Energia Aplicado Xxx (2017) xxx-xx

Listas de conteúdo disponíveis em ScienceDirect

Energia Aplicado

página inicial do www.elsevier.com/locate/apenerge

Um modelo de aprendizagem profunda gpu para previsão de séries temporânficos

Igor M. Coelho [a,b,⇑], Vitor N. Coelho [a,c,*], Eduardo J. da S. Luz [d], Luiz S. Ochi [c], Frederico G. Guimarães [e], Eyder Rios [f]

a Grupo da Causa Humana, Ouro Preto, Brasil [b] Departamento de Computação, Universidade do Estado do Rio de Janeiro, Rio de Janeiro, Instituto Brasil [c] de Computação, Universidade Federal Fluminense, Niterói, Brasil [d] Departamento de Computação, Universidade Federal de Ouro Preto, Ouro Preto, Brasil [e] Departamento de Engenharia Elétrica, Universidade Federal de Minas Gerais, Belo Horizonte, Brasil [f] Instituto de Computação, UESPI, Paranaíba, Brasil

highli ghts

- Um mecanismo de CPU-GPU é proposto para acelerar o aprendizado de séries temporâneais.
- A previsão de demanda de energia doméstica desagregada é usada como caso de estudo.
- Sugestões para incorporar o sistema proposto baseado em GPU de baixa energia em sensores inteligentes.
- Avaliação de precisão do modelo de previsão paralela com uma fase de treinamento metaheurístico.

article paranfo

Histórico do artigo:
Recebido em 30 de setembro de 2016
Recebido em formulário revisado em 2 de janeiro de 2017
Aceito 3 janeiro 2017 Disponível
online xxxx

Keywords:
Aprendizado profundo
Unidade de processamento gráfico
Modelo de previsão híbrida
Sensores inteligentes
Demanda de eletricidade doméstica
Série de tempo de big data

abstract

À medida que a nova geração de sensores inteligentes está evoluindo para sistemas de aquisições de alta amostragem, a quantidade de informações a serem tratadas por algoritmos de aprendizagem tem aumentado. A arquitetura da Unidade de Processamento Gráfico (GPU) oferece uma alternativa mais verde com baixo consumo de energia para mineração de big data, trazendo o poder de milhares de núcleos de processamento para um único chip, abrindo assim uma ampla gama de aplicações possíveis. Neste artigo (uma extensão substancial da versão curta apresentada no REM2016 em 19 a 21 de abril, Maldivas [1]), projetamos uma nova estratégia paralela para o aprendizado de séries temporais, na qual diferentes partes das séries temporais são avaliadas por diferentes tópicos. A estratégia proposta é inserida dentro do núcleo de um modelo metaheurístico híbrido, aplicado para padrões de aprendizagem a partir de um importante problema de previsão de mini/microgrid, a previsão da demanda de eletricidade doméstica. As futuras cidades inteligentes certamente contarão com a geração distribuída de energia, na qual os cidadãos devem estar cientes sobre como gerenciar e controlar seus próprios recursos. Nesse sentido, a pesquisa de desagregação de energia fará parte de várias aplicações típicas e úteis de microgrid. Os resultados computacionais mostram que a estratégia de aprendizagem proposta da GPU é escalável à medida que o number de rodadas de treinamento aumenta, emergindo como uma ferramenta promissora de aprendizagem profunda a ser incorporada em sensores inteligentes.

1. Introdução

Às vezes chamada de a máquina mais enorme já construída, a rede elétrica tem passado por várias melhorias. Pesquisadores e a indústria têm se concentrado em integrar eficientemente recursos de energia renovável (RER) na rede. A inserção maciça de RER é geralmente assistido por Artificial Algoritmos e modelos baseados em Inteligência (IA) [2], que estão sendo incorporados em Medidores Inteligentes (SM) [3]. A proposta descrita neste presente estudo é uma ferramenta potencial a ser incorporada em MS, podendo nos prevereful informações de big data desagregado série de tempo de carga. Essas séries de tempo de carga têm o potencial de auxiliar a integração rer em sistemas mini/microgrid, nos quais os usuários podem empregar dispositivos inteligentes para autogerir seus recursos e demandas.

Os SM são "inteligentes" no sentido de que o modesto uso de sensores está sendo substituído por dispositivos com muitas habilidades computacionais. Geralmente, essas habilidades computacionais são desenvolvidas com base em técnicas de IA ou estratégias específicas vislumbradas por seu criador/programador. Essa classe de medidores está começando a se comunicar

Corresponding authors at: Grupo da Causa Humana, Ouro Preto, Brazil.
E-mail addresses: igor.machado@ime.uerj.br (I.M. Coelho), vncoelho@gmail.com (V.N. Coelho).

http://dx.doi.org/10.1016/j.apenergy.2017.01.003 0306-2619/ 2017
Publicado pela Elsevier Ltd.

uns com os outros [4] e a introduzir informações importantes a serem tratadas pelos tomadores de decisão. Esses sensores baseados em software são cruciais para o processo de tomada de decisão sobre esses cenários cheios de indeterminações.

Entre as técnicas de IA encontradas na literatura, as baseadas em aprendizagem profunda estão em evidência. O aprendizado profundo tem sido aplicado a vários problemas de classificação e regressão. Parte de seu sucesso se deve à extração automática de recursos em diferentes níveis de abstração. A extração automática de recursos promove a fácil reutilização de modelos em diferentes domínios sem uma intervenção humana especialista em campos. Além disso, o deep learning permite a representação das não linearidades, muitas vezes associadas a dados do mundo real complex. Modelos de aprendizagem profunda têm sido usados para alcançar resultados de última geração no campo da visão computacional [5,6] e também foram aplicados ao problema da previsão de séries temporais [7-10].

Abordagens populares de deep learning são baseadas em redes convolutionais [5], máquinas boltzman restritas (redes de crenças profundas) [11] e autoencodificadores profundos [12]. No entanto, esses métodos são muitas vezes difíceis de interpretar e reproduzir. De acordo com Hu et al. [13] vários autores tratam arquiteturas de rede profunda como uma caixa preta. Outra limitação dos métodos populares de aprendizagem profunda é o alto consumo de memória [14]. Em contrapartida, o método proposto neste trabalho é de fácil interpretação e tem baixo consumo de memória, o que significa uma vantagem competitiva sobre os métodos popular de aprendizagem profunda.

Coelho et al. [15] introduziram recentemente um Modelo de Previsão Híbrida (HFM), que calibra suas regras difusas usando procedimentos baseados em metaheurística. Sem aplicar qualquer filtro ou pré-processamento de séries tempo de consumo de energia, o modelo HFM mostrou-se competitivo com outras abordagens especializadas da literatura e facilmente generalizado para a realização de n-passos-ahead forecast.

A extensão proposta aqui (uma extensão substancial da versão curta apresentada no REM2016 em 19 a 21 de abril, Maldivas [1]) explora as capacidades de aprendizagem da ferramenta HFM, na qual a extração de recursos é feita por Estruturas de Vizinhança (NS). Fig. 1 detalha uma versão generalizada de como funciona o modelo proposto. Neste trabalho atual, considera-se apenas a camada 2, na qual o operador especial retorna os valores médios de todas as funções ativas, ou seja, "ativações'. NS são usados para calibrar cada parâmetro de cada função de ativação: entrada de lag para o operador backshift, posição de regra e peso da aplicação. Além disso, o algoritmo de calibração metaheurística é capaz de regular o tamanho da camada, adicionando ou removendo funções.

Motivados pela nova classe de séries temporais de big data, que são realidade em diversas áreas (como na indústria de energia, biologia, neurociência, processamento de imagem, entre outras), decidimos aprimorar o modelo de HFM com uma nova estomes de avaliação de previsão paralela. Em particular, neste trabalho atual, a Unidade de Processamento Gráfico (GPU) foi projetada para ser usada para prever diferentes partes de uma série de tempo de carga microgrid. O uso de arquiteturas baseadas em GPU pode fornecer uma alternativa mais verde com consumption de baixa energia para obter informações de mineração de tais enormes conjuntos de dados [16]. Cada GPU fornece milhares de núcleos de processamento com operações aritméticas muito mais rápidas do que uma clássica Unidade Central de Processamento (CPU). Em poucas palavras, nosso objetivo é gerar o processo de aprendizado de roscas de GPU de conjunto, que fornecem previsões independentes, otimizadas a fim de reduzir uma determinada medida de qualidade estatística. A GPU parece se encaixar no escopo do HFM, uma vez que o modelo pode ser implementado e adaptado à computação de GPU, especialmente porque o método usa algoritmos metaheurísticos e foi implementado no núcleo do OptFrame [17]. O processo de calibração de parâmetros automáticos do HFM também corresponde aos requisitos da série de tempo de big data, principalmente devido à sua fase de aprendizagem baseada em metaheurística. Para isso, a NS desempenha um papel vital na calibração do modelo e na busca de soluções mais eficientes. Associada ao poder e flexibilidade da metaheurística, a ausência de parâmetros que ajustam simplificam a aplicação do quadro proposto a diferentes tempos, em particular, quando é necessária uma previsão n-passo-ahead.

Este artigo considera um problema de previsão de mini/microgrid como caso de estudo, a Previsão de Demanda de Energia Elétrica Doméstica Desagregada. Os pesquisadores começaram a divulgar publicamente seus conjuntos de dados, como o Conjunto de Dados de Desagregação de Energia de Referência (REDD) [18], que fornece medições de energia de baixa frequência (intervalos de 3 a 4 s) disponíveis para circuitos monitorados individualmente de 10 a 25. A previsão da demanda de eletricidade doméstica tem grande potencial para aplicações de microgrid, como o projeto de edifícios verdes e casas [19]. Prever diferentes séries temporizadas de uma casa abre uma ampla gama de possibilidades para uma integração RER eficiente. Considerando que bilhões de dólares estão sendo gastos para instalar O SM [20], os pesquisadores estão defendendo que os dados do nível do aparelho possam promover inúmeros benefícios.

No restante deste artigo introduzimos a arquitetura da GPU em detalhes (Seção 2) e o processo de previsão desagregado da GPU (Seção 3). Os resultados computacionais the e os parâmetros analisados são apresentados na Seção 4 e, finalmente, a Seção 5 desenha algumas considerações finais.

2. Arquitetura de GPU

A GPU foi originalmente projetada para aplicações gráficas (recebendo assim o nome de uma Unidade de Processing Gráfico) de modo que qualquer algoritmo não gráfico projetado para GPU teve que ser escrito em termos de APIs gráficas como OpenGL. Isso permitiu o desenvolvimento de aplicações escaláveis para problemas computacionais caros, como simulações de colisão na fisioterapia [21]. O modelo de programação de GPU evoluiu para a moderna GPU (GpGPU), com ferramentas mais fáceis de usar e maduras para o desenvolvimento de aplicativos, como o CUDA, uma extensão de linguagem C++ proprietária da NVIDIA, uma das principais fabricantes de GPU [16].

A arquitetura gpu é organizada como uma matriz de multiprocessadores de streaming altamente roscados, cada um contendo uma série de unidades de processamento (núcleos), além de uma estrutura de memória multiníníveis. A configuração do hardware de GPU depende da capacidade de computação, um parâmetro relacionado à microarquia GPU que define quais recursos de hardware estarão disponíveis para o desenvolvimento do CUDA. Um programa CUDA consiste em uma ou mais fases que são executadas em CPU ou GPU. O código gpu é implementado como functions C++ conhecidos como kernels, que são lançados pelo código da CPU em uma grade de computação, geralmente formada por um grande número de segmentos que executam o mesmo kernel com o objetivo de explorar o paralelismo de dados. Os fios em uma grade são organizados em uma hierarquia de dois níveis, onde o primeiro nível é organizado como uma matriz tridimensional de blocos, cada um contendo até 1.024 threads. No segundo nível, cada bloco também é organizado de forma tridimensional. As dimensões de uma grade são projetadas pelo programador e devem observar as limits determinadas pela capacidade de computação do hardware.

Depois que um kernel é lançado, cada bloco é atribuído a um único multiprocessador de streaming, que executa os segmentos de um bloco em grupos chamados dobras. Cada dobra é processada de acordo com o modelo SIMD (Instrução Única, Dados Múltiplos), o que significa que todos os segmentos em uma dobra executam a mesma instrução a qualquer momento. Acessos de memória globais ou operações aritméticas realizadas por algum segmento também afetam a execução de dobra, forçando todos os fios na mesma dobra a esperar até que a operação seja concluída. Para ocultar a latência relacionada a essas operações, a GPU agenda outra dobra para manter a SM ocupada. Isso é possível porque a GPU é capaz de lidar com mais threads por SM do que os núcleos disponíveis.

Por favor, cite este artigo na imprensa como: Coelho IM et al. Uma GPU de aprendizado profundo metaheurística baseada modelo para tempo serprevisão de ioes. Appl Energy (2017), http:// dx.doi.org/10.1016/j.apenergy.2017.01.003

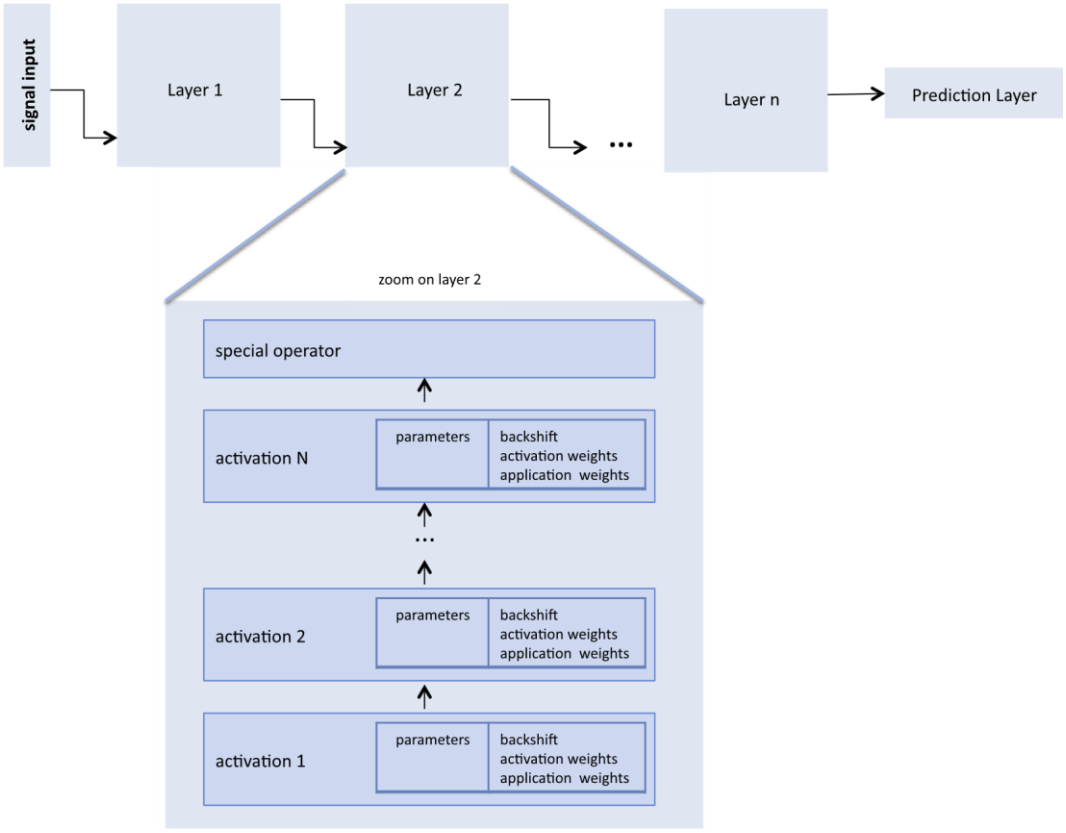


Fig. 1. Modelo de aprendizagem profunda hfm.

A comunicação entre a CPU e a GPU é feita através da memória global da GPU. No entanto, a transferência de dados de/para esse recurso geralmente é um gargalo para o desempenho do aplicativo. Portanto, a quantidade de cópias de dados entre CPU e GPU deve ser minimizada para implementação eficiente, e os outros níveis da hierarquia de memória gpu devem ser explorados. A hierarquia de memória é uma das características mais distintas das GPUs atuais. Ele inclui a memória global, uma memória de registro local para cada thread e uma memória compartilhada para cada bloco de threads. Este último tipo de memória é geralmente de grande importância para alcançar o alto desempenho na programação de GPU, atribuindo tarefas semelhantes a threads no mesmo bloco de GPU. Devido ao baixo consumo de energia das GPUs e à alta potência de processamento, a integração entre CPUs e GPUs pode ser explorada a fim de melhorar os algoritmos de previsão.

3. HFM com GPU processo de previsão desagregado

Consideremos uma série de tempo alvo ts 1/4 y1;... ; yt, compreendendo um conjunto de observações t. O objetivo é estimar/prever uma sequência finita fy^tþ1;... ; y^tþkg, com k indicando o k-passos-ahead a ser previsto, ou seja, o horizonte de previsão.

A ideia explorada em nossa abordagem conta com a independência dos treinamentos, que são então realizados por diferentes fios de GPU. Diferentes rodadas de treinamento podem ser vistas como janelas independentes, que são conhecidas por serem um aplicativo interessante para projetar hardware de alto desempenho com computação paralela [22]. Na verdade, essa applicação é especialmente adequada ao paradigma SIMD de computação de GPU [23]. Os fios em uma dobra gpu executam o mesmo instrução simultaneamente, mas sobre diferentes partes do conjunto de treinamento. Com este processo de previsão desagregado, é possível reduzir drasticamente o esforço de computação compartilhando-o com milhares de unidades de processamento independentes em uma única GPU.

Fig. 2 retrata um exemplo de uma série temporal dividida durante o processo de avaliação de desempenho do modelo, realizada por n diferentes segmentos. Devido à natureza da série de tempo de carga tratada neste trabalho atual, ela só depende de dados passados (em verde[1]) para prever os próximos valores. Nesse sentido, cada segmento pode lidar com uma rodada de previsão ao longo de uma parte da série tempo completa (uma analogia ao bag de problemas de tarefas), gerando, como saída (em azul), os valores previstos para essa parte específica, armazenados na memória global da GPU. Formalmente, em uma única etapa, cada segmento executa um processo de previsão independente, retornando seus valores precedidas para um determinado horizonte de previsão k-passos-ahead (uma sequência finita de fy^$_{tthreadi}$ þ1;... ; y^$_{tthreadi}$ þkg valores previstos, com $_{tthreadi}$ 2 1/21;...; t k).

Na estratégia de janela deslizante [24], as duas caixas juntas devem se mover ao longo de toda a série temporal. Finalmente, a CPU concatena o conjunto de valores previstos retornados por cada segmento. Seguindo essa estratégia, a CPU deve apenas dedicar seus esforços na realização da avaliação de precisão das previsões retornadas. Para a família de séries temporidas aqui, o consumo de energia das famílias desagregado, o HFM agora terá uma estratégia paralela para avaliar sua precisão, que verificará a qualidade das soluções vizinhas mais rapidamente.

Em particular, o HFM requer um número mínimo de amostras $_{tsmaxLag}$ para alimentar seu modelo. Este parâmetro guia o intervalo em que o NS pode calibrar os operadores retrolocuis. O parâmetro $_{tsmaxLag}$ limita a defasagem mais antiga a ser usada pelo modelo de previsão (representado pelo comprimento das caixas verdes).

[1] Para interpretação de cor na Fig. 2, o leitor é encaminhado para a versão web deste artigo.

Por favor, cite este artigo na imprensa como: Coelho IM et al. Uma GPU de aprendizado profundo metaheurística baseada modelo para previsão de séries tempores. Appl Energy (2017), http:// dx.doi.org/10.1016/j.apenergy.2017.01.003

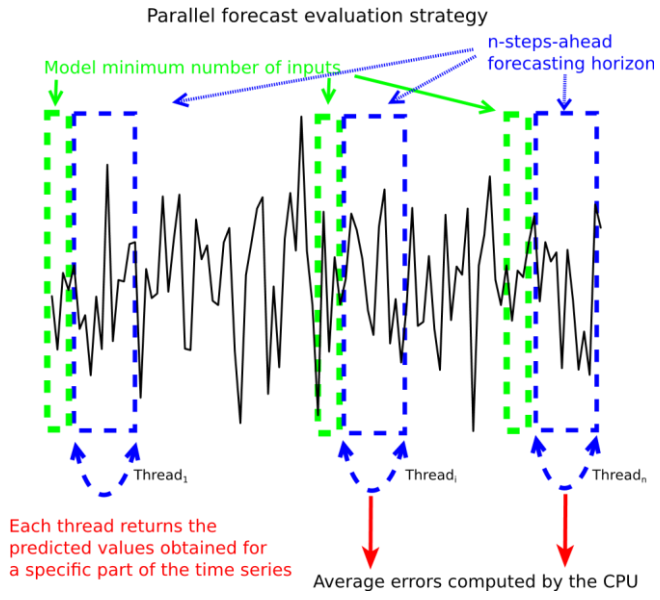


Fig. 2. HFM com um processo de previsão de GPU paralela de janela deslizante.

4. Experimentos computacionais

O REDD considerado neste estudo fornece dados de baixa frequência para a previsão da demanda de eletricidade doméstica. A análise estacética de modelos de previsão bem conhecidos sobre este conjunto de dados foi feita por Veit et al. [25]. Seguindo sua descrição, extraímos os dados da casa 1 de Apr 18th 2011 22:00:00 GMT a 2 de maio de 2011 21:59:00 GMT.

Quatro séries vezes diferentes são analisedas aqui: três circuitos monitorados individualmente diferentes; e uma das entradas principais de duas fases. Como a série temporal tem várias lacunas, buracos/quebras devido a medidores ou sensores que não fornecem medições, eles foram interpolados. Cada component casa individual passou por interpolações lineares a fim de ter uma granularidade de 1 s para cada série temporal analisada. Os testes foram realizados em um computador Intel i7-4790 K 4.00 GHz, 16 GB de RAM, equipado com uma GPU NVIDIA GTX TITAN X, 12 GB de memória global, 3072 núcleos CUDA, em conformidade com a microarquitetura Maxwell.

4.1. Análise de aceleração

Foram analisadas diferentes partes da série temporal analisada, sendo compostas por um número total de amostras a seguir chamadas nSamples: 5.000; 10.000; 50.000; 100.000; 200.000; 500.000 and 1.000.000. A capacidade da proposta na gestão de diferentes horizontes de previsão também foi analisada para os seguintes valores de passos à frente k (s): 1, 2, 5, 10, 60, 360, 720, 1800, 3600 e 7200. Assim, o horizonte de previsão máxima analisado foi de 2h, equivalente a 7200 s.

Após este design, o modelo HFM foi alimentado com amostras nSamples þ k, permitindo que ele realizasse pelo menos uma rodada de treinamento para cada configuração. Para cada configuração, foram realizadas 1000 avaliações de funções durante a fase de treinamento baseado em metaheurístico do HFM. Fig. 3 retrata um gráfico de interação sobreposta, gerado usando o pacote Ggplot2 R. O tempo foi medido com precisão de milissegundo (ms).

Para comparar a CPU pura e a estratégia de CPU-GPU paralela, contamos com o conceito de aceleração, onde o tempo gasto pelo algoritmo sequencial da CPU é divided pelo tempo gasto pelo paralelo (para executar a mesma tarefa). Analisando fig. 3, pode-se concluir que, à medida que o número de amostras aumenta, a GPU supera o desempenho da CPU com uma aceleração média de quase 15 vezes. Por outro lado, também podemos concluir que o processo de previsão desagregado não era adequado para séries temporais de pequeno porte, quando o ganho de GPU não compensa a sobrecarga inicial e a falta de trabalho para os fios de GPU.

Também estudamos as acelerações de uma forma diferente, relacionando o crescimento de aceleração com o Número de Rodadas de Treinamento, ou seja, NTR, relacionado a cada configuração. Basicamente, o NTR, também conhecido como Número de Ciclos de Treinamento, seguindo uma estratégia de janela deslizante k passo à frente, é igual ao número de amostras divididas por the forecasting horizon, como descrito em Eq. (1). O NTR é igual ao número de tarefas lançadas pela GPU.

$$tsNTR\ 1/4\ \frac{tsnSamples}{tsmaxLaged1Th}\ tsk$$

Assim, calculamos o NTR para cada configuração, arredondamos-no e cortamos nos seguintes intervalos: 0; 200; 500; 1.000; 10.000; 50.000; 100.000; 500.000 e 1.000.000. Como se pode notar, os dois últimos intervalos mais elevados só foram executados para prever horizontes com passos à frente iguais a 1, 2, 5, 10 ou 60.

Fig. 4 mostra a aceleração em relação ao NTR, considerando diferentes horizontes de previsão. Considerando os ganhos em diferentes rodadas de treinamento é possível verificar um aumento da velocidade. Mesmo para a previsão de longo prazo, a quantidade de memória global acessada em GPU não reduziu consideravelmente a velocidade. Na verdade, a GPU parece ser capaz de produzir maiores acelerações de 10 a 60 passos à frente. Quando o método realiza previsões de longo prazo, a alta quantidade de trabalho feito por cada fio de GPU implica em velocidades mais baixas.

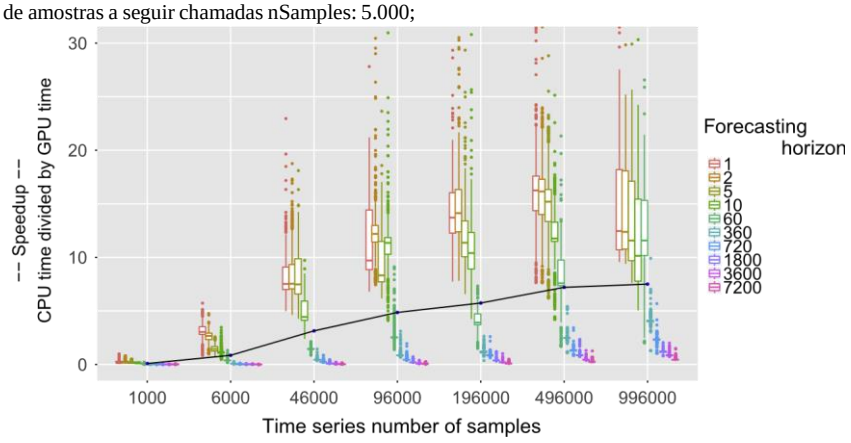


Fig. 3. Aceleração paralela do modelo de forecasting de acordo com o número de amostras.

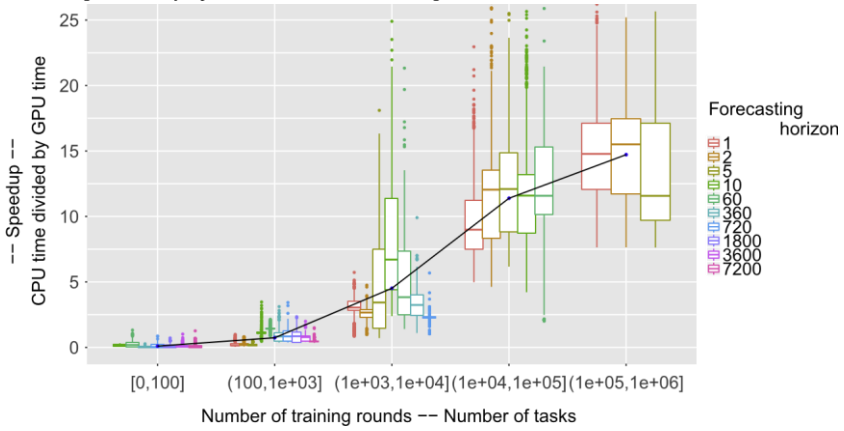


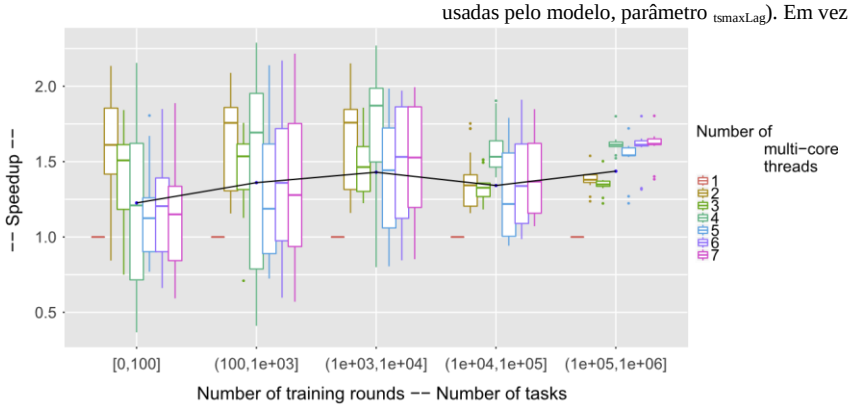
Fig. 4. Aceleração de acordo com o número de tarefas.

Além disso, verificamos o desempenho da estratégia de avaliação paralela proposta em uma arquitetura multi-CPU. Os experimentos foram executados usando um Intel Core i7-7500U, 2,70 GHz, com o OpenMP de g+ + 5.4. O lote de experimentos foi projetado para verificar a aceleração com 1 thread a 7 threads de CPU. Analogamente aos experimentos anteriores, analisamos o mesmo conjunto de valores para k e nSamples. O melhor desempenho do modelo foi verificado quando 4 threads estavam rodando em paralelo. Neste caso, a aceleração média foi de cerca de 180%, bastante baixa em comparação com a aceleração da estratégia de GPU proposta. No entanto, acreditamos que investir nesse tipo de arquitetura multi-CPU também é promissor, em particular, devido ao seu speedup mais uniforme, como pode ser verificado em ambas as parcelas depicted em Fig. 5.

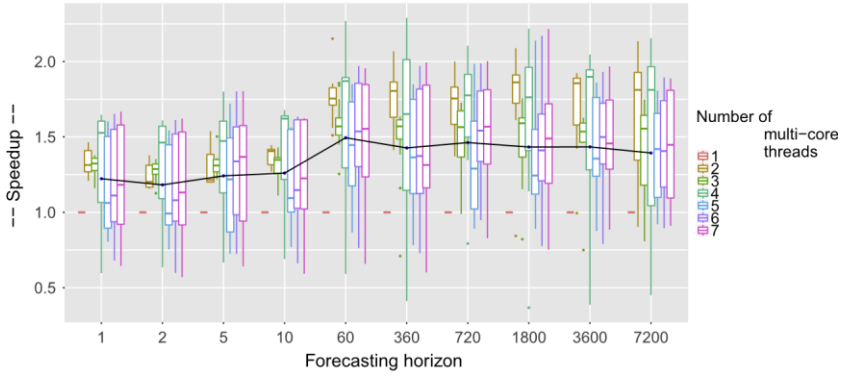
4.2. Resultados do conjunto de dados REDD com altas granularidades

Nesta seção, relatamos alguns resultados iniciais com fases de treinamento de 1 s do modelo paralelo de HFM proposto, seguindo um lote semelhante de experimentos relatados por Veit et al. [25]. Como feito por eles, verificamos o desempenho do nosso modelo em relação a 18 configurações difusas, com granularidades de 15, 30 e 60 min, cobrindo sete horizontes de previsão diferentes de 1440, 720, 360, 180, 60, 30 e 15 min.

A estratégia de janela deslizante foi utilizada para treinamento iterativo e teste do modelo. Assim, o conjunto de dados was dividido em janelas com comprimentos definidos (3 dias + 3 dias como possíveis entradas a serem usadas pelo modelo, parâmetro $_{tsmaxLag}$). Em vez de definir o deslizamento



(a) Speedup according to the number of tasks



(b) Speedup for different forecasting horizons

Fig. 5. Aceleração de acordo com o número de threads de CPU.

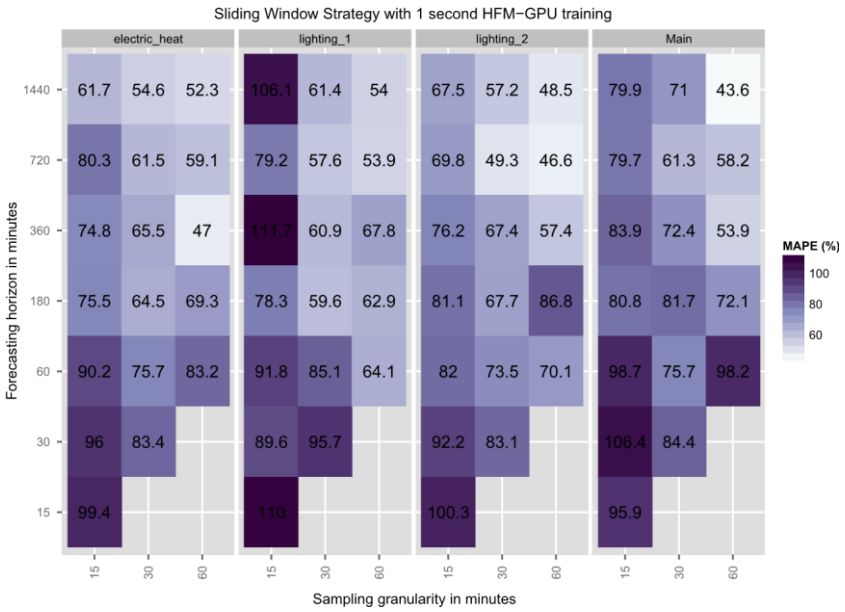


Fig. 6. MAPE para previsão n-passos-ahead e diferentes granularidades com uma estratégia de janela deslizante.

comprimento até 24 horas, exploramos toda a série temporal, movendo as janelas deslizantes com uma única unidade (15, 30 ou 60 min).

Fig. 6 mostra mapas de calor do Erro Percentual de Absolute Médio (MAPE). Esses resultados iniciais sugerem a competitividade da nossa proposta com os resultados relatados na literatura. Em particular, destacamos nosso sólido desempenho em horizontes de previsão superiores a 60, principalmente devido à estratégia metaheurística de auto-calibração utilizada pelo nosso quadro proposto. Uma vez que a estratégia de avaliação de previsão paralela proposta não interfere na qualidade do modelo de previsão, a convergência do modelo paralelo de HFM proposto segue o modelo de HFM puro validado.

Incorporar a estrutura proposta dentro de SM e dispositivos diários [26], usando supercomputadores pessoais com aceleradores como FPGA [27], GPUs, ARM, arquitetura Intel MIC, permitiria uma tomada de decisão mais precisa em tempo real. Mais para aplicações energéticas, como a incorporação da ferramenta proposta em SM, as GPUs têm sido defendidas como a alternativa verde, com baixo custo por computação. Mais especificamente, estudos futuros devem projetar novas estratégias para limitar, ou auto-calibração, a defasagem máxima mais antiga a ser usada pelo modelo HFM. Quando o modelo solicita insumos de atrasos recentes, valores que tinham sido previstos devem ser feedback para o modelo, o que gera uma computação altamente dependente (sobre valores anteriores) que pode não ser facilmente explorada por paradigmas paralelos. Explorar as vantagens e desvantagens dessas entradas em uma série de tempo paralela n-passo-ahead prevê sons um tópico que vale a pena ser estudado.

Por fim, sugerimos a aplicação do quadro proposto para a desativação de outras importantes séries de big data do setor energético, como radiação solar, previsão de chuvas e velocidade do vento, ou mesmo outros tipos de séries temporentais, como para a Eletroencefalografia e a magnetoencefalografia e a classificação.

5. Conclusões

Neste artigo, uma estratégia baseada em multithread foi projetada para verificar o desempenho de um modelo de previsão de séries temporais baseadas em n-stepsahead metaheurística. O principal núcleo da nossa estratégia foi dividir a série temporal em parts de treinamento independentes. Como caso de estudo, foram utilizados dados de um importante problema de previsão de energia de microgrid.

A estratégia de aprendizado profundo proposta pela GPU parece ser escalável à medida que o número de rodadas de treinamento de séries tempopacos aumenta, alcançando até 45 vezes a aceleração sobre uma única implementação de CPU roscada e, em média, cerca de 15 vezes. Sendo extensível para a previsão nsteps-ahead, verificamos que o aumento do horizonte de previsão, em um certo nível, pode diminuir o desempenho total de aceleração, uma vez que mais trabalho é efetivamente atribuído a cada um dos milhares de núcleos de processamento em uma GPU.

Explorando o fato de que as técnicas de aprendizagem de machine podem ser usadas para quebrar dados de consumo de energia das famílias em aparelhos individuais, exploramos nossa proposta realizando previsão em um conjunto de dados de desagregação de energia pública. Nesse sentido, usamos esse tipo de problema de forecasting de séries temporências como exemplo esperando que o uso de algoritmos eficientes de IA, em conjunto com medidores inteligentes, seja uma solução econômica e escalável para o surgimento de casas inteligentes. Os resultados obtidos sugeriram que a proposta poderia ser aplicada na nova geração de sensores baseados em software mini/microgrid , uma vez que mostrou um desempenho razoável para prever o consumo de energia de diferentes componentes domésticos.

Reconhecimento

Vitor N. Coelho agradece às agências brasileiras de fomento capes e ao PPGEE/UFMG, por patrocinar a viagem à REM 2016, e à FAPERJ (dentro do escopo do projeto "Autonomia em redes inteligentes assistidas por ferramentas da inteligência computacional"), por apoiar e incentivar o desenvolvimento deste estudo. Frederico G. Guimarães contou com o apoio das agências brasileiras CNPq (312276/2013-3 e 306850/2016-8) e FAPEMIG. Luiz S. Ochi foi apoiado pela FAPERJ e CNPq (301593/2013-2) e Igor M. Coelho pela FAPERJ.

Os autores gostariam de agradecer aos revisores anônimos por seus valiosos comentários e feedbacks, bem como as cadeiras de sessão REM 2016, por nos motivarem a estender o trabalho apresentado.

Referências

[1] Um modelo híbrido de previsão de aprendizagem profunda utilizando avaliações de funções desagregadas gpu aplicadas para previsão de demanda de electricidade doméstica, Energy Procedia 2016;103C:280-285. http://dx.doi.org/10.1016/j.egypro.2016.11.286.

[2] Kow KW, Wong YW, Rajkumar RK, Rajkumar RK. Uma revisão sobre o desempenho da inteligência artificial e o método convencional na mitigação de eventos relacionados à qualidade de energia relacionada à rede PV. Renovar o Sust Energy Rev 2016;56:334-46. http:// dx.doi.org/10.1016/j.rser.2015.11.064.

[3] McHenry MP. Considerações técnicas e de governança para infraestrutura avançada de medição/medidores inteligentes: tecnologia, segurança, incerteza, custos, benefícios e riscos. Política Energética 2013;59:834-42. http://dx.doi. org/10.1016/j. enpol.2013.04.048.

[4] Bertoldo R, Poumadère M, Rodrigues Jr LC. Quando os medidores começam a falar: o encontro do público com medidores inteligentes na França. Energy Res Soc Sci 2015;9:146-56. http://dx.doi.org/10.1016/j.erss.2015.08.014 [Especial Edição sobre Smart Grids e as Ciências Sociais].

[5] LeCun Y, Bengio Y, Hinton G. Deep learning. Natureza 2015;521(7553):436-44.

[6] Sermanet P, Eigen D, Zhang X, Mathieu M, Fergus R, LeCun Y. Overfeat: reconhecimento integrado, localização e detecção utilizando redes convolucionais. In: Conferência Internacional sobre Representações de Aprendizagem (ICLR 2014); 2014. arXiv pré-impressão arXiv:1312.6229.

[7] Mirowski PW, LeCun Y, Madhavan D, Kuzniecky R. Comparando redes svm e convolucionais para previsão de convulsão epiléptica de eg intracraniano. In: 2008 oficina iEEE sobre aprendizado de máquina para processamento de sinais. IEEE; 2008. p. 244-9.

[8] Mohamed A-r, Dahl GE, Hinton G. Modelagem acústica usando redes de crenças profundas. IEEE Trans Audio Speech Language Proc 2012;20(1):14-22.

[9] Ngiam J, Khosla A, Kim M, Nam J, Lee H, Ng AY. Aprendizado profundo multimodal. In: Proceedings of the 28th international conference on machine learning (ICML11). p. 689-96.

[10] Lv Y, Duan Y, Kang W, Li Z, Wang F-Y. Previsão de fluxo de tráfego com big data: uma abordagem de aprendizagem profunda. IEEE Trans Intell Transp Syst 2015;16(2):865-73.

[11] Hinton GE, Osindero S, Teh Y-W. Um algoritmo de aprendizagem rápida para redes de crenças profundas. Neural Comput 2006;18(7):1527-54.

[12] Bengio Y, Yao L, Alain G, Vincent P. Generalizou denoising auto-codificadores como modelos generativos. In: Avanços em sistemas de processamento de informações neurais. p. 899-907.

[13] Hu G, Yang Y, Yi D, Kittler J, Christmas W, Li SZ, et al. Quando o reconhecimento facial se encontra com o aprendizado profundo: uma avaliação das redes neurais convolucionais para reconhecimento facial. In: Proceedings of the IEEE international conference on computer vision workshops. p. 142-50.

[14] Rhu M, Gimelshein N, Clemons J, Zulfiqar A, Keckler SW. Virtualizando redes neurais profundas para rede neural eficiente em memória. arXiv pré-impressão arXiv:1602.08124.

[15] Coelho VN, Coelho IM, Coelho BN, Reis AJ, Enayatifar R, Souza MJ, et al. Um modelo auto-arrombável evolutivo para problemas de previsão de carga no ambiente de smart grid. Appl Energy 2016;169:567-84. http://dx.doi.org/10.1016/j. apenergy.2016.02.045.

[16] Kirk DB, Wen-mei WH. Programação massivamente paralela processadores: a handson approach. 2nd ed. Morgan Kaufman; 2012.

[17] Coelho IM, Munhoz PLA, Haddad MN, Coelho VN, Silva MM, Souza MJF, et al. Um quadro computacional para problemas de otimização combinatória. In: VII Workshop ALIO/EURO sobre otimização combinatória aplicada, Porto. p. 51-4.

[18] Kolter JZ, Johnson MJ. Redd: um conjunto de dados públicos para pesquisa de desagregação de energia . In: Workshop on Data Mining Applications in Sustainability (SIGKDD), San Diego, CA.

[19] Pipattanasomporn M, Kuzlu M, Rahman S. Um algoritmo para análise inteligente de energia doméstica e resposta à demanda. IEEE Trans Smart Grid 2012;3(4):2166-73. http://dx.doi.org/10.1109/TSG.2012.2201182.

[20] Armel KC, Gupta A, Shrimali G, Albert A. A desagregação é o Santo Graal da eficiência energética? O caso da eletricidade. Política Energética 2013;52:213-34. http://dx.doi.org/10.1016/j.enpol.2012.08.062 [seção especial: Caminhos de Transição para uma Economia de Baixo Carbono].

[21] Zou Y, Zhou X, Ding G, He Y, Jia M. Um algoritmo de detecção de colisão baseado em GPGPU. In: ICIG '09. Quinta conferência internacional sobre imagem e gráficos, 2009. p. 938-42. http://dx.doi.org/10.1109/ICIG.2009.127.

[22] Nedjah N, Macedo Mourelle L. Hardware de alto desempenho do método de janela deslizante para computação paralela de exponenciações modulares. Programação Paralela Int J 2009;37(6):537-55. http://dx.doi.org/10.1007/s10766 009-0108-7.

[23] Flynn MJ. Algumas organizações de computador e sua effectiveness. IEEE Trans Comput 1972; C-21(9):948-60.

[24] Fowers J, Brown G, Cooke P, Stitt G. Uma comparação de desempenho e energizar de FPGAs, GPUs e multicores para aplicações de janela deslizante. In: Proceedings of the ACM/SIGDA international symposium on field programable gate arrays. FPGA '12. Nova Iorque, NY, EUA: ACM; 2012. p. 47-56. http://dx.doi.org/ 10.1145/2145694.2145704.

[25] Veit A, Goebel C, Tidke R, Doblander C, Jacobsen H-A. Previsão da demanda de eletricidade doméstica: benchmarking métodos de última geração. In: Proceedings of the 5th international conference on future energy systems. e-Energy '14. Nova Iorque, NY, EUA: ACM; 2014. p. 233-4. http://dx.doi.org/10.1145/ 2602044.2602082.

[26] Gradl S, Kugler P, Lohmüller C, Eskofier B. Monitoramento de ecg em tempo real e detecção de arritmia usando despejos móveis baseados em android. In: 2012 conferência internacional anual da engenharia IEEE em medicina e biologia.

[27] Qiu J, Wang J, Yao S, Guo K, Li B, Zhou E, et al. Vai mais fundo com plataforma FPGA incorporada para rede neural convolucional. In: Proceedings of the 2016 ACM/SIGDA international symposium on field-programable gate arrays. ACM; 2016. p. 26-35.

Por favor, cite este artigo na imprensa como: Coelho IM et al. Uma GPU de aprendizado profundo metaheurística baseada modelo para previsão de séries tempores. Appl Energy (2017), http:// dx.doi.org/10.1016/j.apenergy.2017.01.003