Área Científica de Matemática

Redes Neuronais e

Reconhecimento de voz

Autores:	44868	Diogo Leandro
	44805	José Pedro Jesus
	44827	Tiago Brito Matias

Relatório para a Unidade Curricular de Modelação Bayesiana

Professor: Gonçalo Morais

17-01-2020

Índice

Introdução

No âmbito da unidade curricular de Modelação Bayesiana foi-nos proposto fazer um trabalho acerca de um tema de inteligência artificial à nossa escolha. Foi escolhido pelo nosso grupo o tema do reconhecimento de voz e das redes neuronais associadas ao mesmo.

Este tema é atual visto que muitas ferramentas que utilizamos no nosso dia-a-dia possuem funcionalidades com reconhecimento de voz.

Com isto, o objetivo é desenvolver métodos e tecnologias que permitam o reconhecimento e a transcrição de linguagem falada de maneira automática. Para tal, são usadas redes neuronais, modelos computacionais baseados no sistema nervoso central de um animal (em particular o cérebro) que são capazes de realizar a aprendizagem da máquina bem como o reconhecimento de padrões. Estas são geralmente apresentadas como sistemas de "neurônios interconectados, que podem computar valores de entradas", simulando o comportamento de redes neuronais biológicas.

Este trabalho acabou por abordar determinadas redes neuronais e a sua aplicação ao problema de reconhecimento de voz, sendo essas redes as redes de convolução e as redes *feed-forward*.

Discutiremos também os resultados dessas redes na realização de uma solução para o nosso problema utilizando a nossa implementação dessas redes em *Python*.

Exemplos de reconhecimento de voz

-*Cortana*, o assistente do *Windows*

-*Siri*, o assistente pessoal da *Apple*

-*Google Assistant*, pertencente ao sistema operativo Android e a produtos da marca *Google Home -Alexa,*

o assistente da *Amazon*

Exemplos de Redes Neuronais

Multilayer Perceptron (MLP)

Convolutional Neural Network (CNN)

Recurrent Neural Network (RNN)

Long-Short Term Memory Network (LSTM)

Redes Feedforward: Introdução

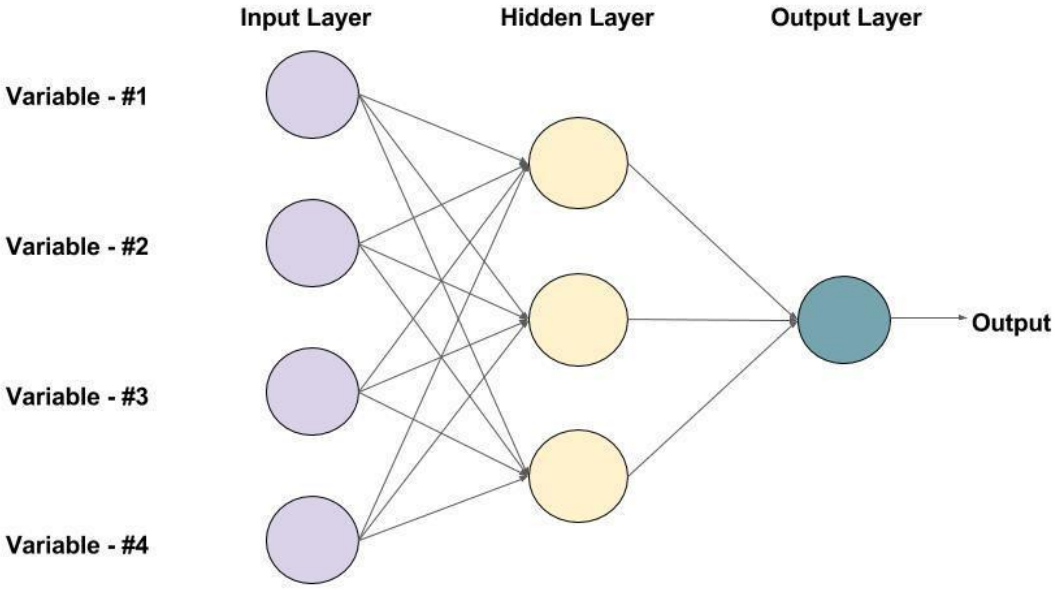
Nas redes *Feedforward* as camadas de rede são independentes umas das outras, assim, uma camada pode ter um número arbitrário de nós (sendo que, tipicamente, o número de nós arbitrários tem de ser superior ao de nós de input). Quando aplicadas a funções de aproximação, geralmente existe um input e um output. Por contraste, quando usadas como classificadores, o número de nós de input e output irá variar consoante o input apresentado. Adicionalmente, terá de ter no mínimo uma camada oculta e todos os nós das camadas têm peso 1.

Funcionamento das Redes Feedforward

O principal objetivo das redes Feedforward é fazer uma aproximação de uma dada função f.

Por exemplo, a função y = f(x) atribui a um dado valor de x um valor de y. Uma rede Feedforward define o mapeamento através de uma função y = f(x, θ), aprendendo os valores do parâmetro θ que resultam na melhor função de aproximação.

Estas redes são representadas por uma série de diferentes funções. Cada rede é também acompanhada por um grafo acíclico dirigido:



An example of a Feed-forward Neural Network with one hidden layer (with 3 neurons)

Representação de uma rede Feedforward através de um grafo acíclico dirigido:

https://www.learnopencv.com/tag/multilayer-perceptron/

Por exemplo, podemos ter três funções, f1, f2 e f3 interligadas formando então f (x) = f3(f2(f1(x))). Substituindo f2(f2(x)) por *a*, ficamos com f3(a). Neste caso, f1 é primeira camada de input, f2 a segunda e f3 a camada de output.

As camadas entre as camadas de input e output são conhecidas como "camadas ocultas", sendo que o *traning data* não expressa o output desejado a estas camadas. A rede pode conter infinitas camadas de camadas ocultas com um qualquer número de unidades. Uma unidade representa um neurónio que obtém o input de unidades de camadas prévias e calcula o seu valor de ativação.

Porque precisamos de redes Feedforward?

Quando uma determinada amostra está impossibilitada de ser separada linearmente, os modelos lineares apresentam alguma dificuldade a fazer aproximações, contrastando com a facilidade apresentada nas redes *Feedforward*. As camadas ocultas são, portanto, usadas para aumentar a "não linearidade" e alterar a representação de uma dada amostra para uma melhor generalização de uma função

CNN (Convulutional Neural Netowrks)

Na área de *deep learning,* redes de convolução (CNN, *Convolutional Neural Netowrks*) são uma classe de redes neuronais tipicamente utilizadas para a análise de imagens. São também conhecidas como Shift *Invariant* ou *Space Invariant Artificial Neural Networks* (SIANN). Este tipo de redes é tipicamente utilizado para o reconhecimento de imagem e vídeo, análises médicas e processamento de voz.

Redes de convolução: Introdução

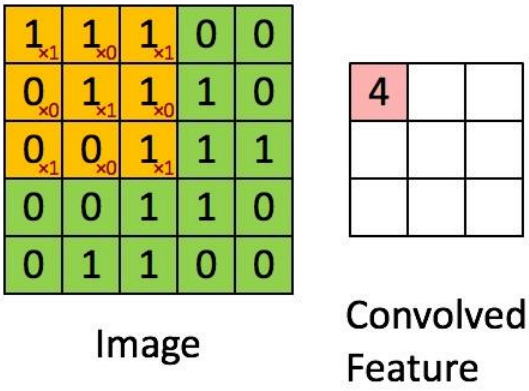
Na área de redes neuronais, CNN (redes de convolução) são uma das principais categorias para análise e classificação de imagens, como tal o uso deste tipo de rede tornou-se reincidente quando o objetivo é detetar objetos, reconhecer indivíduos ou até ler textos manuscritos.

Na nossa abordagem ao reconhecimento de voz, iremos dissecar a utilidade desta rede quando o assunto de estudo se trata não de uma simples imagem de duas dimensões (altura * largura), mas sim de reconhecer uma voz, algo dependente de um grande número de variáveis.

Convolução

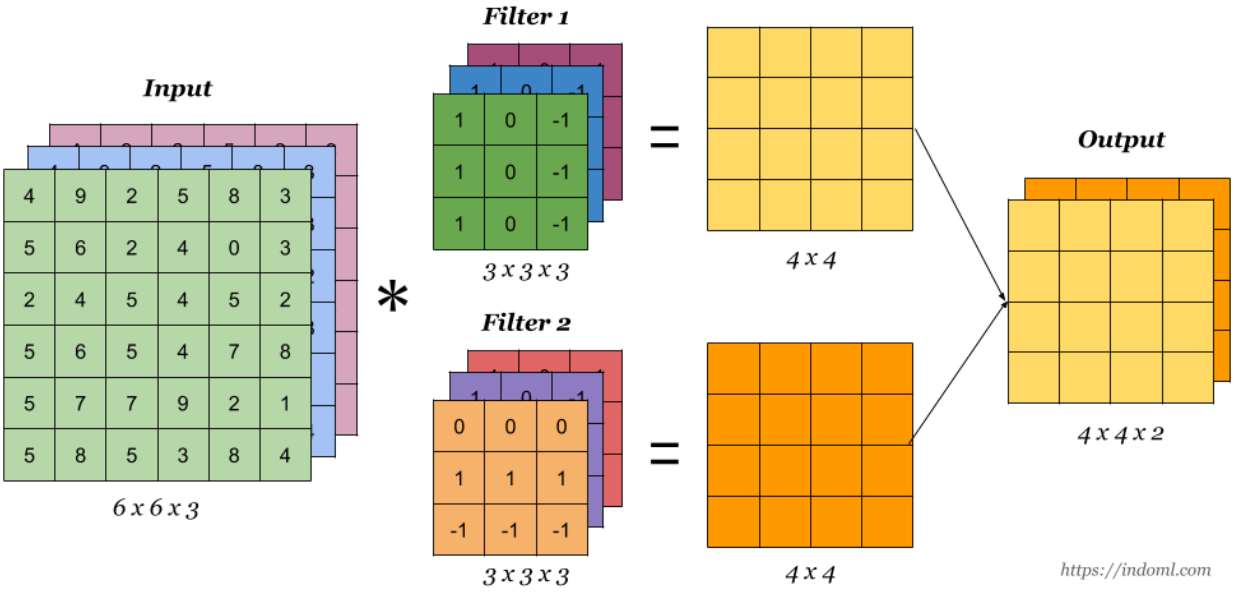
Uma convolução é a primeira camada de acesso a dados, que extrai com base num input uma amostra. Com essa amostra são depois extrapolados detalhes acerca do input aplicando um filtro/kernel.

Um kernel é uma matriz menor que a matriz de input, também denominada de matriz de convolução, cuja função é iterar a matriz de input aplicando-lhe um produto.



https://towardsdatascience.com/beginners-guide-to-understanding-convolutional-neural-networks-ae9ed58bb17d

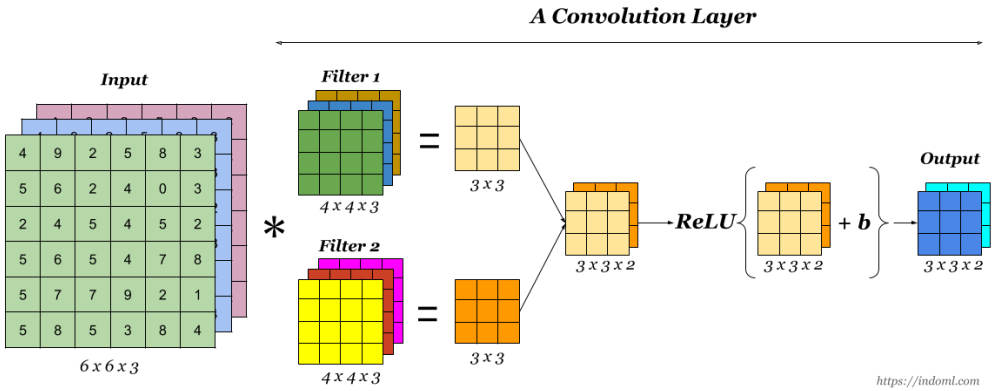
Quando o objetivo é extrair do output múltiplas propriedades usamos vários *kernels* do mesmo tamanho, de modo a que seja possível empilhar os vários resultados.



Processo de convolução com mais de um kernel:

https://towardsdatascience.com/beginners-guide-to-understanding-convolutional-neural-networks-ae9ed58bb17d

Numa última fase da convolução é aplicada à matriz filtrada uma função de ativação (normalmente sendo esta uma *Rectified Linear Unit (ReLu)* ou de *Tanh*) com o intuito de deixar de ter um output linear.



Processo completo de convolução:

https://towardsdatascience.com/beginners-guide-to-understanding-convolutional-neural-networks-ae9ed58bb17d

Redes de convolução versus Feedforward

No estudo de reconhecimento de voz e principalmente no estudo de reconhecimento de imagem podemos chegar à conclusão que as redes de convolução são muito mais poderosas na classificação de dados do que uma rede *Feedforward*, isto acontece com base nas extrações de amostras de um determinado input, ou seja, quando temos um elevado número de inputs as redes de convolução tendem a ser muito eficientes pois reduzem este número de inputs em grande escala.

No caso particular do reconhecimento de voz, frequentemente classificamos os dados em espectrogramas, sendo que é com esta representação de um troço de voz na forma de uma imagem que podemos classificar e descobrir os seus padrões, o que nos leva a concluir que redes de convolução são uma melhor escolha devido ao número de inputs.

Back-propagation

Back-propagation (conhecido como *backprop*) é um algoritmo vastamente utilizado no treino de redes Feedforward na área de *Machine Learning*.

A técnica de *backprop* calcula eficazmente o gradiente da função de erro com os respetivos pesos da rede para cada par de input e output. Desta forma, torna-se possível utilizar o gradiente para o treino de redes de várias camadas, alterando os pesos para minimizar a erro.

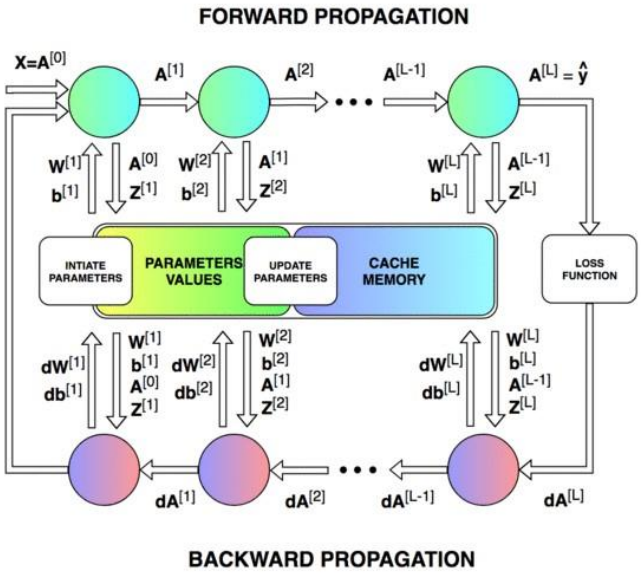
Normalmente, são usadas a descida do gradiente ou o SGD (*Stochastic Gradient Descent*). A técnica em questão acaba por ser a essência do treino de redes neuronais.

Back-propagation: Funcionamento

O algoritmo de *Back-propagation* calcula a descida do gradiente da função de erro com o

respetivo peso através da regra da cadeia, iterando para trás, camada a camada, da última cadeia, evitando cálculos redundantes.

Através da função de erro da iteração anterior, os pesos da rede neuronal são ajustados. O ajuste dos pesos visa a garantir a menor margem de erro possível tornando assim o modelo o mais estável possível.



Funcionamento de *Forward-Propagation* e de *Back-Propagation*: https://medium.com/datathings/neural-networks-and-backpropagation-explained-in-a-simple-way-f540a3611f5e

Reconhecimento de Imagem vs. Reconhecimento de Voz

Por convenção tem-se vindo a estabelecer uma regra geral no que toca a redes neuronais, *RNN's* são ótimas para tarefas de dados sequenciais tais como reconhecimento de voz, enquanto que *CNN's* estão maioritariamente redirecionadas para o uso em tarefas ligadas a vídeo e som, no entanto recente abordagem a modelos de informação sequencial feitas pelo Facebook tem vindo a mostrar excelentes resultados apenas com CNN's.

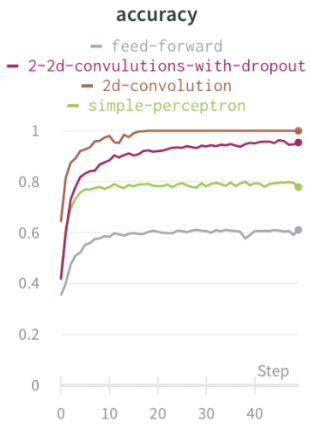
Resultados experimentais

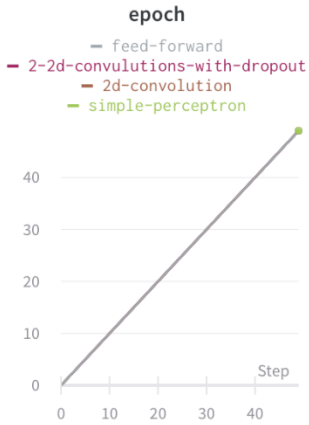
De modo a compreender melhor qual seria a melhor rede neuronal para resolver o problema de reconhecimento de voz decidimos fazer um pequeno projeto em *Python* utilizando um simples perceptrão, uma rede *Feed-Forward*, uma rede de convolução com duas convoluções e com apenas uma convolução, e também uma rede recorrente mais propriamente o algoritmo LSTM(Long Short-Term Memory loss), sendo esta última talvez a mais utilizada para o reconhecimento de voz.

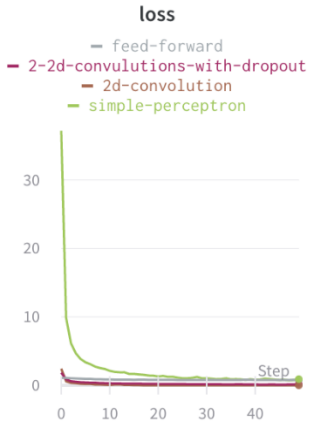
O código experimental está disponível no nosso repositório de *Github* em: https://github.com/Tdm9/Bayesian-Modulation

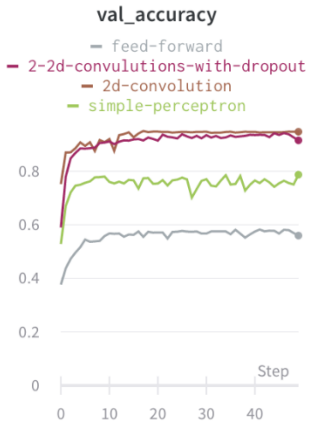
Conclusões

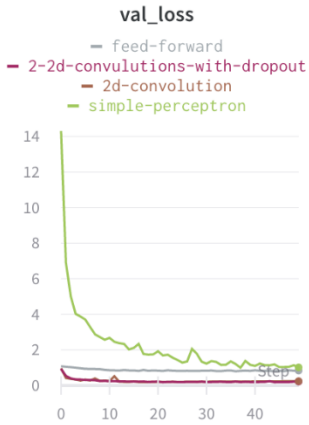
Utilizando o nosso Código em Python presente no nosso github apresentado acima e utilizando também a aplicação wandb (https://www.wandb.com/) para podermos visualizar os resultados obtidos, podemos confirmar determinadas suspeitas que já tínhamos.











No gráfico podemos verificar que utilizamos quatro redes neuronais para esta experiência, sendo estas, uma rede feed forward, um perceptrão, uma rede de convolução com uma camada e outra rede de convolução, mas desta vez com duas camadas.

Através dos gráficos conseguimos perceber que as duas redes de convolução obtêm os melhores resultados, tal como previsto. De qualquer das formas isto poderá dever-se a variados fatores como o tamanho das amostras e a forma como estão implementados os modelos.

Bibliografia

-https://nordicapis.com/5-best-speech-to-text-apis/

-https://machinelearningmastery.com/recurrent-neural-network-algorithms-for-deep-learning/

-https://www.quora.com/Which-neural-network-type-is-best-for-speech-recognition-and-speechsynthesis

-https://www.coursera.org/lecture/nlp-sequence-models/different-types-of-rnns-BO8PS

-https://medium.com/towards-artificial-intelligence/introduction-to-the-architecture-of-recurrentneural-networks-rnns-a277007984b7

-https://medium.com/@datamonsters/artificial-neural-networks-for-natural-language-processing-part1-64ca9ebfa3b2

-https://towardsdatascience.com/beginners-guide-to-understanding-convolutional-neural-networksae9ed58bb17d

-https://towardsdatascience.com/feed-forward-neural-networks-c503faa46620

-https://dzone.com/articles/the-very-basic-introduction-to-feed-forward-neural

-https://towardsdatascience.com/how-does-back-propagation-in-artificial-neural-networks-workc7cad873ea7

-https://medium.com/datathings/neural-networks-and-backpropagation-explained-in-a-simple-wayf540a3611f5e

-https://en.wikipedia.org/wiki/Backpropagation

-https://towardsdatascience.com/what-the-hell-is-perceptron-626217814f53 -

https://github.com/silversparro/wav2letter.pytorch