Uma arquitetura de aprendizagem profunda para desagregação de energia

Eduardo G. dos Santos[1], **Cristopher G. S. Freitas**[1], **André L.L. de Aquino**[1]

[1]Instituto de Computação – Universidade Federal de Alagoas (UFAL)

{egs, cristopher, alla}@laccan.ufal.br

Abstract. *Energy disaggregation intends to distinguish the power consumption of distinct devices connected to a single channel, in a non-intrusive way, from a single point of measurement. Deep learning is very promising in this field, since it already performs better when compared to previous models like FHMM (Factorial Hidden Markov Model). In this work, we used the UK-DALE database, which we pre-processed and soon after we used a neural network with convolutional layers for energy disaggregation. In this way, we achieved an improvement of 13.74% and 14.92% on average for two metrics used in comparison to previous work.*

Resumo. *Desagregação de energia busca distinguir o consumo de energia elétrica de diferentes dispositivos ligados a um único canal, de forma não intrusiva a partir de um único ponto de medição. Aprendizagem profunda tem se mostrado promissora nesse campo, visto que já apresentam melhores resultados quando comparadas à modelos anteriores que não fazem o uso como o FHMM (Factorial Hidden Markov Model). Nesse trabalho, utilizamos a base da dados UK-DALE, na qual fizemos um pré-processamento e aplicamos uma rede neural com camadas convolucionais (CNN) para desagregação de energia. Dessa forma conseguimos uma melhora de 13.74% e 14.92% em média em duas métricas usadas em comparação à trabalhos anteriores.*

1. Introdução

Monitoramento de carga não intrusivo, ou NILM (*Non-intrusive load monitoring*), é um problema de extração e identificação do consumo individual de cada dispositivo presente no ambiente, a partir de um único ponto de monitoramento, o qual fornece apenas o consumo agregado de todo o ambiente [Zoha et al. 2012]. Com isso, a *desagregação de energia* busca ajudar a entender o consumo individual de cada dispositivo no ambiente monitorado para diversas aplicações, entre elas temos a possibilidade de cobrança pelo consumo de energia por cada aparelho individualmente, oferecendo ao consumidor uma conta mais transparente. Observa-se também que com o NILM o custo de monitoramento de cada dispositivo é bem menor, visto que apenas é necessário um medidor em um único ponto, ao invés de vários medidores distribuídos pelo ambiente.

Desagregação de energia é um problema complexo, visto que diversos aparelhos demandam energia simultaneamente de diferentes formas, também existem dispositivos diferentes, mas com sinais similares; além de aparelhos trocando de estado à todo instante, presença de ruído e etc. Contudo, no decorrer dos anos a quantidade de dados nessa

área vem crescendo, e com isso, a possibilidade de aplicação de técnicas utilizando inteligência artificial e aprendizagem de máquina ganharam espaço, como modelos variantes do modelo de Markov [Parson et al. 2012, Kim et al. 2011] que utilizam aprendizagem não supervisionada, extraindo assinaturas dos dispositivos apenas a partir do sinal agregado.

Aparelhos possuem assinaturas próprias que são compostas por duração da ativação do dispositivo, quantidade de energia consumida em cada ativação e etc. [Zhang et al. 2018] mostra que a rede neural é capaz de aprender essas assinaturas sem que seja necessário inserir nenhum conhecimento de forma manual no modelo, diferente de abordagens que usam GPS (*Graph-Based Signal Processing*) e FHMM ([Tabatabaei et al. 2017]; [Zhao et al. 2015]) para desagregação nas quais é preciso inserir informações da ativação do dispositivo. Por conta disso, aprendizagem profunda se torna um ótima abordagem para desagregação de energia.

Nesse trabalho, utilizamos uma rede neural convolucional inicialmente proposta por [Zhang et al. 2018], com algumas modificações e um pré-processamento mais elaborado. Daí, conseguimos melhorar a desagregação em comparação ao estado da arte, avaliando as métricas de erro médio absoluto (MAE) e erro do sinal agregado (SAE). O restante do trabalho está organizado da seguinte forma: Na seção 2, discutimos sobre aprendizagem profunda; na seção 3, discutimos os trabalhos relacionados; na seção 4, definimos o problema formalmente; na seção 5, apresentamos o modelo proposto; na seção 6, apresentamos o experimento e os resultados obtidos; por fim, na seção 7 discutimos as conclusões obtidas e os trabalhos futuros.

2. Aprendizagem Profunda

O conceito de aprendizagem profunda é inspirado no funcionamento do cérebro, como a aprendizagem organiza-se em múltiplas camadas de forma hierárquica [Utgoff and Stracuzzi 2002]. Motivados por esse mecanismo, pesquisadores dedicaram-se à adaptar esse paradigma de aprendizagem em múltiplas camadas para uma arquitetura computacional de rede neural artificial, modelando essa abstração através de grafos e funções de ativações entre as conexões dos nós [Bengio et al. 2009].

Em geral, redes neurais artificiais podem possuir até três elementos principais: (i) a camada de entrada; (ii) uma ou mais, camadas ocultas; e (ii) a camada de saída, cada uma contendo um número variado de neurônios artificias. Nas camadas ocultas, cada neurônio de uma camada pode ser visto como um vértice conectado com neurônios da camada anterior e posterior, e cada aresta possui um peso associado para simular as diferentes intensidades de sinal que um neurônio pode receber. Durante o processo de treinamento, a rede vai ”aprendendo”de acordo com os dados e a arquitetura utilizada, nesse processo, temos duas operações críticas: *forward pass* e *backward pass* [Goodfellow et al. 2016].

O *forward pass* acontece quando os dados fluem da camada de entrada, passando através das camadas ocultas, fazendo as operações necessárias e chegando na camada de saída. No entanto, a aprendizagem acontece durante o *backward pass*, no qual após fazer o *forward pass* através de toda a rede, é computado o erro relativo da saída da rede. Quando a função de ativação é diferenciável, é usado a descida do gradiente para modificar os pesos subtraindo o valor do gradiente multiplicado pela taxa de aprendizagem, dessa maneira os pesos estão sempre mudando visando diminuir o erro. O *backward pass*

apenas acontece após uma certa quantidade de *forward pass*, a qual é definida por *batch*.

Nos últimos anos, novas arquiteturas de redes neurais foram surgindo, dando início às técnicas de aprendizagem profunda, que ganharam notoriedade resolvendo problemas de visão computacional, processamento de linguagem natural e reconhecimento de fala. Entre algumas das arquiteturas mais conhecidas estão as Redes Neurais Convolucionais (CNNs), Redes Neurais Recorrentes (RNNs), e *Autoencoders*.

3. Trabalhos relacionados

Com o avanço de aprendizagem profunda em classificação de imagens e reconhecimento de fala, redes profundas têm sido uma abordagem interessante para desagregação de energia.

[Kelly and Knottenbelt 2015a] compararam três arquiteturas de redes neurais diferentes, duas utilizando a abordagem sequência para sequência a qual a entrada é uma janela do sinal agregado e a saída é uma janela do sinal desagregado, e uma outra abordagem foi treinar a rede neural para estimar o ponto inicial onde o aparelho alvo ficou ativo, a média de consumo em sua ativação e o ponto final onde o aparelho encerra sua ativação. Para a primeira abordagem as arquiteturas utilizadas foram RNN com unidades LSTM e uma dAE com camadas convolucionais e densas, para a segunda abordagem foi utilizada uma arquitetura com camadas convolucionais e densas também.

[Mauch and Yang 2015] construíram uma arquitetura com várias camadas LSTM, desse modo criando uma rede LSTM profunda. Com essa arquitetura ele mostra que com aparelhos que possuem uma certa periodicidade a rede desempenha uma desagregação melhor.

[Zhang et al. 2018] reduziram a complexidade do dAE fazendo com que a rede neural inferisse apenas um ponto de cada janela do sinal agregado, dessa forma diminuindo a complexidade das múltiplas predições, ele definiu essa abordagem como aprendizagem sequência para ponto. No trabalho ele mostra a partir de observações nos mapas característicos aprendidos que a rede neural aprendeu assinaturas cruciais do dispositivo para a qual foi treinada.

[Sirojan et al. 2018] utilizaram a aprendizagem sequência para sequência com tamanho de janela diferente para cada dispositivo, e construiu uma rede neural com uma arquitetura VAE composta de camadas convolucionais e densas, assim obtendo resultados superiores ao trabalho de Zhang.

[Xiao and Cheng 2019] abordaram o problema de uma forma diferente, usando uma rede neural com camadas densas e de poucos parâmetros para inferir o estado operacional de cada aparelho. Compararam seus resultados com as abordagens baseadas na cadeia de Markov e GPS, e obtiveram resultados melhores.

4. Definição do problema

Seja $C[t_i]$ o consumo de todo o ambiente no instante t_i e seja X_j o conjunto que contém as leituras em cada instante de tempo de um dispositivo j, ou seja, $X_j = \{x_{ij}, \ldots, x_{Tj}\}$. O consumo agregado então, é a soma do consumo de todos os dispositivos presentes no ambiente, ou seja, $C[t_i] = \sum_{j=1}^{|A|} x_{ij} + \epsilon$, onde $x_j \in A$, tal que, A é o conjunto

de dispositivos no ambiente o qual estamos interessados, e ϵ está associado a um ruído gaussiano com média zero e desvio padrão um.

A desagregação é feita para cada intervalo de tempo, dado o consumo $C[t_i]$, deve-se inferir o consumo de cada dispositivo x_{ij} individualmente. Nessa abordagem, a rede neural recebe uma janela do sinal agregado, ou seja, $C_W = \{C[t_1], C[t_2], ..., C[t_W]]\}$, tal que W é o tamanho da janela. No nosso caso, dado C_W infere-se apenas o ponto do meio da janela do sinal desagregado, ou seja, $x_{\lceil W/2 \rceil j}$ tal que a janela do sinal desagregado correspondente é $X_j = \{x_{1j}, x_{2j}, ..., x_{Wj}\}$.

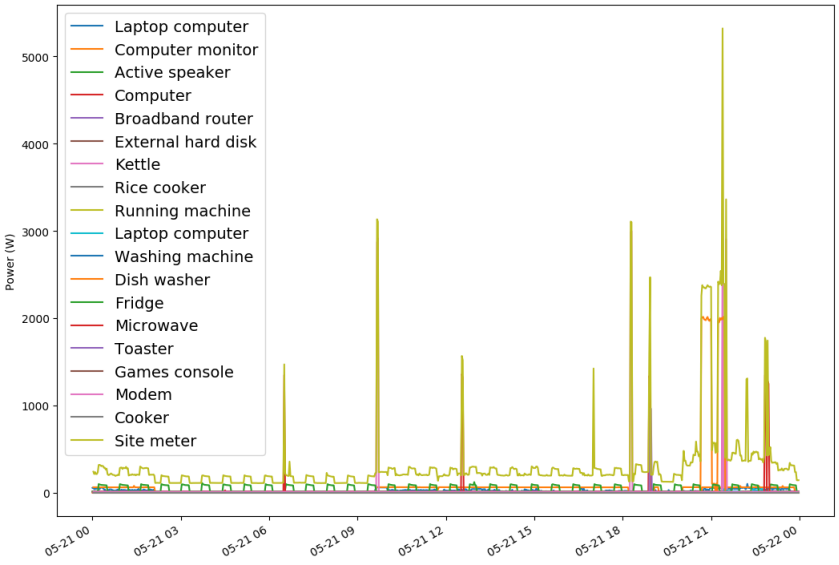

Figura 1. Exemplo de desagregação gerado usando a ferramenta NILMTK.

Na figura 1, temos um exemplo de desagregação de energia retirado da casa 2 do conjunto de dados UK-DALE [Kelly and Knottenbelt 2015b] em uma janela de tempo de 2 dias.

5. Modelo proposto

5.1. Pré-processamento

Em aprendizagem supervisionada deve-se tomar cuidado no pré-processamento dos dados, visto que qualquer erro pode ocasionar em uma aprendizagem falha. O conjunto de dados utilizado foi o UK-DALE [Kelly and Knottenbelt 2015b], o qual tem uma taxa de amostragem de $1\,Hz$ na leitura do sinal principal e $1/6\,Hz$ na leitura individual de cada aparelho, porém essa frequência pode variar um pouco por conta de perda ou atraso de pacotes na coleta dos dados como especificado em [Kelly and Knottenbelt 2015b].

No pré-processamento dos dados, o NILMTK (Non-Intrusive Load Monitoring Toolkit) [Batra et al. 2014] foi utilizado para converter o *timestamp* de cada valor em uma data legível, e com esses dados em mãos, fizemos nosso próprio algoritmo de sincronização de dados, visto que se utilizarmos as funções do NILMTK para modificar a taxa de amostragem, em outras palavras, modificar o sinal agregado de $1\,Hz$ para $1/6\,Hz$, os dados não ficarão sincronizados com o sinal desagregado, por conta das falhas mencionadas acima.

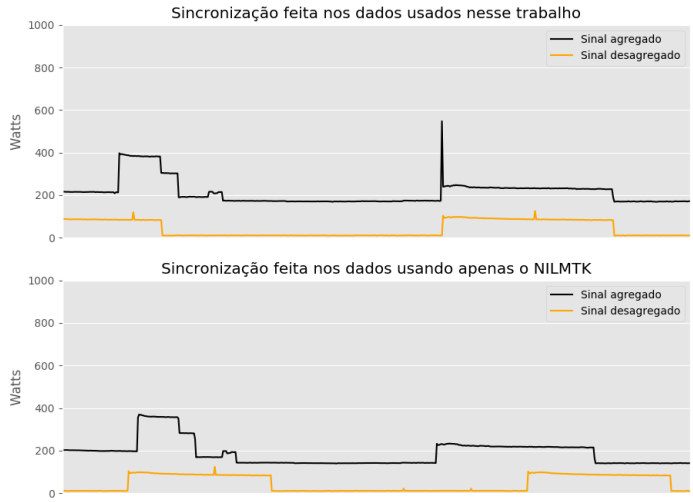

Figura 2. Comparação entre a sincronização feita nos dados usando nosso algoritmo de sincronização e apenas funções do NILMTK.

Na figura 2, pode-se ver uma sincronização feita pelo nosso algoritmo, e uma feita pelo NILMTK, apenas modificando a frequência de amostragem do sinal agregado para $1/6\,Hz$. Com os dados sincronizados em mãos, ao pegarmos uma janela de tamanho W do sinal agregado e pegar o ponto central da janela do sinal desagregado tínhamos certeza que esses valores tinham sido coletados exatamente no mesmo instante de tempo.

Normalizar os dados é importante para evitar a saturação das funções de ativações da rede. Nós normalizamos utilizando os mesmos valores de média e desvio padrão que foram utilizados no trabalho de [Zhang et al. 2018]. Vale notar que não perder a escala dos dados é importante nesse contexto, por isso não normalizamos usando a média e o desvio padrão da própria janela.

5.2. Modelo usado

Na arquitetura desenvolvida, usamos diversas camadas convolucionais 1D, i.e, unidimensional, visto que estamos tratando de séries temporais; e uma camada de neurônios totalmente conectados, como mostra a figura 3. A rede possui cinco camadas convolucionais com tamanhos de filtros e quantidade variável. Após isso, temos uma camada *flatten*, que prepara os dados das camadas convolucionais para a camada de neurônios. Na camada densa, definimos a quantidade de neurônios em 1024, que é igual ao tamanho do maior *batch* que usamos.

Como especificado em [Zhang et al. 2018], a rede aprende uma regressão não linear entre o sinal agregado e o sinal desagregado de um dispositivo, com isso, decidimos optar pela função mais usada: a função de perda de regressão que é o erro quadrático médio, ou MSE (*Mean Square Error*), que é dado por

$$MSE = \frac{1}{N}\sum_{i=1}^{N}(\hat{x}_t - x_t)^2. \tag{1}$$

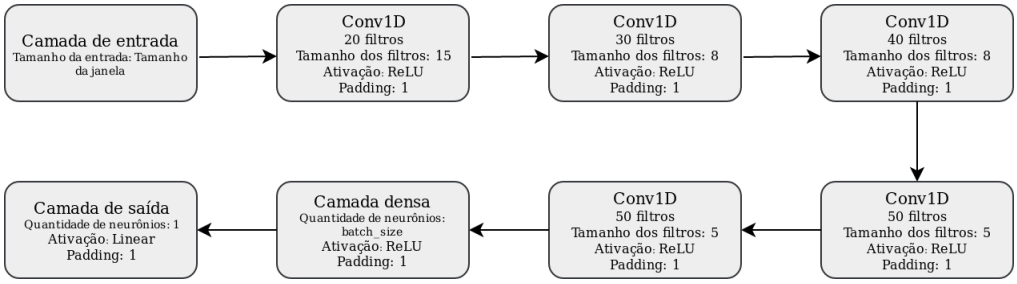

Figura 3. Arquitetura da rede neural proposta nesse trabalho.

onde $\hat{x_t}$ e x_t denotam respectivamente, o valor inferido e o valor real.

Essa função é mais sensível à *outliers* quando comparada a outras funções de perda de regressão. Nesse contexto que estamos trabalhando, os *outliers* podem representar a ativação de algum aparelho que está presente no ambiente, por conta disso, é interessante que a função de perda seja sensível à esses picos de consumo de energia.

Seja $X_{t:t-W-1}$ e $Y_{t:t-W-1}$ a janela do sinal agregado e desagregado respectivamente, W o tamanho da janela e $y = Y_k$, onde $k = \lceil W/2 \rceil$. Seja F a função que representa a rede neural. Conforme a arquitetura da rede, no treinamento a rede recebe $X_{t:t-W-1}$ como entrada e o erro é calculado através da função de perca usando y e o ponto inferido pela rede, denotado por $\hat{y}$. Dessa forma, através do *backfoward pass* os pesos são ajustados até que a rede aprenda a inferir o ponto y apenas usando $X_{t:t-W-1}$. Essa é uma aprendizagem sequência para ponto ([Zhang et al. 2018]).

Batchs (ou lotes) pequenos possuem uma dinâmica de treinamento mais rápida e possibilitam uma maior generalização dos conjuntos de dados. Pelo fato do tamanho do lote ser pequeno, isso permite que o modelo inicie a aprendizagem sem ver todo o conjunto de dados, fazendo com que o modelo venha a convergir para ótimos locais. Por outro lado, usar tamanho de lotes muito grande, apesar de garantir a convergência para um ótimo global, torna a aprendizagem muito lenta e acaba se tornando até inviável em alguns casos.

Ao começar com um tamanho de lote menor e depois aumentá-lo, é uma boa estratégia, visto que o modelo começa a aprender rápido, tendendo a convergir para ótimos locais e logo depois, nós permitimos que ele dê saltos maiores em busca do ótimo global. Nesse trabalho nós treinamos a rede com 30 épocas com lotes de tamanho 512, e 20 épocas com lotes de tamanho 1024. De antemão, nós iremos nos referir ao modelo proposto como *SConv1d*.

Para implementar a rede neural profunda, nós utilizamos a biblioteca Keras que é uma biblioteca que foi desenvolvida em python para experimentações com aprendizagem profunda e utilizamos o *tensorflow* como *backend*. O treinamento da rede foi feito em uma máquina com duas placas NVIDIA GTX 1080 Ti.

6. Resultados

Nós comparamos a nossa proposta à três diferentes modelos: (i) *AFHMM* [Kolter and Jaakkola 2012]; (ii) *CVAE* [Sirojan et al. 2018]; e (iii) *seq2point* [Zhang et al. 2018]. Na nossa proposta, utilizamos todos os dados do

conjunto de dados UK-DALE ([Kelly and Knottenbelt 2015b]).

Para validação da proposta, vamos focar em dois dispositivos distintos, uma geladeira e um lava-louças. A complexidade desses dispositivos é maior quando comparados a outros como uma chaleira ou micro-ondas pois esses dispositivos possuem múltiplos estados enquanto outros possuem apenas os estados de ligado ou desligado.

Nós usamos os dados da casa 1,3,4,5 para aprendizagem e reservamos a casa 2 apenas para testes, visto que a casa 2 contém os dois aparelhos. Observe que nesse cenário os aparelhos apresentam demanda de energia diferentes visto que são de diferentes casas, e assim a rede é treinada para se adaptar de diferentes formas. Isso também faz com que a rede neural infira o sinal desagregado a partir de um sinal agregado a qual nunca teve contato antes, simulando uma aplicação prática de NILM.

Para uma avaliação quantificada, recorremos à duas métricas: (i) erro absoluto médio (MAE); e o erro agregado de sinal (SAE). Quando estamos interessados no erro em cada instante de tempo, utilizamos o MAE, definido por

$$MAE = \frac{1}{T}\sum_{t=1}^{T}|\hat{x_t} - x_t| \tag{2}$$

onde $\hat{x_t}$ e x_t denotam respectivamente, o valor inferido e o valor real.

No entanto, ao avaliarmos o consumo total em um período de tempo, usamos o SAE, definido por

$$SAE = \frac{|\hat{r} - r|}{r} \tag{3}$$

onde $\hat{r} = \sum_t \hat{x_t}$ e $r = \sum_t x_t$ denotam o consumo inferido e o real respectivamente.

Tabela 1. MAE e SAE para o conjunto de dados UK-DALE

Métrica	Métodos	Geladeira	Lava-louças
MAE	AFHMM	42.35	199.84
	seq2point(Zhang)	20.894	27.704
	CVAE	18.1	**19.6**
	SConv1d	**12.457**	20.08
SAE	AFHMM	0.98	4.50
	seq2point(Zhang)	0.121	0.645
	CVAE	0.132	0.317
	SConv1d	**0.102**	**0.281**

Observando a tabela 1, vemos que a SConv1d inferiu o consumo da geladeira melhor, tendo um desempenho superior de 31.2% e 22.7% em relação ao MAE e SAE quando comparados ao CVAE. No lava-louças nosso modelo obteve resultados inferiores ao modelo CVAE para a métrica MAE, mas com a métrica SAE, obtivemos um resultado superior de 11.67%.

7. Conclusão e trabalhos futuros

Uma vez que cada rede for treinada, podemos as unir de modo que conseguiríamos obter o consumo individual de cada dispositivo no ambiente. Note que nós utilizaríamos a rede neural para inferir o consumo do dispositivo o qual ela foi treinada, e para isso não é

preciso muito processamento computacional. Como mencionado no trabalho do Kelly ([Kelly and Knottenbelt 2015a]), poderíamos armazenar dados de um ano de dez milhões de pessoas em treze terabytes e supondo que seja gasto um segundo para processar uma semana de dados de um usuário, para processar dados de dez milhões bastaria dezesseis GPU's.

Aqui mostramos que com pequenas alterações na arquitetura e no treinamento da rede, no pré-processamento de dados e nas ferramentas usadas podemos obter resultados superiores. Com isso, vimos que é de suma importância sempre explorar as pequenas variações nas arquiteturas e a forma de treinar a rede neural, possibilitando uma aprendizagem melhor e mais genérica.

Em trabalhos futuros pretendemos explorar outros tipos de camadas de rede neural visando em reduzir a quantidade de parâmetros, para que sua implementação em sistemas embarcados se torne mais factível, uma vez que para seu uso prático, tais sistemas serão usados. Também pretendemos construir uma rede neural que seja capaz de inferir o estado operacional de cada dispositivo presente no ambiente, e que o aumento na quantidade de aparelhos no ambiente não influencie tanto na quantidade de parâmetros da rede.

Referências

Batra, N., Kelly, J., Parson, O., Dutta, H., Knottenbelt, W., Rogers, A., Singh, A., and Srivastava, M. (2014). Nilmtk: an open source toolkit for non-intrusive load monitoring. In *Proceedings of the 5th international conference on Future energy systems*, pages 265–276. ACM.

Bengio, Y. et al. (2009). Learning deep architectures for ai. *Foundations and trends® in Machine Learning*, 2(1):1–127.

Goodfellow, I., Bengio, Y., and Courville, A. (2016). *Deep learning*. MIT press.

Kelly, J. and Knottenbelt, W. (2015a). Neural nilm: Deep neural networks applied to energy disaggregation. In *Proceedings of the 2nd ACM International Conference on Embedded Systems for Energy-Efficient Built Environments*, pages 55–64. ACM.

Kelly, J. and Knottenbelt, W. (2015b). The UK-DALE dataset, domestic appliance-level electricity demand and whole-house demand from five UK homes. *Scientific Data*, 2(150007).

Kim, H., Marwah, M., Arlitt, M., Lyon, G., and Han, J. (2011). Unsupervised disaggregation of low frequency power measurements. In *Proceedings of the 2011 SIAM international conference on data mining*, pages 747–758. SIAM.

Kolter, J. Z. and Jaakkola, T. (2012). Approximate inference in additive factorial hmms with application to energy disaggregation. In *Artificial intelligence and statistics*, pages 1472–1482.

Mauch, L. and Yang, B. (2015). A new approach for supervised power disaggregation by using a deep recurrent lstm network. In *2015 IEEE Global Conference on Signal and Information Processing (GlobalSIP)*, pages 63–67. IEEE.

Parson, O., Ghosh, S., Weal, M., and Rogers, A. (2012). Non-intrusive load monitoring using prior models of general appliance types. In *Twenty-Sixth AAAI Conference on Artificial Intelligence*.

Sirojan, T., Phung, B., and Ambikairajah, E. (2018). Deep neural network based energy disaggregation. In *2018 IEEE International Conference on Smart Energy Grid Engineering (SEGE)*, pages 73–77. IEEE.

Tabatabaei, S. M., Dick, S., and Xu, W. (2017). Toward non-intrusive load monitoring via multi-label classification. *IEEE Transactions on Smart Grid*, 8(1):26–40.

Utgoff, P. E. and Stracuzzi, D. J. (2002). Many-layered learning. *Neural Computation*, 14(10):2497–2529.

Xiao, P. and Cheng, S. (2019). Neural network for nilm based on operational state change classification. *arXiv preprint arXiv:1902.02675*.

Zhang, C., Zhong, M., Wang, Z., Goddard, N., and Sutton, C. (2018). Sequence-to-point learning with neural networks for non-intrusive load monitoring. In *Thirty-Second AAAI Conference on Artificial Intelligence*.

Zhao, B., Stankovic, L., and Stankovic, V. (2015). Blind non-intrusive appliance load monitoring using graph-based signal processing. In *2015 IEEE Global Conference on Signal and Information Processing (GlobalSIP)*, pages 68–72. IEEE.

Zoha, A., Gluhak, A., Imran, M., and Rajasegarar, S. (2012). Non-intrusive load monitoring approaches for disaggregated energy sensing: A survey. *Sensors*, 12(12):16838–16866.