework*, desenvolvido e aplicado inicialmente à linguagem *Python,* embora hoje em dia possua várias distribuições para múltiplos sistemas operacionais. Escrever um código utilizando esta ferramenta é praticamente escrever um código em *Python*. Depois de instalar o *framework* é possível misturar ambas as abordagens (*Python* + *Tensor Flow*) e executar tudo junto, utilizando o *Tensor Flow* em sua essência.

```
import tensorflow as tf

hello_world = tf.constant("Hello, world!")

with tf.Session() as sess:
    run = sess.run(hello_world)

print(run)
```

Imagem 05: Exemplo de código *Olá mundo* utilizando *Python* + *Tensor Flow.*

O uso do *Tensor Flow* como uma biblioteca *Python* requer que, antes da execução do código, seja definido um *backbone* daquilo que será rodado pelo interpretador. A partir deste, é possível executar de fato as instruções escritas, através da criação de uma sessão. Na prática, é criado um grafo computacional, e é possível observar que, enquanto a sessão *Tensor Flow* não for aberta, os dados não irão fluir através das instruções.

Partindo desta ideia, é possível criar um pequeno exemplo de rede neural utilizando o *Tensor Flow.*

```
# layer 1 weight
w1 = tf.Variable(tf.random_normal([n_input, n_hidden_1]))
# layer 1 bias
b1 = tf.Variable(tf.random_normal([n_hidden_1]))
# sigmoide function in layer 1
camada_1 = tf.nn.sigmoid(tf.add(tf.matmul(x,w1),b1))
```

Imagem 06: Criação de uma rede neural de uma camada utilizando *Python* + *Tensor Flow.*

O *Tensor Flow* é uma biblioteca que nos permite criar uma rede neural utilizando pouquíssimas linhas de código, abordando um design simplista que também caracteriza um fator positivo ao seu uso. Este é um *framework* extremamente poderoso, que provê uma ampla gama de recursos teóricos necessários para que se implemente modelos de aprendizado de máquina baseados em redes neurais profundas, como por exemplo, funções pré-definidas para estruturar qualquer tipo de

redes neurais. Em outras palavras, a parte mais difícil do desenvolvimento é identificar o que acontece no processamento feito pelo *Tensor Flow,* ou seja, entender como decorre o processo de aprendizado da rede neural.

IV. Para que serve o Tensor Flow?

As áreas de estudo do aprendizado de máquina, especialmente daquelas que utilizam algoritmos de *Deep Learning*, são bastante complexas. Algoritmos aplicados a este campo da ciência demandam um sólido conhecimento de matemática e estatística, porque o processo de aprendizado de um sistema computacional é totalmente baseado no cálculo de probabilidades baseado no reconhecimento de padrões. Este tipo de algoritmo é capaz de fazer predições a partir de amostras, ou ainda, tomar decisões que são orientadas apenas por dados, sem que qualquer tipo de programação seja envolvida no processo. Embora pareça semelhante da estatística computacional, é importante salientar que o aprendizado de máquina é usado especialmente em tarefas computacionais cuja dificuldade atinge graus nos quais programar algoritmos explícitos é praticamente impossível.

Não como uma bala de prata, mas sim como um coringa que cobre e facilita muitos cenários da aplicação do aprendizado de máquina apoiado em algoritmos de aprendizado profundo, o *Tensor Flow* é uma das formas mais simples de treinar um modelo de *Machine Learning*, através da API *TF.Learn,* uma interface de programação de alto nível que é compatível com a biblioteca *Scikit-Learn*, do *Python.* O *Tensor Flow* é um *framework* que surge para reduzir a complexidade do aprendizado e aplicação dos conceitos relacionados à Inteligência Artificial, disponibilizando uma série de algoritmos e funções prontas para serem aplicadas sobre os mais complexos tipos de dados.

Este *framework* facilita a construção de redes neurais profundas onde trabalha-se com muitos dados - especialmente aqueles sem uma estrutura declarada -, um problema complexo, tal qual fala, visão ou processamento de linguagem natural, e onde é necessário um modelo extremamente preciso. Além de simplificar a estrutura de um modelo de aprendizado de máquina, o *TS* também é extremamente útil para o processamento de grandes volumes de dados, através da aplicação de programação distribuída.

Atualmente, muitas aplicações utilizam o *Tensor Flow* para criar sistemas inteligentes, entre os quais destacam-se alguns exemplos voltados ao cuidado com a saúde, sistemas de recomendações, anúncios personalizados, entre muitos outros, normalmente voltados à classificação, percepção, entendimento, descobrimento, predição e criação. Em certos cenários, alguns dos exemplos anteriormente citados não seriam possíveis sem as facilidades implementadas pelo *TS.*

V. Por que usar o Tensor Flow?

Existem diversos fatores que podem influenciar no uso de uma ferramenta, tanto internos: a estrutura e os métodos de uso, quanto externos: a empresa que a criou e a relação com o resto do mercado. Um grande fator externo a favor do Tensor Flow é o fato de ter sido desenvolvido pela Google Brain, garantindo um selo de qualidade Google ao produto.

Em relação ao uso da ferramenta em si, a questão mais atrativa aos consumidores com certeza é a sua sintaxe, considerada extremamente mais acessível e legível se comparado com os concorrentes, como citado anteriormente. Andando de mãos dadas com esta questão da simplicidade está o fato do Tensor Flow ser extremamente flexível em relação às linguagens que pode ser aplicado, estando disponível de alguma forma em mais de 15 linguagens de programação variando das mais utilizadas no mercado até algumas mais nicho. Esta maior acessibilidade resulta em mais facilidade para encontrar mão de obra capacitada e maior menos tempo de treinamento de nova mão de obra.

Quando uma tecnologia começa a ficar mais conhecida e ser mais utilizada, é natural que ocorra um efeito bola de neve no qual mais pessoas são atraídas para esta tecnologia. Afinal, se vários projetos estão usando uma nova tecnologia, então ela deve oferecer vantagens em relação às outras. Hoje em dia o Tensor Flow está entre as ferramentas de *machine learning* mais populares e utilizadas do mercado e isso é refletido nos fóruns de ajuda de programação, qualquer dúvida relacionada à tecnologia é respondida rapidamente devido à grande popularidade e o alto número de usuários com conhecimento na ferramenta.

Caso o usuário deseje fazer algo que não é possibilitado pela API padrão da tecnologia, são facilmente encontradas diversas extensões à mesma, cada uma focando de forma mais profunda em algum ramo de *Machine learning*. Algumas destas foram criadas pela própria Google Brains, possuindo uma documentação tão robusta quanto a ferramenta original, e outras foram criadas por usuários que identificaram lacunas na usabilidade da ferramenta e decidiram preenchê-la.

VI. Projetos nos quais o Tensor Flow pode ser aplicado.

Com o crescimento da popularidade de *Machine learning* nos currículos de programadores pelo mundo é cada vez mais fácil encontrar projetos ostentando o uso de Tensor Flow. Uma simples pesquisa traz dezenas de milhares de projetos open source, tutoriais e dicas de como fazê-lo. Qualquer projeto que faça uso de *Machine learning* em alguma parte pode fazer uso da tecnologia Tensor Flow, então a gama de projetos é limitada somente à imaginação do usuário. Alguns exemplos de projetos:

- Reconhecimento de imagens: O exemplo mais clássico de projeto de *Machine learning*, se constitui de um programa que aprende a identificar elementos em imagens e quando as recebe, indica se o elemento que procura está presente nela ou não.

- Prever resultados: Projetos deste tipo são responsáveis por prever resultados e acontecimentos com base nos dados anteriores referentes ao tópico escolhido. Por exemplo, um programa que prevê o preço de mercado de uma casa com base em suas características (tamanho, número de quartos, localidade, etc).

- Resolvedor de Sudoku: Focado em ensinar um programa a resolver quaisquer grades de sudoku que forem apresentadas para o mesmo.

- Reconhecimento de voz: Se baseia em conseguir identificar palavras e associá-las aos significados corretos. Versões mais complexas permitem ao programa responder o usuário.