Univesp – Universidade Virtual do Estado de São Paulo
COM510 – Redes Neurais
Prof. Marcos G. Quiles

Questões de Revisão

1. **O que são as redes neurais artificiais?**
   R: São modelos computacionais (redes) inspirados na estrutura e no funcionamento do cérebro (sistema nervoso). Basicamente, as redes neurais artificias são redes formadas por neurônios matemáticos interconectados. Esses neurônios são modelos matemáticos que representam uma aproximação qualitativa do neurônio biológico. Há diversos tipos de redes neurais a depender do tipo de neurônio, conexões, fluxo de informação etc. Esse tema foi abordado na Videoaula 1 e no material correspondente.

2. **O que são os neurônios artificiais (matemáticos)?**
   R: São modelos matemáticos que representam, de forma qualitativa, o neurônio biológico. Há diversos tipos de neurônios, desde o simples MCP, que representa uma célula binária, até células dotadas de memórias de curto e longo prazo, como as células LSTM. É importante destacar que há neurônios matemáticos que descrevem a célula biológica de forma quantitativa, como o modelo de Hodgkin-Huxley, porém esse tipo de neurônio não é comumente utilizado na composição de redes neurais artificiais, mas sim em neurociência computacional. Este tema foi abordado na Videoaula 2 e no material correspondente.

3. **Qual foi o problema do Perceptron apontado por Minksy e Papert em seu livro publicado em 1969?**
   R: Minsky e Papert afirmaram que a rede Perceptron só era capaz de resolver problemas simples (lineares) e, mesmo com a adição de mais camadas, esse modelo não seria capaz de resolver problemas mais complexos com mapeamentos não lineares, como o XOR (ou-exclusivo). Esse tema foi abordado na Videoaula 1 e no material correspondente.

4. **Quais são os principais fatos históricos responsáveis pela retomada do interesse nas redes neurais na década de 1980?**
   R: O modelo de Hopfield, proposto em 1982, e o algoritmo de retropropagação do erro, publicado em 1986. Esse tema foi abordado na Videoaula 1 e no material correspondente.

5. **O que possibilitou o avanço da área de redes neurais na direção dos modelos profundos?**
   R: Além de alguns avanços científicos, como o desenvolvimento de novos tipos de células e funções de ativação menos susceptível ao problema do desaparecimento ou explosão do gradiente, como a ReLU, os dois principais pilares que deram suporte

ao desenvolvimento da área de aprendizado profundo foram: a era do Big Data, ou seja, coleta e armazenamento de grandes volumes de dados; e o desenvolvimento das placas aceleradoras gráficas (GPU). Esse tópico foi abordado na Videoaula 1.

6. **O que é o neurônio MCP? Quais são suas partes fundamentais?**
   R: O neurônio MCP é uma célula binária composta por três partes fundamentais: as sinapses, associadas às entradas – dendritos; a soma ou corpo celular; e uma saída, representando o axônio da célula biológica. Essa célula recebe um padrão **x** de entrada. Na sequência, o corpo celular faz uma soma ponderada das entradas pelos seus respectivos pesos sinápticos gerando o campo local induzido, denominado v. Se v for maior que um dado limiar *theta*, a célula dispara (saída 1), caso contrário, a célula permanece em repouso (saída 0 ou -1). Esse tópico foi discutido na Videoaula 2 e material correspondente.

7. **O que são as funções de ativação? Cite alguns exemplos.**
   R: São funções utilizadas na saída dos neurônios matemáticos. Têm por finalidade transformar o sinal gerado pelo corpo celular e produzir a saída do neurônio. Há diferentes tipos de funções de ativação, desde as simples funções lineares f(x)=x, passando pela função degrau (utilizada no neurônio MCP), até as funções contínuas, como a sigmoide logística, tangente hiperbólica e ReLU. Esse tema foi abordado na Videoaula 2 e no material correspondente.

8. **Qual é a diferença entre um neurônio determinístico de um estocástico?**
   R: A saída de um neurônio determinístico, como o MCP, é gerada de forma determinística a partir do valor do campo local induzido da célula e o seu limiar de disparo, ou seja, sempre que o campo local induzido (v) for superior a *theta* (limar de disparo), a célula dispara, caso contrário, ela permanece em repouso. No caso da célula estocástica, há uma probabilidade associada ao estado da célula, ou seja, a célula terá probabilidade p(v) de disparar e 1-p(v) de permanecer em repouso, no qual v é o campo local induzido do neurônio. É comum associar a função de probabilidade ao conceito de temperatura, responsável por controlar a estocasticidade da célula. Quando a temperatura tende a zero (limite), a célula se comporta de forma determinística, se a temperatura for para infinito, a célula passa a se comportar de forma totalmente aleatória: 50%/50% para disparar ou permanecer em repouso. Esse tema foi discutido na Videoaula 2 e no material correspondente.

9. **O que é aprendizagem em redes neurais?**
   R: Também denominado treinamento, é o processo pelo qual os parâmetros livres da rede neural são ajustados. Basicamente, o processo de aprendizagem consiste em três etapas:
      1) A rede recebe um estímulo (apresentação de um exemplo na entrada da rede).
      2) A rede sofre modificações em função do estímulo e de sua resposta. Essas modificações, definidas por Delta w, depende diretamente da técnica de aprendizagem utilizada.

3) Por fim, a rede passa a responder de forma diferente aos estímulos do ambiente.

Se o processo de aprendizagem estiver correto, a rede deve responder melhor no tempo t+1 do que respondia no tempo t. Esse tópico foi abordado na Videoaula 3 e no material correspondente.

10. O que é a técnica de aprendizagem denominada: aprendizagem por correção de erros?

R: Essa técnica é base para a atualização dos pesos do erro calculado na saída do modelo. Ou seja, calcula-se um erro e = d-y, no qual d é o valor desejado e y é a saída fornecida pelo modelo para um dado estímulo x. O valor de atualização dos pesos é calculado a partir desse erro, ou seja, Delta_w = f(e). Uma técnica pertencente à aprendizagem por correção de erros é a regra delta, que utiliza o gradiente descendente para ajustar os pesos do modelo. Essa técnica foi discutida na Videoaula 3 e no material correspondente.

11. O que é o aprendizado competitivo?

R: No aprendizado competitivo, as células da rede (ou de uma dada camada) competem para ganhar o direito de responder a um determinado estímulo. A célula vencedora terá seus parâmetros atualizados fazendo com que o neurônio vencedor responda de forma ainda melhor na próxima vez que esse mesmo padrão for apresentado à rede. Ou seja, a atualização consiste em aproximar os pesos do neurônio vencedor aos respectivos valores do padrão x apresentado à rede. Esse tópico foi abordado na Videoaula 3 e no material correspondente.

12. Explique os paradigmas de aprendizagem: supervisionado, não supervisionado e por reforço, ilustre algumas aplicações de cada paradigma.

R: Cada um desses paradigmas tem seus próprios objetivos, métodos e aplicações.

Aprendizado supervisionado: é um paradigma de aprendizagem em que um modelo é treinado em um conjunto de dados rotulados. O objetivo do modelo é aprender a mapear os recursos (*inputs*) aos rótulos (*outputs*) corretos. O conjunto de dados rotulados é composto por pares de entrada e saída correspondentes, que são usados para ensinar o modelo a realizar uma tarefa específica. Alguns exemplos de aplicações do aprendizado supervisionado incluem: reconhecimento de fala, classificação de imagens e previsão de preços de ações.

Aprendizado não supervisionado: neste paradigma, o modelo é alimentado com dados não rotulados e tem como função principal descobrir padrões ou estruturas dentro desses dados. Alguns exemplos de aplicações de aprendizagem não supervisionada são: agrupamento de dados, detecção de anomalias, redução de dimensionalidade e recomendação de produtos.

Aprendizado por reforço: neste paradigma, o modelo aprende a tomar decisões em um ambiente através de tentativa e erro. O modelo é recompensado ou punido com base nas decisões que toma e, ao longo do tempo, aprende a maximizar sua recompensa. Ao contrário dos paradigmas supervisionado e não supervisionado, que

aprendem por meio de exemplos, o aprendizado por reforço aprende a partir da experiência com o ambiente. Alguns exemplos de aplicações de aprendizado por reforço são: jogos de tabuleiro, jogos de videogame, controle de robôs e tomada de decisões em finanças. Esses paradigmas foram discutidos na Videoaula 3 e no material correspondente.

13. O que é o aprendizado autossupervisionado?

R: O aprendizado autossupervisionado pertence ao paradigma não supervisionado, uma vez que não demanda exemplos rotulados para ser considerado. Especificamente, modelos pertencentes a esse paradigma apresentam como saída desejada o próprio padrão de entrada (por isso autossupervisionado). Um exemplo típico de rede neural que utiliza esse paradigma são os autoencoders. Esse tema foi discutido, de forma sucinta, na Videoaula 3.

14. O que é e quais são as vantagens da representação de redes neurais por grafos direcionados?

R: Os grafos de fluxo fornecem uma descrição funcional dos vários elementos que constituem o modelo de um neurônio artificial, podendo ser utilizado para uma descrição inter ou intra neurônio. Dentre as possíveis vantagens dessa representação de neurônios e redes neurais, podemos citar: simplificação da representação e facilita a derivação através da estrutura do grafo. Esse tipo de representação foi abordado na Videoaula 4 e no material correspondente.

15. Diferencie uma rede com camada única de uma rede com múltiplas camadas.

R: Uma rede com única camada possui apenas uma camada de neurônios ajustáveis. Já uma rede com múltiplas camadas possui duas ou mais camadas de neurônios ajustáveis. Em sua forma padrão, denominada redes densas, todos os neurônios de uma dada camada são ligados a todos os neurônios da camada posterior, da entrada até a saída da rede, camada a camada.

As redes com múltiplas camadas, ao contrário das redes com camada única, podem ser treinadas para resolver problemas não linearmente separáveis.

16. O que caracteriza uma rede recorrente?

R: Uma rede recorrente possui laços de realimentação. Ou seja, o estado de um dado neurônio depende de suas entradas e do estado da própria célula e de outras células da rede. Há diversos tipos de redes recorrentes, como as redes de Hopfield e as redes LSTM. A arquitetura recorrente foi introduzida na Videoaula 4, os modelos recorrentes específicos foram estudados nas Semanas 6 e 7 do curso.

17. O que são arquiteturas profundas?

R: São redes neurais que possuem várias camadas. Em sua definição básica, dizemos que uma rede é profunda se possui duas ou mais camadas ocultas. Esse tema foi mencionado no final da Videoaula 4.

18. Diferencie hiperparâmetros de parâmetros de um modelo.

R: Os hiperparâmetros estão associados à estrutura macro do modelo, como o número de camadas e o número de neurônios em cada camada. Já os parâmetros

representam aspectos micro das células, por exemplo seus pesos sinápticos. Para desenvolvimento de um modelo, primeiro definimos os hiperparâmetros (estrutura da rede) e, na sequência, treinamos o modelo (ajuste dos parâmetros). Esse processo é repetido até encontramos uma rede neural apta para o problema em estudo. Esse tópico foi introduzido na Videoaula 5 e material correspondente.

19. Qual é o papel do conjunto de treino, validação e teste?

R: O conjunto de treino, validação e teste desempenham um papel crucial no aprendizado de máquina, pois eles são usados para avaliar a capacidade do modelo de generalização, ou seja, sua capacidade de realizar previsões precisas em novos dados.

O conjunto de treino é usado para treinar o modelo, ou seja, ajustar os pesos e parâmetros do modelo para minimizar a função de perda. O conjunto de treino é composto por dados rotulados que são usados para ensinar o modelo a mapear os dados de entrada para as saídas corretas.

O conjunto de validação é usado para ajustar os hiperparâmetros do modelo, como o tamanho do lote, a taxa de aprendizado, o número de camadas e o número de neurônios em cada camada. Esses hiperparâmetros afetam a capacidade do modelo de aprender e a rapidez com que ele aprende. O conjunto de validação é usado para avaliar o desempenho do modelo em dados que não foram usados para treiná-lo.

O conjunto de teste é usado para aferir a performance final do modelo treinado. O conjunto de teste é usado apenas no final do processo de treinamento, após o modelo ter sido ajustado e validado. Esse tópico foi abordado na Videoaula 5 e no material correspondente.

20. O que é o underfitting?

R: É definido como o subajuste do modelo aos dados, ou seja, o modelo não é capaz de aprender de forma efetiva os exemplos de treinamento. Geralmente ocorre pela falta de flexibilidade do modelo desenvolvido (alto bias).

21. O que é o overfitting?

R: Consiste no sobreajuste do modelo ao conjunto de treino. Geralmente é acarretado pelo excesso de flexibilidade (parâmetros livres). O overfitting é observado quando o modelo atinge um baixo erro no conjunto de treino (tendendo a zero) e alto erro no conjunto de validação, indicando que o modelo decorou os dados de treinamento e perdeu sua capacidade de generalização. Esse tópico foi abordado na Videoaula 5 e material correspondente.

22. O que é a validação cruzada e qual é sua principal vantagem em relação a divisão tradicional do conjunto de desenvolvimento em treino/validação?

R: A validação cruzada (*cross-validation*) é uma técnica usada para avaliar o desempenho do modelo e estimar sua capacidade de generalização.

Na validação cruzada, o conjunto de dados é dividido em k partes iguais, k é um número escolhido previamente. Cada parte é usada como conjunto de validação, enquanto as k-1 partes restantes são usadas como conjunto de treinamento. O modelo é treinado k vezes, cada vez usando uma parte diferente como conjunto de validação e as partes restantes como conjunto de treinamento. Os resultados das k execuções são então combinados para fornecer uma estimativa mais precisa do desempenho do modelo.

A principal vantagem da validação cruzada em relação à divisão tradicional do conjunto de desenvolvimento em treino/validação é que ela usa todos os dados para treinar e validar o modelo, reduzindo, assim, a variância do modelo. Na divisão tradicional, a divisão dos dados pode afetar significativamente o desempenho do modelo e pode levar a uma avaliação otimista ou pessimista do modelo, dependendo da divisão. Com a validação cruzada, cada ponto de dados é usado tanto no conjunto de treinamento quanto no conjunto de validação, garantindo que o modelo seja avaliado de forma justa em todo o conjunto de dados.
A validação cruzada foi introduzida na Videoaula 5 e no material associado.

23. Cite alguns problemas comumente observados em dados reais.
R: Presença de ruídos, dados faltantes ou incompletos, dados desbalanceados, dados duplicados, escalas incompatíveis, dentre outros. Tópico abordado na Videoaula 5.

24. Qual é a função do pré-processamento dos dados?
R: O pré-processamento tem por objetivo melhorar a qualidade dos dados; facilitar a utilização de uma determinada técnicas de aprendizado de máquina; e acelerar o treinamento do modelo.

25. Como podemos transformar dados categóricos em numéricos?
a. Nominal (1 de c) e ordinal (escalar).

R: Os atributos categóricos podem ser transformados em atributos numéricos utilizado duas abordagens principais, a depender do tipo de dado. Por exemplo, para dados nominais, podemos usar a codificação 1-de-c, ou seja, cada uma das c categoria é representada por um valor binário com c bits. Por outro lado, para dados ordinais, podemos representá-los como um escalar, por exemplo, para [pequeno, médio, grande], podemos usar [0, 1, 2]. Essas transformações foram introduzidas na Videoaula 5.

26. Qual é a importância da normalização dos dados? Quais são as possíveis desvantagens?
R: A normalização tem como objetivo central equilibrar a amplitude de valores (escala) de atributos distintos, facilitando o treinamento do modelo. Por outro lado, ela pode reduzir a importância de um dado atributo em função dos demais. Portanto, deve ser usada com parcimônia. A normalização foi apresentada na Videoaula 5 e no material correspondente.

27. Quais são as diferenças em normalizar os dados por distribuição *versus* por amplitude?

R: A normalização por amplitude restringe os valores em um intervalo fixo (i.e., min-max), adequada quando precisamos estabelecer esses limites. A normalização por distribuição transforma os dados de tal forma que a média seja zero e o desvio um, ela possui maior tolerância a outliers. A normalização foi apresentada na Videoaula 5 e no material correspondente.

28. O que um neurônio MCP representa? Como configurá-lo para resolver um determinado problema?

R: Um neurônio MCP representa uma fronteira de separação linear, no caso 2D uma rede, 3D um plano, e para dimensões maiores um hiperplano. O MCP, em sua proposta padrão, não possui um algoritmo de aprendizagem, logo, os pesos devem ser configurados de forma manual, por exemplo resolvendo um sistema de inequações. O MCP e o seu ajuste para tratamento de portas lógicas foram introduzidos na Videoaula 6.

29. O que é o Perceptron?

R: O Perceptron é uma rede neural com apenas uma camada de neurônios ajustáveis. Basicamente, o Perceptron consiste em uma rede com neurônios do tipo MCP associada a um algoritmo de treinamento. O Perceptron pode ser utilizado para classificações de padrões que sejam linearmente separáveis.

30. Como é realizado o processo de treinamento do Adaline? Explique o método do gradiente.

R: O Adaline é treinado via regra delta, ou seja, via técnica do gradiente descendente. Primeiro, define-se uma função de custo para o problema, por exemplo o erro quadrático entre o sinal de saída desejado e o sinal produzido pela rede. Na sequência, calcula-se a derivada da função de custo em relação aos parâmetros do modelo. Esse valor, ponderado pela taxa de aprendizagem, é utilizado para atualizar os parâmetros da rede Adaline. O Adaline e o seu algoritmo de aprendizagem foram introduzidos na Videoaula 6.

31. O que diferencia o Perceptron do Adaline?

R: Ambos são redes capazes de tratar problemas lineares. A principal diferença está na saída dos neurônios, no caso do Perceptron, adota-se uma função de ativação do tipo degrau, ou seja, a saída do Perceptron é binária. Por outro lado, a saída do Adaline é o próprio campo local induzido, ou seja, a saída consiste no somatório ponderado das entradas. Esse tema foi introduzido na Videoaula 6 e no material correspondente.

32. Por que um Perceptron não pode implementar uma porta XOR?

R: O Perceptron pode construir apenas fronteiras de separação linear e o problema XOR demanda uma fronteira não linear, logo, o Perceptron não é capaz de implementar essa porta. Esse tema foi introduzido na Videoaula 6 e no material correspondente.

33. O que acontece se utilizarmos funções de ativação linear em redes MLP?

R: Uma rede com múltiplas camadas com adoção de funções de ativação lineares é equivalente a uma rede de camada única com saída linear. Ou seja, não há ganho de aprendizagem ao utilizarmos várias camadas lineares em uma rede MLP. Portanto, para que uma rede MLP seja capaz de resolver problemas não linearmente separáveis, precisamos utilizar funções de ativação não linear nos neurônios das camadas ocultas da rede. Esse tema foi introduzido na Videoaula 7 e no material correspondente.

34. Explique por que conseguimos resolver uma porta XOR com uma rede MLP com uma camada oculta e um neurônio de saída. Qual é o papel de cada neurônio nesse processo?

R: Redes com múltiplas camadas com funções de ativação não lineares são capazes de resolver problemas não lineares. Basicamente, cada camada da rede é responsável por transformar o padrão recebido em uma outra representação. Camada a camada, os padrões originais não linearmente separáveis são transformados em padrões linearmente separáveis na penúltima camada da rede, logo, a camada de saída será capaz de resolvê-lo. Cada neurônio, independente da sua posição na rede, é capaz de gerar uma fronteira de separação linear no espaço de dados de entrada. Esse tema foi introduzido na Videoaula 7 e no material correspondente.

35. O que é o algoritmo de retropropagação?

R: O algoritmo de retropropagação do erro consiste na própria regra delta generalizada. Ele é responsável pelo cálculo das derivadas associadas aos parâmetros pertencentes as camadas internas da rede, permitindo o ajuste dos pesos dos neurônios ocultos, para os quais não temos o erro de saída calculado de forma explícita. O algoritmo de retropropagação (*backpropagation*) foi introduzido na Videoaula 8.

36. Como calculamos o gradiente em um neurônio da camada oculta?

R: O gradiente de um neurônio oculto é calculado a partir da soma ponderada dos gradientes dos neurônios da camada posterior pelos seus respectivos pesos. Esse valor é multiplicado pela derivada da função de ativação do próprio neurônio para gerar o gradiente local. Essa derivação foi apresentada na Videoaula 8 e no material correspondente.

37. Como podemos configurar os hiperparâmetros de uma rede MLP via divisão do conjunto de desenvolvimento em treino e validação?

R: Há diversas formas de se utilizar os erros de treino e validação para ajuste dos hiperparâmetros de uma rede MLP. Abaixo apresento uma heurística para esse processo:

Passo 01 – Iniciando por uma rede simples (linear), amplie o número de camadas/neurônio até atingir uma rede neural com capacidade de aprendizagem suficiente para gerar um erro de treino baixo.

Passo 2 – Considerando a rede obtida no passo anterior, que é capaz de apresentar um erro de treino baixo, avaliamos a capacidade de generalização desta rede com o conjunto de validação. Se o erro de validação estiver baixo, a rede está pronta. Caso o erro de validação esteja elevado, cenário de overfitting, precisamos efetuar ajustes nos hiperparâmetros da rede ou inserir algum mecanismo de regularização, com o objetivo de restringir a capacidade de aprendizagem da rede.

O erro de validação também pode ser utilizado para estabelecer a parada prematura do treinamento. Essa parada pode ser realizada da seguinte forma, avalia-se, época a época, a evolução do erro de treino e o erro de validação. Quando o erro de validação começar a subir, interrompemos o treinamento, pois, a partir deste ponto, é possível que a rede se especialize no conjunto de treino gerando um overfitting dos dados. Esse tópico foi abordado na Videoaula 10 e no material associado.

38. Como podemos acelerar o treinamento de uma rede MLP?

R: Há diversas heurísticas para acelerar o treinamento de uma rede MLP, por exemplo: taxa de aprendizagem dinâmica, adição do termo de *momentum*, normalização do conjunto de dados, utilização de algoritmos otimizados, como RMSProp e Adam, dentre outras. Esse tópico foi abordado nas Videoaulas 10, 11 e nos materiais correspondentes.

39. O que é o termo de *momentum*?

R: O termo de *momentum* consiste numa média móvel do gradiente. Ao adotarmos o *momentum*, ao invés de utilizarmos o valor do gradiente diretamente na atualização dos parâmetros, quando adotamos a média móvel do gradiente. Esse processo traz mais estabilidade do treinamento e tende a acelerar o processo. O termo de *momentum* foi introduzido na Videoaula 10 e no material associado.

40. Compare o algoritmo Adam ao algoritmo de retropropagação original (SGD)

R: O algoritmo Adam pode ser visto como uma generalização do SGD. Em particular, ao invés de calcular a atualização (Delta_w) diretamente a partir do gradiente, utiliza-se o termo de *momentum* (média móvel). Além disso, a taxa de aprendizagem (eta) é normalizada pela média móvel dos gradientes em segunda ordem. Esse algoritmo, juntamente ao RMSProp, foi introduzido na Videoaula 11 e no material de referência associado.

41. Defina o que é o método padrão, o método em lote e o método em mini-lote (*mini-batch*)

R: O método padrão, o método em lote e o método em minilote são diferentes abordagens para otimizar um modelo de aprendizado de máquina usando gradiente descendente.

O método padrão, também conhecido como gradiente descendente estocástico (SGD), atualiza os pesos do modelo para cada ponto de dados de treinamento individualmente. Ele computa o gradiente da função de custo em relação a cada peso do modelo para um ponto de dados de treinamento e atualiza os pesos com uma pequena quantidade proporcional ao gradiente. O método padrão é

computacionalmente eficiente, mas pode ser instável, especialmente quando a superfície da função de custo é irregular ou ruidosa.

O método em lote (*batch gradient descent*) computa o gradiente da função de custo em relação a todos os pontos de dados de treinamento de uma vez. Em seguida, ele atualiza os pesos do modelo com uma pequena quantidade proporcional ao gradiente. O método em lote é mais estável do que o método padrão, pois leva em conta toda a informação dos dados de treinamento ao mesmo tempo, mas pode ser computacionalmente caro, especialmente para conjuntos de dados grandes.

O método em minilote (*mini-batch gradient descent*) é uma abordagem intermediária que computa o gradiente da função de custo em relação a um pequeno subconjunto (lote) dos dados de treinamento em vez de todos os dados. Em seguida, ele atualiza os pesos do modelo com uma pequena quantidade proporcional ao gradiente do lote. O tamanho do lote pode ser ajustado para equilibrar a eficiência computacional e a estabilidade do modelo. O método em minilote é a abordagem mais popular para otimização de modelos de aprendizado de máquina, pois é computacionalmente eficiente e estável. Tema abordado na Videoaula 10 e material de referência correspondente.

42. Por que a regularização auxilia no combate ao overfitting?

R: A regularização é uma técnica usada em modelos de aprendizado de máquina para combater o overfitting, que ocorre quando um modelo se ajusta muito bem aos dados de treinamento, mas não é capaz de generalizar bem para novos dados. A regularização adiciona uma penalidade aos pesos do modelo, forçando-os a serem menores e, assim, reduzindo a complexidade do modelo.

Existem várias formas de regularização, mas a mais comum é a regularização L2, que adiciona uma penalidade proporcional ao quadrado da norma L2 dos pesos do modelo à função de custo. Isso incentiva os pesos a serem pequenos, reduzindo, assim, a capacidade do modelo e impedindo-o de se ajustar muito bem aos dados de treinamento.

A regularização ajuda a combater o overfitting, pois impede que o modelo se ajuste demais aos dados de treinamento, forçando-o a se concentrar nas características mais importantes do problema. Ao adicionar a penalidade aos pesos do modelo, a regularização torna mais difícil para o modelo ajustar-se aos ruídos nos dados de treinamento e, em vez disso, enfatiza as características mais importantes. Esse tema foi abordado na Videoaula 11 e no material correspondente.

43. O que é a regularização L2?

R: A regularização L2 é uma técnica usada em modelos de aprendizado de máquina para reduzir o overfitting adicionando uma penalidade proporcional ao quadrado da norma L2 dos pesos do modelo à função de custo.

A adição da penalidade L2 à função de custo incentiva o modelo a ter pesos menores e, assim, reduzir sua complexidade. Isso pode ajudar a impedir o modelo de se

ajustar muito bem aos dados de treinamento e, em vez disso, concentrar-se nas características mais importantes do problema.

A equação da regularização L2 é dada por:

$E' = E + \lambda * ||w||^2$

E é a função de custo original, λ é o hiperparâmetro de regularização que controla a força da penalidade, e w é o vetor de pesos do modelo. A norma L2 $||w||^2$ é calculada como a soma dos quadrados dos pesos do modelo.

Ao minimizar a função de custo E', o modelo tenta encontrar os pesos que minimizam a função de custo original E, ao mesmo tempo em que mantém os pesos pequenos para reduzir sua complexidade. O hiperparâmetro λ controla o equilíbrio entre a precisão do modelo e sua complexidade. Quanto maior o valor de λ, mais forte é a penalidade e, assim, menor a complexidade do modelo. Esse tópico foi abordado na Videoaula 11.

44. O que é a regularização via Dropout?

R: A regularização dropout é uma técnica de regularização popular em modelos de redes neurais artificiais que ajuda a reduzir o overfitting. O dropout envolve aleatoriamente "desligar" neurônios da rede neural durante o treinamento. Esses neurônios "desligados" não contribuem para a propagação do sinal durante uma época de treinamento específica, portanto, a rede neural aprende a não depender muito de nenhum neurônio específico para prever os resultados.

Durante o treinamento, a rede neural é treinada em várias "versões" diferentes de si mesma, cada uma com um subconjunto diferente de neurônios "desligados". Ao forçar a rede neural a aprender a partir de diferentes "versões" de si mesma, a regularização dropout ajuda a prevenir o overfitting, pois o modelo aprende a generalizar para diferentes subconjuntos de neurônios, em vez de depender excessivamente de um conjunto específico de neurônios. Esse tópico foi abordado na Videoaula 11.

45. O que é uma rede RBF?

R: É uma rede alimentada adiante com múltiplas camadas, similar à rede MLP. Porém, ao contrário da rede MLP, os neurônios ocultos da rede RBF assumem funções de base radial. A rede RBF foi introduzida na Videoaula 13 e no material de apoio correspondente.

46. Qual é a arquitetura padrão de uma rede RBF?

R: A arquitetura padrão de uma rede RBF consiste em uma camada oculta formada por neurônios com função de base radial e uma camada de saída com neurônios lineares. A rede RBF foi introduzida na Videoaula 13 e no material de apoio correspondente.

47. Compare um neurônio RBF com um neurônio MCP.

R: O neurônio MCP é um neurônio binário e é responsável por criar uma fronteira de separação linear no espaço de atributos. Por sua vez, o neurônio RBF é um modelo de neurônio que usa uma função de base radial para transformar as entradas. O MCP atua de forma global (consegue enxergar todo o espaço), já o neurônio RBF constrói um campo receptivo local, ou seja, só responde a sinais posicionados dentro do seu campo receptivo. As unidades RBF foram introduzidas na Videoaula 13 e no material associado.

48. Como podemos treinar uma rede RBF?

R: A camada de saída de uma rede RBF é treinada de forma equivalente à rede MLP, ou seja, de forma supervisionada minimizando uma função de custo. Por sua vez, as unidades ocultas (RBF) podem ser treinadas de três formas principais:

1. Estabelecendo centros fixos sobre os exemplos de treinamento.
2. Abordagem hibrida, no qual os centros são ajustados via agrupamento não supervisionado de dados.
3. Forma supervisionada, utilizando o gradiente descendente sobre os parâmetros da função de base radial.

As redes RBF foram introduzidas na Videoaula 13 e no material associado.

49. Compare a rede RBF com uma rede MLP

R: Normalmente a MLP pode conter diversas camadas ocultas, a RBF apenas uma; os neurônios ocultos da RBF usam funções de base radial (distância entre entradas e o centro); os neurônios da MLP usam um combinador linear (produto interno entre pesos e entradas); as MLP constroem aproximações globais ao mapear a entrada-saída, as RBF aproximações locais. As redes RBF foram introduzidas na Videoaula 13 e no material associado.

50. Por que dizemos que uma rede SOM possui forte inspiração neurofisiológica?

R: Uma rede SOM (*Self-Organizing Map*), também conhecida como mapa auto-organizável de Kohonen, é inspirada em processos neurofisiológicos. Em particular, a rede SOM é inspirada na organização espacial do córtex cerebral. No cérebro, as informações sensoriais são mapeadas de forma topográfica em diferentes regiões do córtex cerebral. Essa organização espacial permite que o cérebro processe as informações de maneira eficiente e aprenda relações entre diferentes estímulos sensoriais. A rede SOM usa uma estrutura semelhante, as entradas são mapeadas em um espaço bidimensional de neurônios, em que a proximidade no mapa representa a similaridade nas características das entradas. A rede SOM foi introduzida na Videoaula 14 e no material correspondente.

51. Qual é a arquitetura padrão de uma rede SOM?

R: Em sua forma padrão, a rede SOM consiste em um grid 2D formado por neurônios. Contudo, há implementações de redes SOM utilizando estruturas com outras dimensões, e.g., 1D, 3D. A rede SOM foi introduzida na Videoaula 14 e no material correspondente.

52. O que cada neurônio em uma rede SOM representa?

R: Um neurônio na rede SOM representa um protótipo ou centro de um grupo de entradas similares. Em outras palavras, cada neurônio da rede SOM é um ponto em um espaço de características que representa as características médias de um conjunto de entradas que são mapeadas para esse neurônio. O vetor de pesos do neurônio está associado à sua respectiva célula de Voronoi. A rede SOM foi introduzida na Videoaula 14 e no material correspondente.

53. Como é realizado o treinamento de uma rede SOM?

R: O treinamento da rede SOM é um aprendizado competitivo e pertencente ao paradigma não supervisionado. Basicamente, o processo de treinamento consiste em três fases distintas: fase de competição, fase de cooperação e fase de ajuste sináptico. A rede SOM foi introduzida na Videoaula 14 e no material correspondente.

54. Como podemos avaliar a qualidade de uma rede SOM treinada?

R: Por se tratar de um modelo não supervisionado, não há como calcular, por exemplo, a acurácia do modelo, uma vez que não temos os valores desejados para cada neurônio. Contudo, há diversas abordagens para se avaliar o mapa gerado, por exemplo, podemos utilizar abordagens visuais como o hit map, heat map, U-Matrix, e mapas de pesos. Também podemos utilizar algumas métricas, como o erro de quantização, que verifica a área de cobertura de um dado neurônio (tamanho da célula) e erro topográfico, que verifica se o mapa está organizado espacialmente. A rede SOM foi introduzida na Videoaula 14 e no material correspondente.

55. Como definir a topologia de uma rede SOM (tamanho do grid)?

R: A topologia depende explicitamente da aplicação. Por exemplo, se o especialista quer ver um mapeamento geral dos dados, podemos usar um grid pequeno. Nesse caso, cada neurônio irá representar uma célula com padrões geralmente heterogêneos. Por outro lado, se o especialista desejar mapear os padrões em microclusters, grids maiores devem ser adotados, neste caso, cada neurônio irá representar uma região bastante homogênea do espaço de atributos.

56. O que é uma rede baseada em energia?

R: A energia da rede é definida em termos da função de energia total da rede, que é uma função matemática que representa o estado atual da rede. A função de energia total é dada pela soma ponderada dos estados de todos os neurônios e das conexões sinápticas entre eles. Cada estado da rede tem uma energia associada, e a rede tenta encontrar o estado de menor energia possível.

Basicamente, parte-se de um estado inicial e, passo a passo, os estados vão se alterando até atingir um mínimo local de energia. Há alguns tipos de redes que considerando essa abordagem baseada em energia, como a rede de Hopfield, seguindo um processo de decisão determinístico, e a máquina de Boltzmann que segue uma abordagem estocástica. As redes baseadas em energia foram introduzidas na Semana 6.

57. Explique a arquitetura de uma rede de Hopfield binária.

R: A rede de Hopfield binária consiste em um grafo completo (um clique) no qual cada vértice representa um neurônio binário. Todas as ligações são simétricas, ou

seja, o mesmo peso que liga o neurônio i ao neurônio j liga o neurônio j ao neurônio i. Não há laços de autorrealimentação, ou seja, um neurônio i não alimenta o próprio neurônio i. A rede de Hopfield foi introduzida na Videoaula 16 e no material correspondente.

58. O que são as memórias fundamentais?

R: São as memórias criadas, explicitamente, durante o processo de treinamento. As memórias fundamentais são representadas como pontos fixos (de equilíbrio) do sistema dinâmico representado pela rede. A rede de Hopfield foi introduzida na Videoaula 16 e no material correspondente.

59. O que são os estados espúrios?

R: São memórias (atratores) criados durante o processo de treinamento da Hopfield. Porém, essas memórias não constituem memórias fundamentais disponíveis no conjunto de treino, mas pontos fixos gerados de forma involuntária. Esse tópico foi abordado na Videoaula 16 e no material correspondente.

60. Como definimos o estado de cada neurônio na rede de Hopfield?

R: O estado de um dado neurônio é definido em função do seu gap de energia (campo local induzido), ou seja, precisamos calcular o produto interno entre o vetor de pesos do neurônio com o vetor de estados dos seus vizinhos. Se o valor do gap for maior que zero, o estado ligado (estado 1). Caso o gap seja menor que zero, o neurônio será desligado (estado 0 ou -1). Caso o gap seja igual a zero, o neurônio permanece com o estado atual. Abaixo apresento a fórmula de atualização do estado dos neurônios:

$$x_i = \begin{cases} 1 \text{ se } \sum_j x_j w_{ij} > 0 \\ -1 \text{ se } \sum_j x_j w_{ij} < 0 \\ x_i \text{ caso contrario} \end{cases}$$

No qual x_j representa o estado do neurônio j e w_{ij} o peso de ligação entre i e j. A rede de Hopfield foi introduzida na Videoaula 16 e no material correspondente.

61. Quais são as principais limitações da rede de Hopfield?

R: A rede de Hopfield possui baixa capacidade de memória, aproximadamente 0.14 x N, no qual N é o número de neurônios, ou seja, para uma rede com 100 neurônios, conseguiremos armazenar aproximadamente 14 memórias. Além disso, por se tratar de uma rede totalmente conectada, a matriz de pesos se tornará muito grande e densa. Por fim, durante o treinamento, além das memórias fundamentais, a rede também pode gerar estados de mínima energia que não foram explicitamente treinados. Esses estados são denominados atratores espúrios. A rede de Hopfield foi introduzida na Videoaula 16 e no material correspondente.

62. O que é o equilíbrio térmico associado às máquinas de Boltzmann?

R: A máquina de Boltzmann, ao longo da sua evolução, associa uma maior probabilidade aos estados de menor energia, logo, ao longo da evolução da rede, esta convergirá uma região de baixa energia, denominada equilíbrio térmico. No equilíbrio térmico os estados dos neurônios podem continuar oscilando, uma vez que a rede é estocástica, mas a energia global da rede tenderá a permanecer no mesmo patamar. A máquina de Boltzmann foi introduzida na Videoaula 17 e no material correspondente.

63. Defina a arquitetura da máquina de Boltzmann

R: A máquina de Boltzmann é formada por neurônios estocásticos totalmente conectados, ou seja, cada neurônio está conectado a todos os demais neurônios da rede (grafo completo – clique). Contudo, ao contrário da rede de Hopfield, a máquina de Boltzmann possui dois tipos de neurônios, os visíveis e os ocultos. Os neurônios visíveis representam a interface da rede com o ambiente, ou seja, esses neurônios são responsáveis por receber os padrões do conjunto de dados e entregar os padrões gerados pelo modelo. Os neurônios ocultos, por sua vez, podem ser vistos como extratores de características e são responsáveis por encontrar as correlações entre os estados dos neurônios da camada visível. A máquina de Boltzmann foi introduzida na Videoaula 17 e no material correspondente.

64. Como a máquina de Boltzmann pode ser treinada?

R: O treinamento da máquina de Boltzmann envolve duas fases, a fase positiva (presa) e a fase negativa (solta). Na fase positiva, a camada visível é fixada em um padrão do conjunto de treinamento e os neurônios ocultos são livres para oscilar até atingir o equilíbrio térmico. Na fase negativa, todos os neurônios podem oscilar livremente na busca pelo equilíbrio térmico. Tanto para a fase positiva quanto para a fase negativa, ao atingir o equilíbrio térmico, nós calculamos as respectivas correlações entre todos os pares de neurônios. Essas correlações são utilizadas para ajustar os pesos da rede. Basicamente, a fase positiva busca reduzir a energia dos estados desejáveis (presentes no conjunto de treinamento) e a correlação obtida na fase negativa é responsável por ampliar a energia dos estados naturais da máquina, ou seja, aqueles estados para os quais a máquina irá convergir naturalmente. O processo de treinamento é interrompido quando ambas as correlações estiverem muito próximas, indicando que a máquina, na fase negativa, está se comportando conforme esperado, ou seja, gerando padrões similares aos presentes no conjunto de treinamento. A máquina de Boltzmann foi introduzida na Videoaula 17 e no material correspondente.

65. Como os neurônios em uma máquina de Boltzmann mudam seus estados?

R: O estado do neurônio depende do gap de energia. Quanto maior for o gap, maior será a probabilidade de ligar o neurônio. Por outro lado, quanto menor for o gap (mais negativo), maior será a probabilidade de ligar o neurônio. A máquina de Boltzmann foi introduzida na Videoaula 17 e no material correspondente.

66. Como podemos utilizar a máquina de Boltzmann?

R: A máquina de Boltzmann pode ser considerada em três tarefas principais:

Inferência: instancia-se um padrão disponível na camada visível. Após, a MB evolui até atingir o equilíbrio térmico. O padrão de saída é formado na camada visível (saída).
Explicação: mantém-se a camada visível fixa e observa-se o padrão formado nos neurônios ocultos no equilíbrio térmico.
Geração: a partir de um estado inicial aleatório, a máquina evolui até o equilíbrio térmico. Após, realiza-se amostragens na camada visível.

67. Quais são as principais limitações da máquina de Boltzmann?

R: As principais limitações da máquina de Boltzmann são:

1) O cálculo das correlações nas fases positiva e negativa é muito custoso computacionalmente.
2) Precisamos de uma grande quantidade de passos de atualização de todos os neurônios da rede para atingir o equilíbrio térmico.
3) Na prática, a máquina de Boltzmann original tem pouca aplicação.

Veja a explicação na Videoaula 17 e no material associado.

68. O que é a máquina restrita de Boltzmann? Qual é a principal diferença para a máquina de Boltzmann original?

R: A máquina restrita de Boltzmann (RBM) é um tipo de rede neural artificial que faz parte da família de modelos de redes neurais baseadas em energia. A RBM é composta por uma camada visível e uma camada oculta, e é usada para modelar a distribuição conjunta de probabilidade entre as camadas.

A principal diferença entre a RBM e a máquina de Boltzmann original é que a RBM é "restrita" em suas conexões. Na RBM, cada unidade na camada visível está conectada a todas as unidades na camada oculta, mas não há conexões entre unidades na mesma camada. Essa restrição torna a RBM mais fácil de treinar e mais eficiente em termos de cálculo do que a máquina de Boltzmann original, que tem conexões entre todas as unidades em ambas as camadas. A RBM foi introduzida na Videoaula 18 e no material correspondente.

69. Por que a fase positiva na RBM é menos custosa que na máquina de Boltzmann original?

R: Como não há ligações entre neurônios de uma mesma camada, ao fixar os neurônios visíveis na fase positiva, com apenas um passo conseguimos atingir o equilíbrio térmico na RBM. Por outro lado, na máquina de Boltzmann original precisaríamos de diversas iterações para atingir o mesmo equilíbrio, uma vez que a alteração no estado de um dado neurônio oculto pode influenciar nos estados de outros neurônios dessa mesma camada. A especificação da RBM foi apresentada na Videoaula 18 e no material correspondente.

70. Explique o algoritmo de divergência contrastiva. Como ele pode ser utilizado para treinar uma RBM?

R: O algoritmo de divergência contrastiva (CD) é um método de treinamento utilizado para ajustar os parâmetros de uma máquina restrita de Boltzmann (RBM). O CD é um algoritmo baseado em gradiente que permite que a RBM aprenda a

representar os padrões nos dados de entrada. Ao contrário do método tradicional, o CD não aguarda a máquina atingir o equilíbrio térmico na fase negativa, mas realiza apenas um (ou alguns) passos de reconstrução visando encontrar a direção de atualização. Basicamente, o algoritmo conste em quatro passos para calcular as correlações das fases positiva e negativa:

1) A camada visível é fixada com um padrão da base de treino.
2) Atualiza os estados das unidades ocultas e calcula as correlações do passo 0 (positivo).
3) Atualiza os estados das unidades visíveis (reconstrução).
4) Atualiza os estados das unidades ocultas novamente e calcula as correlações do passo 1 (estimativa da fase negativa).

O processo é repetido iterativamente até que a máquina esteja treinada. Além do CD com 1 passo, também podemos utilizar o algoritmo com mais passos na fase negativa, minimizando as chances de preservar mínimos locais indesejáveis na superfície de energia da máquina. O algoritmo de divergência contrastiva (CD) foi introduzido na Videoaula 18 e no material correspondente.

71. Diferencie as redes recorrentes autônomas das redes recorrentes não autônomas. Cite exemplos de cada uma.

R: Redes autônomas: normalmente a entrada é fixa e a rede evolui dinamicamente a partir dessa entrada fornecida e.g., rede de Hopfield e máquina de Boltzmann.

Redes não autônomas: a entrada pode variar no tempo [x(t)], dados sequenciais, e.g., redes RNN, GRU, LSTM.
Esse conteúdo foi abordado na Videoaula 19.

72. Cite algumas aplicações de redes recorrentes.

R: As redes recorrentes (RNNs) são amplamente utilizadas em várias áreas de aplicação em que há dependência temporal nos dados, permitindo que a rede possa modelar e aprender a partir de sequências de dados. Algumas aplicações comuns de RNNs: reconhecimento e transcrição de fala; geração de música, geração de texto; classificação de sentimentos; reconhecimento de cenas em vídeos; tradução, dentre diversas outras.
Algumas dessas aplicações foram visitadas na Videoaula 19 e no material associado.

73. Quais são os principais tipos de arquiteturas de redes recorrentes? Cite aplicações para cada tipo de arquitetura.

R: Há diversos tipos de arquiteturas de redes recorrentes, dentre elas: um-para-muitos, muitos-para-um e muitos-para-muitos, sendo que esta última pode assumir duas formas: 1) arquiteturas com sequências de entradas e saídas sincronizadas e a 2) arquitetura encoder-decoder.

Aplicações das arquiteturas:

um-para-muitos – geração de texto, geração de áudio etc.

muitos-para-um – análise de sentimentos, classificação de sentenças, predição em séries temporais etc.

muitos-para-muitos – 1) classificação de tokens, análise de sequências de DNA etc. e 2) tradução, geração de resumo etc.

Essas arquiteturas foram introduzidas na Videoaula 19 e no material associado.

74. É possível utilizar uma rede recorrente para processamento de textos? Como podemos representar os dados nesse cenário?

R: Sim, é possível. Normalmente, as palavras contidas no texto são representadas por vetores binários (*one-hot-encoding*). Se considerarmos que o vocabulário tem 10 mil palavras, podemos utilizar um vetor com 10 mil posições, na qual todas as posições do vetor, exceto a indicação do token, serão iguais a zero.

Por exemplo, para o vocabulário abaixo, com apenas quatro palavras (tokens), teríamos:

Amarelo → [0 0 0 1]
Vermelho → [0 0 1 0]
Azul → [0 1 0 0]
Roxo → [1 0 0 0]

Dessa forma, a entrada do modelo deve receber, de forma sequencial, cada token codificado neste vetor one-hot a fim de processar a sequência. Outra opção é utilizar um método de embedding, como o Word2Vec para codificar cada token em uma representação densa e utilizar essa codificação de embedding como entrada do modelo de rede neural recorrente. Esse tópico foi abordado na Videoaula 19.

75. O que é o algoritmo de retropropagação através do tempo?

R: Consiste uma aplicação do próprio algoritmo de retropropagação, porém na rede "desenrolada", ou seja, além das dependências espaciais que observamos ao aplicar o algoritmo de retropropagação em uma rede MLP, no qual o gradiente de um neurônio k depende dos gradientes de todos os neurônios da camada posterior que k alimenta, no algoritmo através do tempo, também temos a dependência temporal, ou seja, o gradiente do neurônio k no tempo t depende do gradiente calculado nesse mesmo neurônio k no tempo t+1 (retropropagação temporal).

O algoritmo de retropropagação através do tempo (*backpropagation through time*) foi apresentado na Videoaula 19.

76. Por que o problema do desaparecimento/explosão do gradiente é mais evidente em redes recorrentes?

R: A dependência temporal agrava esse problema. Ou seja, os neurônios de uma camada k recorrente, além de depender dos gradientes dos neurônios de uma camada espacialmente posterior, também depende dos gradientes dos neurônios da própria camada k no instante posterior (dependência temporal). Se a sequência for longa, o cálculo do gradiente durante a retropropagação pode divergir ou convergir para zero a depender dos valores dos pesos associados a estes neurônios. Por exemplo, se um dado peso é inferior a 1, digamos 0,9, se a sequência for de tamanho 100, esse valor vai ser multiplicado 100 vezes, ou seja, $0{,}9^{100}$, tornando-se muito

próximo de zero, isso também vale para valores superiores a 1, porém, neste caso, o valor explode. Esse problema foi explicado na Videoaula 20 e no material associado.

77. Quais são as principais limitações das redes recorrentes padrão (RNN vanilla)?
R: Esta rede sofre com o problema do gradiente (explosão/desaparecimento) e não possui memória de longo prazo, portanto, possui limitações para tratar sequências com dependências de longo prazo.

78. Por que as redes LSTM e GRU conseguem resolver o problema do desaparecimento do gradiente?
R: As redes LSTM (*Long Short-Term Memory*) e GRU (*Gated Recurrent Unit*) são redes neurais recorrentes projetadas especificamente para lidar com o problema do desaparecimento do gradiente em sequências longas. Elas conseguem resolver esse problema utilizando portas para controlar o fluxo de informação na rede, permitindo que as informações importantes sejam mantidas e propagadas ou retropropagadas com mais facilidade através das camadas.

79. Compare as células LSTM com as células GRU.
R: As principais diferenças entre as duas arquiteturas são: 1) o número de portas que cada unidade de memória contém. Enquanto a LSTM tem três portas (porta de entrada, porta de esquecimento e porta de saída), a GRU tem apenas duas portas (porta de atualização e porta de relevância em sua versão completa). 2) o número de sinais internos. A LSTM possui dois sinais distintos para mapear a memória de longo e curto prazo. Já a rede GRU possui apenas um sinal interno responsável por lidar com as memórias de curto e longo prazo. A Videoaula 20 aborda esse tópico.

80. Com relação ao número de parâmetros, o que podemos afirmar ao compararmos uma RNN padrão, formada por células tangente hiperbólica, com uma rede formada por células LSTM ou GRU?
R: Uma rede tradicional possui apenas matrizes de pesos associadas às conexões existentes na rede. Por sua vez, as redes LSTM e GRU, além das matrizes tradicionais, possuem matrizes de pesos associadas aos gates. Ou seja, a LSTM terá quatro matrizes de pesos em cada camada LSTM, a GRU terá duas ou três matrizes em cada camada, a depender do tipo de célula utilizada, enquanto a RNN padrão terá apenas uma matriz de peso por camada.

81. Seja uma rede GRU com uma única camada GRU com 15, alimentada por um sinal x de 5 dimensões e uma única unidade de saída. Considerando que essa rede utilizada a célula GRU simplificada, com apenas o gate de atualização, quantos parâmetros livres há nesse modelo?
R: Considerando que temos cinco entradas (x possui 5 dimensões) e 15 neurônios na camada GRU, cada neurônio na camada GRU irá receber 20 entradas, sendo as cinco externas (entradas - x) e 15 referentes aos estados dos próprios neurônios no instante anterior. Logo, a matriz de pesos Wa, associada à geração do sinal candidato, terá tamanho 15x20, totalizando 300 parâmetros livres. Além disso, o cálculo do sinal candidato também depende do bias, como temos 15 neurônios,

temos 15 parâmetros livres para o bias. Somando tudo, totalizamos 315 parâmetros livres associados ao cálculo do sinal candidato.

A mesma conta realizada acima pode ser utilizada para o cálculo dos parâmetros associados ao gate de atualização, ou seja, temos 315 parâmetros associados à geração do sinal de atualização (gate de atualização).

Por fim, como temos apenas um neurônio na camada de saída e este é alimentado pelos 15 neurônios da camada GRU, temos um total de 15 pesos + o bias desse neurônio, totalizando 16 parâmetros livres na camada de saída.

O modelo, como um todo, terá um total de 646 parâmetros ajustáveis.

Uma fórmula fechada para calcular o número de parâmetros (P) dessa rede é:

$P = C\ (n^2 + nm + n)$, no qual C representa o número de matrizes associadas às unidades GRU da rede. Como estamos utilizando a célula simplificada (com apenas um gate), temos duas matrizes, logo C=2. n representa o número de neurônios na camada GRU e m representa o número de entradas (dimensão de entrada), cinco no exemplo acima.

Assim, temos: $P = 2\ x\ (15^2 + 15x5 + 15) = 630$ parâmetros na camada GRU, somando aos parâmetros da camada de saída (16), totalizamos 646.