Resolução simultânea de problemas distintos em única rede neural artificial

Guilherme F. Marchezini
Daniel P. B. Lopes
Daniel S. da Fonseca
Lauro C. J. Santos
Taís R. Silva

Resumo

O objetivo deste trabalho é averiguar e avaliar a eficácia de estratégias no desenvolvimento de Redes Neurais Artificiais que solucionem múltiplos problemas. Este problema foi pouco investigado até agora e pode ser útil em diversas aplicações nas quais mais de uma rede neural seja necessária. Para tanto, foram implementadas duas arquiteturas diferentes que resolvem dois problemas de classificação, sendo que em uma destas arquiteturas foi possível obter um resultado com grande eficácia na resolução de um dos problemas e indícios de que a rede estava aprendendo a solucionar o segundo, alcançando uma acurácia superior a 50% para cada um deles. O resultado promissor aponta para maiores investigações na área utilizando redes mais complexas.

1 Introdução

Ao longo de algumas décadas, estudos científicos demonstraram que as redes neurais artificiais (RNAs) possuem uma enorme capacidade de reconhecimento de padrões, o que faz com que elas sejam consideradas ferramentas poderosas na resolução de problemas de classificação [ZPH98]. Além da classificação de padrões, atualmente as RNAs já estão consolidadas e possuem vasta aplicabilidade em problemas relacionados a previsão de séries temporais, agrupamentos, aproximação de funções, otimização combinatória clássica, dentre muitos outros que se beneficiam das propriedades distintivas presentes nas RNAs.

Segundo [dSSF+17], dentro de uma arquitetura particular, a topologia de uma determinada rede neural pode ser definida a partir das diferentes composições estruturais que ela pode assumir, de modo que é possível em uma mesma arquitetura desenvolver topologias compostas por diferentes quantidades de neurônios. Além disso, o comportamento de um neurônio é diretamente influenciado pela função de ativação associada a ele, de modo que uma rede neural pode apresentar desempenhos completamente diferentes, ainda que sob uma mesma topologia. Em outras palavras, é possível obter uma grande variabilidade de redes neurais, com diferentes comportamentos, a partir de modificações relativamente sutis. Essa característica faz com que, em geral, as redes neurais sejam bastante específicas ao problema tratado.

Por outro lado, como apontado por [dSSF+17], o treinamento de uma arquitetura particular envolve a aplicação de uma metodologia sistematizada para ajustar os pesos e limiares dos neurônios. Esse processo, conhecido como algoritmo de aprendizagem, tem como objetivo o ajuste da rede no intuito de alcançar a generalização das predições, i. e., fazer com que as saídas fornecidas pela rede apresentem resultados próximos aos desejados, ainda que a partir de entradas de dados desconhecidas.

Embora esta especificidade das RNAs seja bastante pertinente, dada a importância de obter-se resultados sempre confiáveis e precisos, a necessidade de gerar uma rede neural para cada problema analisado pode se tornar um grande inconveniente. Há situações, por exemplo, que demandam o processamento de mútiplas fontes de conhecimento em um mesmo sistema. Redes neurais que processam e solucionam múltiplos problemas podem simplificar o gerenciamento de sistemas e a implantação de produtos. Desta forma, a utilização de uma rede neural mais generalista, capaz de suportar o processamento simultâneo de diferentes problemas resultaria em um ganho de desempenho considerável, a depender do tipo de aplicação.

Este trabalho propõe o conceito de uma rede neural capaz de solucionar diferentes problemas de classificação de forma simultânea. Como objetivo, pretende-se avaliar o desempenho de duas arquiteturas projetadas a partir de diferentes tipos de configuração na camada de entrada, considerando dois cenários de classificação distintos. As contribuições deste trabalho são apresentadas a seguir:

- Possibilita uma análise útil sobre a viabilidade de se implementar redes neurais com resolução simultânea de problemas distintos, dada a carência de estudos neste contexto.
- Aponta um segmento de pesquisa de grande relevância ao universo de aplicações das redes neurais artificiais, com um estudo bem fundamentado a respeito da resolução de múltiplos problemas em uma mesma RNA.
- Propõe metodologias e apresenta uma análise a respeito da eficácia de diferentes tipos de combinações de entrada para melhorar o resultado em múltiplos problemas.
- Apresenta resultados quantitativos e qualitativos para duas abordagens propostas já implementadas.

O restante deste trabalho está organizado da seguinte forma: a Seção 2 busca contextualizar o cenário de pesquisa em que este trabalho está inserido, descrevendo alguns estudos correlatos e explicitando trabalhos anteriores. A Seção 3 apresenta a metodologia empregada neste trabalho, que consiste nas etapas de tratamento dos dados, geração da entrada do algoritmo, bem como as estratégias de modelagem e arquitetura das RNAs desenvolvidas. A Seção 4 apresenta e analisa os resultados obtidos. Finalmente, a Seção 5 conclui o artigo e apresenta perspectivas de trabalhos futuros.

2 Trabalhos relacionados

As primeiras referências de processamento simultâneo em redes neurais artificiais surgiram a partir de aplicações em problemas de programação matemática, nomeadamente na resolução simultânea de sistemas de equação linear. Um dos trabalhos precursores nesta área foi a rede neural eletrônica proposta por [Wan92]. Baseada em amplificadores operacionais, essa rede é capaz de gerar soluções em tempo real para uma vasta quantidade de equações lineares simultâneas. No ano seguinte, [Wan93] trouxe a proposta de uma rede neural recorrente de duas camadas, conectadas de forma bidirecional, que se mostrou assintoticamente estável no cálculo de inversão de matrizes e na resolução de equações matriciais de Lyapunov. Posteriormente, no trabalho de [SM94] foram apresentados dois métodos para resolução de sistemas lineares utilizando RNAs, ambos baseados na rede de Hopfield e Tank. Inspirado pelo trabalho de [Wan92], [Che13] apresentou recentemente um modelo neural baseado em um algoritmo de gradiente para resolução simultânea de equações lineares, com o objetivo específico de aplicar o modelo desenvolvido na solução do Problema de Programação Quadrático com Restrições de Igualdade.

Existe uma abordagem, conhecida como "aprendizado multitarefa"(MTL, *multi-task learning)*, que busca melhorar a aprendizagem de uma tarefa utilizando as informações contidas nos sinais de treinamento de outras tarefas relacionadas. O propósito dessa abordagem é fazer com que, ao compartilhar representações entre tarefas relacionadas, seja possível que o modelo desenvolvido generalize melhor na tarefa original. Os trabalhos mais recentes no domínio do aprendizado multitarefa estão vinculados ao uso de *Deep Neural Networks*, ou redes neurais de aprendizado profundo. O trabalho de [Rud17] apresenta uma visão global da literatura, discute técnicas e avanços recentes do MTL com aprendizagem profunda. Especificamente sobre aplicações que sugerem as RNAs para problemas de classificação, destaca-se o trabalho de [LQH17], cujo propósito é utilizar o MTL para classificação textual. Diferentemente do aprendizado multitarefa, a estratégia desenvolvida neste trabalho tem como objetivo o processamento simultâneo de tarefas independentes, que não estão necessariamente relacionadas. Logo, pretende-se alcançar um desempenho satisfatório na generalização de todas as tarefas processadas.

A possibilidade de solucionar vários problemas por meio de uma única RNA é amplamente ambicionada em estudos científicos. A maior motivação acadêmica está na alusão que as redes neurais artificiais fazem às estruturas neurais biológicas, e que os métodos artificiais de aprendizagem fazem aos processos cognitivos. Diferentemente de um cérebro humano, capaz de reter o conteúdo de

diversos processos de aprendizagem sequenciais, as RNAs clássicas perdem as informações adquiridas a cada novo processo de treinamento, pois as informações contidas nos pesos das conexões da rede gerada para uma dada tarefa são substituídas através do cálculo de novos pesos para atender a tarefa seguinte. Diversos trabalhos se dedicam a vários anos a desenvolver estratégias capazes de amenizar ou de superar esse fenômero, conhecido como *catastrophic forgetting*. O trabalho de [VC17], por exemplo, simula uma RNA com liberação de substâncias químicas neuromoduladoras, i. e., substâncias difundidas dentro de uma RNA modular, capaz de aumentar ou diminuir a regularização em uma região espacial. Essa estratégia induz a aprendizagem específica de uma tarefa em grupos de nós e conexões, gera módulos funcionais para cada tarefa e, consequentemente, consegue eliminar o "esquecimento catastrófico". O trabalho de [KPR+16], por sua vez, propõe um algoritmo semelhante ao algoritmo de consolidação sináptica para RNAs, com o intuito de retardar o aprendizado em determinados pesos, de acordo com a importância de tarefas anteriormente vistas. Esse algoritmo pode ser empregado em aplicações que utilizam aprendizagem supervisionada ou aprendizagem por reforço. Dentre outros trabalhos inseridos nesse contexto, destaca-se também o trabalho de [MW17], em que se propõe um modelo neural dependente de pelo menos dois programas neurais, um para representar a soma neuronal, e outro a dendrite, ambos considerados essenciais para permitir que os neurônios movam-se, mudem-se, repliquem-se, ou morram.

3 Metodologia

3.1 *Datasets*

O processo de aprendizagem de uma rede neural artificial tem como objetivo o ajuste dos pesos e limiares das conexões de cada neurônio de modo que, a partir das informações presentes nas características (*features*) de determinada base de dados, seja possível estabelecer um relacionamento entre as entradas e saídas fornecidas e adquirir a capacidade de generalizar soluções. Essas características, portanto, são específicas de cada problema e requerem um treinamento igualmente específico. Tendo-se em vista que proposta deste trabalho é resolver dois problemas distintos simultaneamente, então foram escolhidas duas bases de dados com o intuito de relacionar *features* completamente diferentes.

O primeiro problema é descrito pela base de dados *Adult Dataset* [1], que contém informações de pessoas residentes dos Estados Unidos, e deseja-se discriminar cada uma dessas pessoas em duas classes, rotuladas como indivíduos cuja renda anual excede, ou não, 50 mil dólares. Algumas das propriedades mais significativas desse *dataset* são apresentadas na Tabela 1.

Características dos dados:	Multivariável	**Número de instâncias:**	48842
Características do atributo:	Categórico, Inteiro	**Números de atributos:**	14
Tarefas associadas:	Classificação	**Valores perdidos?**	Yes
Área:	Social	**Acertos na Web:**	1026885

Tabela 1: Propriedades estatísticas do *Adult Dataset*.

O segundo problema refere-se ao conjunto de dados *Mushroom Dataset* [2], que inclui descrições de amostras hipotéticas correspondentes a 23 espécies de cogumelos picados na família *Agaricus* e *Lepiota*. Cada espécie é identificada como definitivamente comestível, definitivamente venenosa, ou com a propriedade "aptidão para consumo desconhecida e não recomendada". Para simplificar o processo de classificação, essa última classe foi combinada com a classe venenosa. A Tabela 2 apresenta um resumo com algumas das propriedades mais significativas desse *dataset*.

Características dos dados:	Multivariável	**Número de instâncias:**	8124
Características do atributo:	Categórico	**Números de atributos:**	22
Tarefas associadas:	Classificação	**Valores perdidos?**	Yes
Área:	Vida	**Acertos na Web:**	295101

Tabela 2: Propriedades estatísticas do *Mushroom Dataset*.

[1] https://archive.ics.uci.edu/ml/datasets/adult
[2] https://archive.ics.uci.edu/ml/datasets/mushroom

3.2 Tratamento dos dados

Para garantir o funcionamento correto das RNAs e permitir que os dados sejam estruturados da forma desejada para cada uma das soluções propostas, é necessário considerar uma etapa prévia à execução do algoritmo, em que acontece um tratamento dos dados.

Os dados provindos dos *datasets* escolhidos não devem ser imediatamente fornecidos ao algoritmo, uma vez que as RNAs recebem apenas dados numéricos como entrada. Como mostrado na Seção 3.1, existem *features* categóricas presentes nos *datasets* que não são numéricas, mas sim representadas por conteúdo textual. Para tornar a entrada de dados compatível ao algoritmo, é necessário substituir cada categoria das classes categóricas por um número. Nesse caso, por exemplo, a *feature* Race poderia ter valor 1 para White, 2 para Asian-Pac-Islander, 3 para Amer-Indian-Eskimo, 4 para Other, e 5 para Black. Essa transformação foi feita em todas as *features* que estavam com tipo de dados não-numéricos.

Um dos grandes problemas existentes após a transformação dos *datasets* é a presença de *features* categóricas e contínuas como números. Por tratarem dados de entrada e pesos, os algoritmos de RNAs fazem uma divisão do espaço considerando uma ordem numérica crescente. Logo, para o algoritmo o número 1 é menor que o número 2 e isso é válido para os contínuos, mas para os categóricos que foram transformados em números a premissa não é válida, pois White considerado 1 não vale menos que Asian-Pac-Islander classifado como 2. Como solução ao problema descrito, optou-se por utilizar um método conhecido como *feature dummyfication*, que cria uma nova *feature* para cada valor das classes categoricas, atribuindo 1 para quem é dessa classe, e 0 para quem é de outra. Esse método aumenta consideravelmente o número de variáveis, embora seja necessário para possiblitar a execução correta das RNAs.

Por fim, cada um dos *datasets* foi usualmente dividido em duas partes, uma utilizada para o treinamento da rede, correspondente a 70% dos dados, e a outra contendo os outros 30% destinados ao teste.

3.3 Modelagem proposta

O maior desafio existente na proposta de resolução simultânea de múltiplos problemas em uma mesma RNA diz respeito a como são combinadas as entradas e saídas dos diferentes *datasets* para que as redes sejam capazes de extrair os padrões existentes nas *features* e obter um modelo de generalização individual para cada um dos problemas. Nesta seção são propostos dois métodos distintos como solução para esse desafio, sendo que cada um dos métodos estabelece uma relação diferente entre as camadas de entrada e a saída da rede, o que constitui a principal diferença entre os métodos.

O primeiro método consiste em combinar todas as entradas, de forma que uma instância de entrada, i. e., uma linha do *dataset*, contenha as informações de ambos os problemas, bem como a saída das duas bases de dados. Desta forma, é possível passar simultaneamente a entrada dos dois problemas, de forma que na saída serão resolvidos ambos. Para este método, a camada de saída da RNA contém dois neurônios, sendo que cada um deles é responsável por produzir e apresentar a saída final da rede relativa a um dos problemas, resultantes do processamento realizado pelos neurônios nas camadas anteriores . Em outras palavras, o primeiro neurônio apresenta a classe predita para a entrada referente ao primeiro *dataset*, e o segundo neurônio apresenta a classe predita para a entrada referente ao segundo *dataset*.

Assim como no primeiro método, o segundo propõe utilizar todas as colunas de ambos os *datasets*. No entando, o segundo método é feito para resolver apenas um problema por ver, para isso as entradas foram estruturadas de forma que apenas as *features* do problema que se deseja resolver deve atribuir valores, enquanto as *features* dos outro problema será inserida os valores padrão -1. A saída também possui estrutura diferente, pois ao invés de se usar a abordagem do primeiro método com dois neurônios, nesse caso é utilizado quatro neurônios com a função *Softmax* que produz uma probabilidade de a resposta ser aquele neurônio. Para fazer sentido essa saída, foi atribuída a ideia de um contra todos, a maior probabilidade indica que ele está resolvendo aquele problema com aquela classe. Essa técnica pode produzir efeitos não desejados, como atribuir a classe do problema 2 como solução ao problema 1.

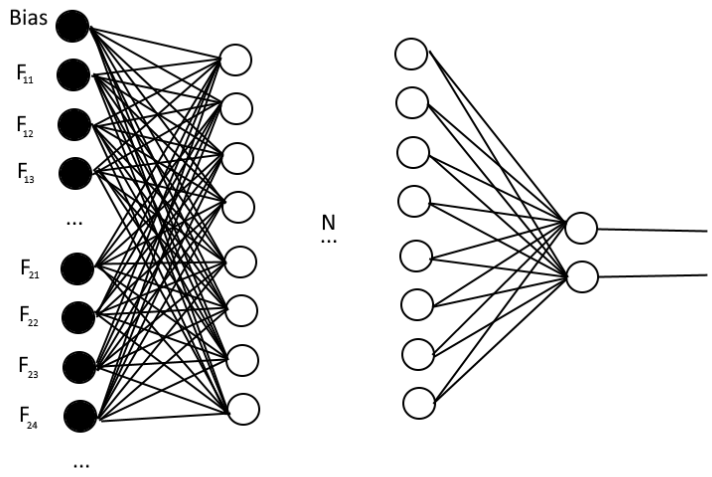

Figura 1: Esta figura representa um esquema ilustrativo da RNA correspondente ao primeiro método de solução. Nota-se que a camada de saída contém dois neurônios, cada qual contendo o valor da predição de classe para um problema distinto.

3.4 Arquitetura das RNAs

Em ambos os modelos propostos, a estratégia utilizada foi desenvolver uma rede MLP (*Multilayer Perceptron*) com uma grande quantidade de camadas escondidas, sendo que a maior quantidade de camadas é justificada pela suposição de que a rede utiliza diferentes combinações para que as entradas de um problema não interfiram na classificação do outro problema.

A RNA referente ao primeiro modelo apresentado é construída com N camadas escondidas, em que o valor N foi escolhido por meio de uma variação gradativa de valores, com o intuito de avaliar a melhora ou piora dos resultados. Essa rede possui na saída apenas dois neurônios com função de ativação *sigmoid*, de forma que, cada neurônio com valor igual a 1 significa uma classe de um problema, e 0 a outra classe. Essa abordagem é possível apenas se a rede é treinada colocando as entradas dos dois problemas ao mesmo tempo, de forma que o aprendizado e a predição necessitam sempre dos dados de dois problemas e da saída para os dois problemas para funcionar corretamente.

A segunda RNA apresentada também possui N camadas escondidas para poder processar o resultado. O diferencial dessa rede é que a camada de saída utiliza a função *Softmax*, que oferece um porcentual de chances para cada uma das classes. Por funcionar assim a rede foi alterada para que cada entrada possuisse os dados de apenas um problema enquanto as features do outro problema possuiam valor padrão de -1. Os outputs foram organizados de forma que os neurônios de saída 0 e 1 representavam as classes de um problema e os neurônios 2 e 3 representavam as classes do outro problema. Dessa forma a Rede é treinada para receber a entrada e resolver apenas um problema por vez.

Para ambas as estratégias foram testados valores de N iguais a 2, 4 e 13, sendo que na primeira estratégia, com duas e quatro camadas, cada uma é composta por 8 neurônios e função de ativação *sigmoid*. A segunda estratégia corresponde a uma RNA com duas e quatro camadas, que são compostas de 8 neurônios com função de ativação *relu*. Para a rede com 13 camadas, a segunda camada e a última são Camadas de *Dropout*. Essas camadas aleatoriamente escolhem neurônios para serem ignorados, dessa forma evita que a rede se especialize e dependa demais de um único neurônio, forçando os pesos a serem distribuídos.

4 Resultados

Após o processo de treinamento, cada uma das arquiteturas foi testada utlizando a parte de teste dos *datasets*. A partir das predições encontradas, foram calculadas as métricas acurácia e taxa de acerto de cada classe. A acurácia se refere ao percentual de instâncias que a rede previu

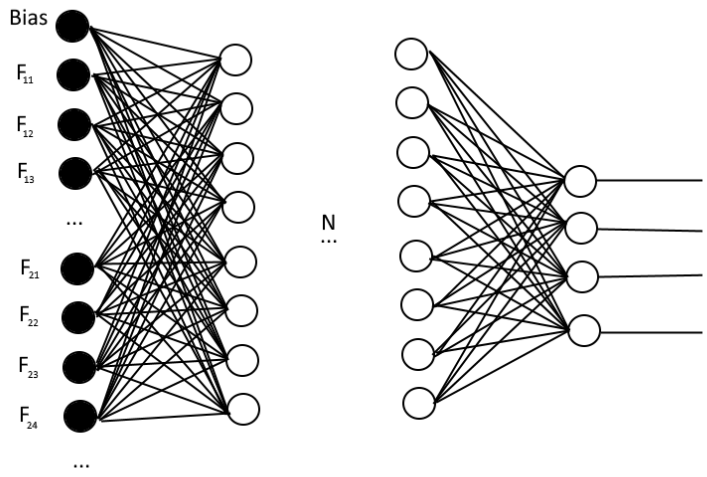

Figura 2: Esta figura representa um esquema ilustrativo da RNA correspondente ao primeiro método de solução. Nota-se que a camada de saída contém quatro neurônios, ...

corretamente, enquanto a taxa de acerto de cada classe mostra especificamente a porcentagem de acerto para as instâncias de cada classe específica. A Tabela 3 apresenta os principais resultados de acurácia obtidos com diferentes números de camadas em cada arquitetura. É possível visualizar que a acurácia para a Arquitetura 1 se manteve quase constante em todos os casos, enquanto a Arquitetura 2 obteve um aumento de desempenho com o aumento da complexidade.

	Arquitetura 1		**Arquitetura 2**
Camadas escondidas	problema 1	problema 2	problema total
2	0,7441	0,49	0,6733
4	0,7441	0,5217	0,835
13	0,7441	0,5062	0,8326

Tabela 3: Avaliação dos resultados obtidos a partir da métrica de acurácia.

A Tabela 4 apresenta os respectivos resultados da taxa de acerto para cada classe em cada arquitetura. Estes resultados nos permitem entender melhor o comportamento e o desenvolvimento da rede na resolução destes problemas. A Arquitetura 1 se mostrou bastante ineficaz, ao invés de aprender como realizar a classificação dos problemas, ela apenas se fixava em classificar uma única classe para cada um dos problema. O resultado da Arquitetura 2 foi mais promissor, com uma rede pouco complexa ela resolvia o segundo problema, não gerando nenhuma classificação para o primeiro, porém aumentar a complexidade lhe concedeu maior capacidade de aprender a resolver ambos, sendo que com 4 camadas ela começou a resolver o problema 1 de maneira simplória, apontando a classe 1 para todos os casos, e com 13 camadas, das quais duas eram de dropout, ela passou a realizar uma classificação com uma taxa de acerto superior a 50% para o problema 1 e manteve uma taxa alta para o problema 2.

	Arquitetura 1				**Arquitetura 2**			
Camadas escondidas	problema 1		problema 2		problema 1		problema 2	
	C0	C1	C0	C1	C0	C1	C0	C1
2	1	0	1	0	0	0	1	1
4	1	0	0,032	0,999	0	1	1	1
13	1	0	0	1	0,588	0,502	1	0,8228

Tabela 4: Acerto de cada classe.

Com as informações de ambas as tabelas é possível ver que, embora a acurácia para a resolução da arquitetura 2 com 13 camadas tenha reduzido levemente em relação a obtida para 4 camadas, a resolução geral do problema ainda foi melhor, pois ela passou a resolver ambos os problemas, justificando-se essa acurácia principalmente por causa da piora na resolução do problema 2 para a classificação da classe 1.

5 Considerações finais

Através deste trabalho, pode-se comprovar que é possivel resolver mais de um problema em uma mesma rede neural, mas que tal tarefa pode ser mais eficiente e precisa, visto que as técnicas apresentadas aqui são apenas ideias iniciais e podem ser aprimoradas. Como trabalhos futuros podem ser desenvolvidas técnicas mais precisas do que as apresentadas aqui, explorando outras arquiteturas mais complexas e utilizando técnicas de *deep learning*. Também é interessante analisar as possibilidades de resolver mais de dois problemas simultaneamente, pois as aplicações reais naturalmente vão resolver a maior quantidade de problemas possíveis. Outra área que pode ser estudada dentro desse tema é a forma de aproveitar a relação entre os diferentes problemas.

Referências

[Che13] Linear simultaneous equations; neural solution and its application to convex quadratic programming with equality-constraint. *Journal of Applied Mathematics*, page 6, 2013.

[dSSF+17] Ivan Nunes da Silva, Danilo H. Spatti, Rogerio Andrade Flauzino, Luisa Helena Bartocci Liboni, and Silas Franco dos Reis Alves. *Artificial Neural Networks*, chapter 2, pages 21–28. Springer, 2017.

[KPR+16] James Kirkpatrick, Razvan Pascanu, Neil C. Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A. Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, Demis Hassabis, Claudia Clopath, Dharshan Kumaran, and Raia Hadsell. Overcoming catastrophic forgetting in neural networks. *CoRR*, abs/1612.00796, 2016.

[LQH17] Pengfei Liu, Xipeng Qiu, and Xuanjing Huang. Adversarial multi-task learning for text classification. *CoRR*, abs/1704.05742, 2017.

[MW17] Julian F. Miller and Dennis G. Wilson. A developmental artificial neural network model for solving multiple problems. In *Proceedings of the Genetic and Evolutionary Computation Conference Companion*, GECCO '17, pages 69–70, New York, NY, USA, 2017. ACM.

[Rud17] Sebastian Ruder. An overview of multi-task learning in deep neural networks. *CoRR*, abs/1706.05098, 2017.

[SM94] M. A. Styblinski and Jill R. Minick. *Two Methods for Solving Linear Equations Using Neural Networks*, pages 217–229. Springer US, Boston, MA, 1994.

[VC17] Roby Velez and Jeff Clune. Diffusion-based neuromodulation can eliminate catastrophic forgetting in simple neural networks. *CoRR*, abs/1705.07241, 2017.

[Wan92] J. Wang. Electronic realisation of recurrent neural network for solving simultaneous linear equations. *IET Electronics Letters*, 28(5):493 – 495, 1992.

[Wan93] J. Wang. Recurrent neural networks for solving linear matrix equations. *Computers & Mathematics with Applications*, 26(9):23 – 34, 1993.

[ZPH98] Guoqiang Zhang, B. Eddy Patuwo, and Michael Y. Hu. Forecasting with artificial neural networks:: The state of the art. *International Journal of Forecasting*, 14(1):35 – 62, 1998.