

Data de Entrega: 03 de Dezembro de 2018

1 Introdução

O objetivo deste trabalho é colocar em prática conceitos relacionados a redes neurais. Iremos implementar uma rede capaz de resolver um problema de classificação supervisionada, ou seja, seu modelo neural artificial deverá avaliar uma instância dos dados e prever qual é a classe do objeto descrito por seus atributos (features). O foco está na parametrização da rede e não na implementação dos algoritmos em si, e por isso o uso de uma biblioteca que implemente redes neurais é primordial.

Diversas bibliotecas oferecem suporte para implementação de redes neurais. Recomendamos duas delas: Keras com TensorFlow¹ e PyTorch². Caso você queira utilizar uma outra biblioteca, nos consulte. Ferramentas como o Weka ou Orange não devem ser utilizadas.

2 Base de Dados

A base de dados fornecida é uma amostra de dados do Sloan Digital Sky Survey. Ela possui 5000 instâncias cada uma com 18 colunas: 17 features e 1 classe (*class*) a ser predita. Cada instância pode representar uma galáxia (GALAXY), uma estrela (STAR) ou um quasar (QSO).

3 Desenvolvimento

1. O primeiro passo após baixar a base de dados é verificar se serão necessárias transformações nos dados de entrada.
2. Escolha o tipo da rede neural que você irá utilizar (Perceptron, RBF, etc).
3. Escolhida a rede, deve-se pensar em sua arquitetura. É importante definir o que os nós de entrada e os nós de saída representarão, e quantos neurônios serão utilizados nessas duas camadas. Se a rede tiver uma camada escondida, temos também que escolher seu número de neurônios. A definição da função de ativação também é importante nessa fase.
4. Definida a arquitetura da rede, deve-se inicializar os pesos da rede e definir o valor inicial da taxa de aprendizagem e o número de épocas ou outro critério de parada.

¹<https://keras.io/>

²<http://pytorch.org/>

5. A função de custo utilizada pelo algoritmo de aprendizado (e.g., *back-propagation*) também é um parâmetro importante da rede e afeta diretamente a velocidade de aprendizagem. Escolha com cuidado.
6. O treinamento pode ser feito utilizando um exemplo por vez (stochastic gradient descent) ou uma abordagem de mini-batches. Nessa segunda abordagem, os pesos da rede são atualizados depois que uma quantidade n de exemplos é visto pela rede, onde n é o tamanho do mini-batch (n normalmente assume valores que variam de 2 a 100).
7. Os dados utilizados devem ser divididos em duas ou três partes, dependendo do critério de parada selecionado. Quando os dados são divididos em duas partes, a primeira (que normalmente corresponde a 70% do total) é utilizada para treinar a rede por um número máximo de épocas. Ao fim do treinamento, a segunda parte é utilizada para medir a capacidade de generalização da rede.

Já quando os dados são divididos em 3 partes, a primeira parte é novamente utilizada para treinar a rede. A segunda é utilizada para validar a rede, isto é, para medir seu erro. Nesse caso, o treinamento pode parar quando a rede atingir um erro mínimo nesse segundo conjunto de dados, chamado conjunto de validação. A terceira parte dos dados é utilizada apenas ao fim do treinamento, para medir a capacidade de generalização da rede.

Independente da abordagem utilizada, o ideal é que um procedimento de validação cruzada seja utilizado para garantir que os resultados de generalização não estão sendo obtidos ao acaso. Uma validação cruzada de 3 partições é suficiente para esse problema.

4 Guia de Experimentação

1. O que acontece quando se aumenta o número de neurônios da camada escondida da rede? Isso afeta o número de épocas necessárias para convergência?
2. O que acontece quando se aumenta o número de camadas escondidas? O ganho no erro é grande o suficiente para justificar a adição de uma nova camada?
3. Qual o impacto da variação da taxa de aprendizagem na convergência da rede? O que acontece se esse parâmetro for ajustado automaticamente ao longo das diferentes épocas?
4. Compare o treinamento da rede com gradient descent estocástico com o mini-batch. A diferença em erro de treinamento versus tempo computacional indica que qual deles deve ser utilizado?
5. Qual a diferença do erro encontrado pela rede no conjunto de treinamento ou validação em relação ao erro encontrado no teste? Existe overfitting? Como ele pode ser evitado?
6. A base com que você trabalhou é um pouco desbalanceada. Você pode tentar contornar esse problema usando a técnica de oversampling, ou seja, fazendo cópias dos exemplos das classes minoritárias para balancear melhor a base. Por exemplo, a classe QSO tem 412 exemplos, e a classe GALAXY 2501. De forma simples, você poderia fazer 6 cópias de cada exemplo da classe QSO, aumentando o número de exemplos dessa classe para

2472, e utilizando todos eles no treinamento da rede. Fazendo um oversampling das classes minoritárias e retreinando a rede com os melhores parâmetros encontrados, o erro diminuiu? Por quê?

5 Entrega

Deve ser entregue um arquivo compactado contendo:

- Código;
- Relatório:
 - O relatório deve ser dividido em no mínimo 4 tópicos: Introdução, Modelagem e Implementação, Análise Experimental e Conclusão. Ele deve apresentar obrigatoriamente a arquitetura da rede utilizada e gráficos que mostrem como diferentes valores dos parâmetros listados impactam na taxa de erro da rede, assim como uma possível explicação para os resultados encontrados. Além disso, você deve concluir o relatório listando os parâmetros que obtiveram os melhores resultados ao fim da análise experimental.