UNIVERSIDADE da MADEIRA

FACULDADE DE CIÊNCIAS EXATAS E ENGENHARIA

PROJETO / ESTÁGIO

2022/2023

Empirical exploration of performance metrics for power and energy estimation using NILM

Relatório Final do Projeto / Estágio

Orientador: Lucas Pereira

Orientando: Pedro Jaques

Índice

Índice de Tabelas

Índice de Figuras

Abstrato

O projeto a ser discutido ao longo deste relatório foi elaborado em conjunto com o ITI, LARSyS, Técnico Lisboa, tendo como orientador o professor Lucas Pereira e como orientador na Universidade da Madeira o professor Filipe Quintal.

A proposta presente, foi o desenvolvimento de uma análise empírica sobre métricas de desempenho aplicadas à estimação de energia e potência de NILM, e como as mesmas podem estar ou não relacionadas com o aparelho cuja energia está a ser estimada.

NILM significa Non-Intrusive Load Monitoring (Monitorização não intrusiva da carga). É uma tecnologia e uma metodologia utilizadas para desagregar e identificar o consumo de energia de aparelhos elétricos individuais num agregado familiar ou num edifício comercial, analisando os dados globais de consumo de energia. O NILM fornece informações sobre os padrões de utilização de energia de diferentes aparelhos e ajuda os utilizadores a compreender a quantidade de energia que cada aparelho está a consumir.

O desenvolvimento deste projeto passou por diversas fases:

1. Investigação e aquisição de conhecimento através da leitura de artigos e pesquisa de conceitos pela internet;
2. Preparação do ambiente de desenvolvimento em Python com o NILMTK e todos dados necessários para efetuar a análise;
3. Aplicação de algoritmos de cálculo de métricas de desempenho aplicadas à estimação de energia e potência para obtenção de resultados para analisar;
4. Análise empírica dos resultados obtidos recorrendo a técnicas de análise de dados.

Neste relatório foram agrupadas em 3 capítulos as fases do projeto.

Capítulo I	• Investigação e aquisição de conhecimento • Preparação do ambiente
Capítulo II	• Cálculo de métricas de estimação de energia • Análise empírica dos dados
Capítulo III	• Conclusão • Limitações • Trabalho Futuro

Capítulo I- Investigação e Aquisição de Conhecimento

1.1. Revisão de Literatura

Nesta fase a leitura de artigos e pesquisa de conceitos foi essencial para me familiarizar com NILM e perceber alguns conceitos básicos sobre a tecnologia. Esta fase demorou algum tempo pois há uma grande variedade de conceitos associados a esta tecnologia, que tive de procurar entender e um vasto conhecimento foi adquirido para poder trabalhar com a mesma. Foram indicados pelo orientador alguns artigos sobre NILM que me ajudaram a familiarizar com toda a tecnologia, nomeadamente:

- "Energy management using non-intrusive load monitoring techniques – State-of-the-art and future research directions"
- "Towards Trustworthy Energy Disaggregation: A Revie of Challenges, Methods, and Perspectives for Non-Intrusive Load Monitoring"
- "Performance evaluation in non-intrusive load monitoring: Datasets, metrics, and tools – A review"
- "A comparison of Performance Metrics for Event Classification in Non-Intrusive Load Monitoring"
- "An empirical exploration of performance metrics for event detection algorithms in Non-Intrusive Load Monitoring"
- "NILMTK: An Open Source Toolkit for Non-intrusive Load Monitoring, Towards reproducible state-of-the-art energy disaggregation"
- "Unlocking the Full Potential of Neural NILM: On Automation, Hyperparameters & Modular Pipelines"

1.1.1. Sumário da Revisão

Após a leitura dos dois primeiros artigos foi possível perceber do que se trata NILM e o que o mesmo visa solucionar e apoiar nas sociedades de hoje. Esta tecnologia é focada no controle dos consumos energéticos aplicando para tal o mínimo de intervenções em uma rede elétrica, pretende, através de algoritmos de desagregação de energia, capturar os consumos individuais de cada aparelho eletrónico de um edifício de um conjunto agregado de energia. Foi

possível entender como a mesma é utilizada e algumas barreiras que ainda necessitam ser ultrapassadas para o aperfeiçoamento da mesma.

Os três artigos seguintes, falam sobre quais os conjuntos de ferramentas, conjuntos de dados e métricas de desempenho que existem para trabalhar e estudar NILM. Com a leitura destes artigos foi possível perceber que existem algumas ferramentas para trabalhar os dados recolhidos dos consumos agregados, dados esses que, de modo a facilitar o estudo e a investigação da tecnologia, foram recolhidos por diversas instituições em diversos lugares e guardados em conjuntos de dados públicos para que estudantes e investigadores possam explorar e melhorar esta tecnologia utilizando medidas de referência sem terem de despender tempo na recolha de novos dados para análise, como é evidente, no caso de ser necessário o estudo sob alguma condição específica novos dados teriam de ser recolhidos. As métricas referidas nestes artigos visam medir o desempenho em duas categorias, deteção de eventos e a estimação de energia e potência. Um dos artigos refere algumas ferramentas que podemos utilizar para trabalhar com estes dados, analisar e obter informações úteis dos mesmos.

Os últimos dois artigos falam exatamente dos conjuntos de ferramentas que podem ser utilizados para trabalhar com os conjuntos de dados de referência que existem. Nesses artigos são referenciados 3 conjuntos de ferramentas que existem e que estão disponíveis publicamente para utilização, cada um com as suas ferramentas diferentes.

Até então, já tinha reunidas as condições teóricas necessárias para trabalhar com a tecnologia, já tinha investigado sobre os conceitos-chave que me poderia deparar ao longo do projeto a desenvolver e já sabia que conjuntos de ferramentas necessitaria para trabalhar com NILM. A linguagem de programação mais utilizada para trabalhar comeste tipo de informação é o Python, linguagem na qual já tinha algumas bases e foi fácil adaptar o que sabia ao que tinha de trabalhar.

1.2. Preparação do ambiente

Após a revisão de literatura percebi que seria necessário preparar um ambiente para poder trabalhar a informação dos conjuntos de dados públicos e conseguir resolver o problema que me havia sido proposto. Neste ambiente tive de preparar tudo, desde o conjunto de ferramentas para trabalhar com os conjuntos de dados, experiências feitas sob um conjunto de dados para extrair informação de um conjunto de sinais e as métricas de estimação de energia

para poder efetuar os cálculos e por fim conseguir realizar uma análise empírica para poder responder ao problema em questão.

1.2.1. Revisão de Ferramentas, Conjuntos de Dados e Métricas de Estimação de Energia

Para trabalhar com a tecnologia e de modo a facilitar a comparação entre algoritmos de desagregação de energia foram desenvolvidos alguns conjuntos de ferramentas de fonte aberta, nomeadamente Non-intrusive Load Monitoring Toolkit (NILMTK), NILMTK-contrib, Deep-NILMtk, entre outras. Para resolução do problema proposto utilizei duas destas ferramentas, o NILMTK e o NILMTK-contrib.

1.2.1.1. Non-intrusive Load Monitoring Toolkit (NILMTK)

Este toolkit contém um conjunto de funcionalidades para trabalhar com dados de desagregação de energia, nomeadamente, funcionalidades como, analisadores para uma série de conjuntos de dados existentes, uma coleção de algoritmos de pré-processamento, um conjunto de estatísticas para descrever conjuntos de dados, dois algoritmos de desagregação de referência e um conjunto de métricas de precisão. Este toolkit conta atualmente com mais de 30 contribuidores que contribuíram para melhorar e corrigir alguns erros deste toolkit, sendo que, a equipa que desenvolveu a base do mesmo foi Nipun Batra, Jack Kelly e Oliver Parson. Mais documentação acerca do toolkit pode ser encontrada na página criada acerca do mesmo: https://nilmtk.github.io/.

1.2.1.2. NILMTK-contrib

Alguns contribuidores acharam necessário a implementação de mais algumas ferramentas para poder trabalhar os conjuntos de dados disponíveis utilizando diferentes métodos e algoritmos, então desenvolveram uma extensão deste toolkit, o NILMTK-contrib, para suportar novos algoritmos e uma melhoria ao objeto API que permite ao utilizar trabalhar mais facilmente com os modelos sem ter muito conhecimento dos mesmos.

Os novos algoritmos adicionados com esta extensão foram, nomeadamente:

- Additive Factorial Hidden Markov Model
- Additive Factorial Hidden Markov Model with Signal Aggregate Constraints
- Discriminative Sparse Coding
- RNN
- Denoising Auto Encoder
- Seq2Point
- Seq2Seq
- WindowGRU

1.2.1.3. Deep-NILMtk

Os avanços nos modelos de redes neurais artificiais levaram a que fossem desenvolvidas novas ferramentas baseadas no funcionamento de redes neurais profundas para poder trabalhar com a vasta quantidade de dados obtida pelos medidores inteligentes e assim poder obter resultados mais corretos. Nesta nova extensão, o Deep-NILMtk, desenvolvida para adicionar tais ferramentas, foi adicionada compatibilidade com diversas estruturas de aprendizagem profunda, conjuntos de processos de NILM customizáveis, templates de experiências e ferramentas de aprendizagem automática.

1.2.1.4. Conjuntos de Dados

Existem diversos conjuntos de dados disponíveis para facilitar o trabalho com o NILMTK, estes conjuntos de dados foram obtidos de diferentes edifícios, desde conjuntos habitacionais a edifícios comerciais, entre outros, e são armazenados em diversos formatos desde frequências a valores numéricos e como tal o NILMTK facilita a conversão destes conjuntos de dados em valores que possam ser utilizados para treinar modelos, ou seja, convertem os diferentes tipos de dados num formato unificado e percetível pelos algoritmos de desagregação de energia.

Na Tabela 1 podemos observar de forma resumida a informação de alguns conjuntos de dados que estão disponíveis publicamente.

Conjunto de Dados	*Instituição*	*Localização*	*Duração por casa*	*Número de casas*	*Frequência de amostragem dos aparelhos*	*Frequência de amostragem agregada*
REDD (2011)	MIT	MA, USA	3–19 days	6	3 sec	1 sec & 15kHz
BLUED (2012)	CMU	PA, USA	8 days	1	N / A*	12 kHz
Smart (2012)*	UMass	MA, USA	3 months	3	1 sec	1 sec
Tracebase (2012)	Darmstadt	Germany	N / A	N / A	1–10 sec	N / A
Sample (2013)	Pecan Street	TX, USA	7 days	10	1 min	1 min
HES (2013)	DECC, DEFRA	UK	1 or 12 months	251	2 or 10 min	2 or 10 min
AMPds (2013)	Simon Farser U.	BC, Canada	1 year	1	1 min	1 min
iAWE (2013)	IIIT Delhi	Delhi, India	73 days	1	1 or 6 sec	1 sec
UK-DALE (2014)	Imperial College	London, UK	3–17 months	4	6 sec	1–6 sec & 16 kHz
REFIT (2014)	University of Strathclyde	Loughborough, UK	2 years	20	6-8 sec	

*Tabela 1: Comparação de conjuntos de dados energéticos domésticos. *BLUED rotula as transições de estado para cada aparelho.*

Para o desenvolvimento deste projeto foi utilizado o conjunto de dados REFIT, pois este contém bastante informação, sendo que foram coletados dados de 2 anos sobre 20 edifícios. Os resultados das desagregações energéticas realizadas a este conjunto de dados foram previamente fornecidos pelo orientador.

1.2.1.5. Métricas de Estimação de Energia

As métricas de estimação de energia no NILM são utilizadas para avaliar o desempenho e a precisão dos algoritmos NILM na estimação do consumo de energia ao nível dos aparelhos. Estas métricas ajudam os investigadores a avaliar o desempenho dos algoritmos NILM, a comparar diferentes abordagens e a identificar áreas de melhoria.

Na Tabela 2 encontram-se algumas das métricas utilizadas no desenvolvimento deste projeto.

Métrica	*Descrição*	*Equação*
RE	O erro relativo dá uma indicação da qualidade da estimativa de energia relativamente aos dados de referência.	$\frac{\sum_{i=1}^{N} E_i - \sum_{i=1}^{N} \hat{E}_i}{\sum_{i=1}^{N} E_i}$
RMSE	A raiz do erro quadrático é o desvio padrão dos erros de estimativa da energia. O RMSE informa sobre o grau de dispersão destes erros. Por outras palavras, diz-nos quão concentradas estão as estimativas em torno dos valores reais. O RMSE é apresentado na mesma unidade que os dados, o que o torna uma métrica intuitiva.	$1 - \sqrt{\frac{1}{N}\sum_{i=1}^{N}\left(E_i - \hat{E}_i\right)^2}\overline{E}$
AE	O erro médio indica se a energia estimada está, em média, sobrestimada ou subestimada. Um AE positivo implica uma maior proporção global de sobrestimação, um AE negativo implica uma maior proporção de subestimação.	$\frac{1}{N}\sum_{i=1}^{N}\Delta E_i$
SDE	O desvio-padrão do erro indica o grau de dispersão das diferenças em torno da estimativa do AE. Um SDE maior implica uma dispersão mais alargada dos valores individuais estimados, um SDE menor implica distribuições mais apertadas.	$\sqrt{\frac{1}{N}\sum_{i=1}^{N}\left(\Delta E_i - \overline{\Delta E}\right)^2}$
r2	O R-quadrado é uma medida estatística da proximidade entre as estimativas e os dados do terreno verdadeiro. Quanto maior o coeficiente, maior a percentagem de estimativas que está em linha com a verdade terrestre. Valores de 1 ou 0 indicam que a estimativa representa todos ou nenhum dos dados, respetivamente.	$1 - \sum_{i=1}^{N} \frac{\left(E_i - \hat{E}_i\right)^2}{\left(E_i - \overline{E}_i\right)^2}$
%SDx	A percentagem (%) de desvio-padrão explicado é a percentagem em que o desvio-padrão dos erros é inferior ao desvio-padrão dos dados medidos. Acredita-se que seja mais intuitivo do que o r2, uma vez que é apresentado nas mesmas unidades que os dados reais.	$1 - \sqrt{1 - r^2}$
EE	A métrica do erro de energia é o rácio entre a diferença absoluta entre a energia estimada e a energia real e a quantidade total de energia real.	$\frac{\sum_{i=1}^{N} \lvert \hat{E}_i - E_i \rvert}{\sum_{i=1}^{N} E_i}$
EAv1	A exatidão da energia foi proposta como uma tentativa de reportar o erro de energia entre 0 e 1.	$e^{-\alpha(EE)}$
MR	A taxa de correspondência é uma métrica em que a avaliação se baseia na taxa de sobreposição da energia verdadeira e estimada. Varia entre 0 e 1. Quando o valor tende para 1, a métrica indica uma forte correspondência entre a energia estimada e a verdadeira. Pelo contrário, um valor que tende para 0 indica uma fraca correspondência. Um valor de zero só é possível se a energia verdadeira e a energia estimada forem zero.	$\frac{\sum_{i=1}^{N} min\left\{E_i, \hat{E}_i\right\}}{\sum_{i=1}^{N} max\left\{E_i, \hat{E}_i\right\}}$

SEM	O erro padrão da média informa sobre a forma como a média varia com diferentes experiências que medem a mesma quantidade. No caso da estimativa de energia, se houver erros significativos, o SEM será mais elevado. Pelo contrário, se houver poucos ou nenhuns erros significativos, o SEM tenderá para zero.	$\frac{\sigma}{\sqrt{N}}$
FEE	A fração de energia explicada é o rácio entre a energia total estimada e a energia efetivamente utilizada. Em Berges (2010), esta métrica é designada por taxa de identificação de energia (EIR).	$\frac{\sum_{i=1}^{N}\hat{E}_i}{\sum_{i=1}^{N}E_i}$
TECA	A energia total corretamente atribuída é o erro total na energia atribuída, normalizado pelo consumo real de energia em cada fatia de tempo, calculada a média de todos os aparelhos.	$\frac{1-\sum_{i=1}^{N}\sum_{a=1}^{A}\lvert\hat{E}_i^{(a)}-E_i^{(a)}\rvert}{2\sum_{i=1}^{N}E_i}$
ETEA	O erro em Energia total atribuída é a diferença entre a energia total atribuída e a energia efetivamente consumida por um determinado aparelho no conjunto de dados.	$\left\lvert\sum_{i=1}^{N}\hat{E}_i^{(a)}-\sum_{i=1}^{N}E_i^{(a)}\right\rvert$
Dev	O desvio determina o desvio entre o consumo de eletricidade inferido e o consumo real de eletricidade no conjunto de dados. É o rácio entre a ETEA e a energia efetivamente consumida.	$\frac{ETEA}{\sum_{i=1}^{N}E_i^{(a)}}$
T	Número de observações ou eventos com base em cada passo de tempo	
A	Número de aparelhos considerados	
E_t	Energia medida no intervalo de tempo t	
$E_t^{(a)}$	Energia medida para o aparelho 'a' no intervalo t	
$\bar{E}$	Energia média medida no conjunto de dados	
$\hat{E}_t$	Energia estimada no instante t	
$\hat{E}_t^{(a)}$	Energia estimada para o aparelho 'a' no instante t	
ΔE_t	Erro entre os dados da NILM e os dados medidos no instante t	
$\overline{\Delta E}$	Erro entre o NILM e os dados medidos em todo o conjunto de dados	

Tabela 2: Visão geral dos indicadores de desempenho na categoria de estimativa de energia.

1.2.2. Instalação dos Toolkits- Tutorial

Preparar uma máquina para trabalhar com o NILMTK foi um desafio bastante grande. A instalação do toolkit não é tão simples, e a escassez de documentação de como instalar o mesmo não ajudou no processo. Após alguma pesquisa e tentativa e erro a instalação acabou por ser bem-sucedida na minha máquina, pelo que levei algum tempo para conseguir estabilizar um ambiente com este toolkit.

Para instalar o toolkit recomenda-se a utilização do Anaconda, um software que permite criar ambientes de desenvolvimento em Python ou R onde é possível criar ambientes separados para aplicações distintas, é bastante eficiente quando queremos desenvolver alguma pesquisa ou ciência. No Anaconda são disponibilizados vários canais onde podemos obter pacotes de ferramentas Python tal e qual fazemos no gestor de pacotes Python. O pacote do NILMTK bem como do NILMTK-contrib está disponível através do canal nilmtk no Anaconda de onde pode ser instalado.

Para conseguir o ambiente de Anaconda com o NILMTK a funcionar corretamente na minha máquina tive de instalar cada requisito do pacote na versão indicada pelos proprietários para poder instalar o toolkit corretamente.

O processo que funcionou corretamente para a instalação na minha máquina foi o seguinte:

1. Instalar o Anaconda e abrir o Anaconda Prompt;
2. Criar um ambiente no Anaconda com o Python na versão 3.6.x:
   - conda create -n <environment-name> python=3.6
   - conda activate <environment-name>
3. Fazer o download do pacote do NILMTK, NILMTK-contrib e NILM Metadata diretamente do canal do nilmtk no Anaconda.org (https://anaconda.org/nilmtk/repo):

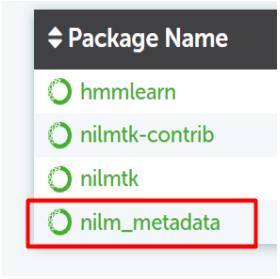


Figura 1: Pacote do Anaconda.

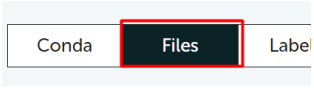


Figura 2: Aba de ficheiros para download do pacote.

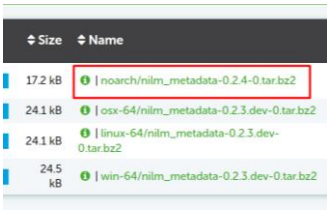


Figura 3: Ficheiro do pacote Anaconda para descarregar.

- Repetir o processo para os 3 pacotes.

4. Instalar as dependências do NILMTK que se encontram no environment.yml (https://github.com/nilmtk/nilmtk/blob/master/environment.yml) do repositório do nilmtk utilizando o pip que é o Python package manager que vem definido por defeito no ambiente Anaconda:
   - pip install numpy==1.13.3
   - pip install tables
   - pip install pandas==0.25.3
   - pip install matplotlib==3.1.0
   - pip install networkx==2.1
   - pip install scipy==1.0.0
   - pip install scikit-learn==0.21.2
   - pip install jupyterlab
   - pip install nose
   - pip install hmmlearn
5. Instalar as dependências do NILMTK-contrib que se encontram no setup.py (https://github.com/nilmtk/nilmtk-contrib/blob/master/setup.py) do repositório do nilmtk-contrib utilizando o pip que é o Python package manager que vem definido por defeito no ambiente Anaconda:
   - pip install tensorflow==2.0
   - pip install cvxpy=1.0.0
   - pip install keras==2.2.4
6. Instalar os pacotes descarregados no passo 3. navegando no Anaconda Prompt para o diretório onde se encontram os arquivos descarregados. Uma vez no diretório só temos de executar os seguintes comandos através do instalador de pacotes do Anaconda:

- conda install nilm_metadata-x.x.x
- conda install nilmtk-x.x.x
- conda install nilmtk-contrib-x.x.x

7. Por fim temos o NILMTK e NILMTK-contrib completamente instalado e podemos utilizá-lo através do Jupyter ou através do seu IDE de eleição. Em certos IDE's poderá ser necessário instalar um pacote extra, o jupyter:
   - pip install jupyter

Para instalar o Deep-NILMTK temos de seguir com a instalação dos pacotes requeridos por este toolkit, podemos fazê-lo seguindo os próximos passos:

8. Instalar as dependências do Deep-NILMTK que se encontram no requirements.txt (https://github.com/BHafsa/deep-nilmtk-v1/blob/master/requirements.txt) do repositório do Deep-NILMTK:
   - pip install mlflow==1.23.1
   - pip install optuna==2.10.0
   - pip install pytorch-lightning==1.5.10
   - pip install torch==1.10.2
   - pip install torchaudio==0.10.2
   - pip install torchvision==0.11.3
9. E por fim instalar o pacote do Deep-NILMTK diretamente do repositório utilizando o pip:
   - pip install git+https://github.com/BHafsa/deep-nilmtk-v1
10. Para finalizar e certificar que tudo funciona como esperado é recomendado verificar as versões dos pacotes pois algumas podem alterar com as instalações dos seguintes. Para verificar as versões dos pacotes podemos correr no prompt o comando:
    - conda list

1.2.3. Impedimentos

Foi disponibilizada uma máquina no ARDITI para poder realizar este trabalho visto que o processamento com NILM é bastante complexo por estarmos a lidar com uma imensa quantidade de dados. Esta máquina foi bastante desafiante para tentar preparar o ambiente de desenvolvimento de Python com o NILMTK, pois havia sempre alguns erros de dependências que não consegui resolver, então para poder avançar com o trabalho decidi deixar esta máquina

para o processamento das métricas de estimação de energia e processar as experiências para obter dados reais e previstos de consumo diretamente na minha máquina.

No início tínhamos acordado a utilização dos novos algoritmos do Deep-NILMTK para poder fazer uma análise de mais dados, mas por falta de tempo e instabilidade deste novo toolkit não foram realizados esforços no sentido de obter dados utilizando métodos do Deep-NILMTK. No futuro seria bom incluir também o cálculo das métricas de estimação de energia sobre experiências realizadas com os métodos introduzidos pelo Deep-NILMTK para podermos ter uma comparação dos valores utilizando redes neurais artificiais e redes neurais profundas.

Capítulo II- Obtenção de Resultados

2.1. Introdução

Após o sucesso da fase anterior e ter o ambiente anaconda completamente estável a correr o NILMTK, estavam reunidas as condições para poder avançar com a obtenção dos valores reais e previstos de consumo de cada experiência efetuada para poder calcular as métricas de estimação de energia, pois estas são baseadas exatamente na correspondência entre o que realmente foi consumido de energia e o que o algoritmo conseguiu prever.

2.2. O que existe?

2.2.1. Experiências

Experiências é o que designamos ao resultado obtido do treino de um algoritmo de desagregação de energia sobre um determinado conjunto de dados com os parâmetros que definimos. O resultado destas experiências são objetos API de cada eletrodoméstico desagregado do consumo agregado proveniente do conjunto de dados.

Previamente foi fornecido um conjunto de ficheiros num formato comprido, cada um destes ficheiros contendo um objeto API comprimido por meio de uma biblioteca de Python, o compress-pickle, que é uma ferramenta Python que permite uma fácil serialização de objetos do Python para serem guardados localmente no disco, isto permite que um objeto API, de grandes dimensões, seja armazenado de forma reduzida localmente para mais tarde ser utilizado em outro tipo de experiências.

As experiências fornecidas foram realizadas sobre o conjunto de dados REFIT, utilizando diferentes algoritmos de desagregação de energia, nomeadamente:

- Combinatorial Optimisation (CO)
- Exact Factorial Hidden Markov Model (FHMME)
- Discriminative Sparse Coding (DSC)
- Edge Detection (Hart85)
- Mean
- RNN

Dessas mesmas experiências resulta a desagregação da energia agregada em cada um dos eletrodomésticos das 20 casas observadas no REFIT, máquina de lavar louça (DW), frigorífico com congelador (FF), frigorífico (FR/FRI), congelador (FZ), chaleira (KT), micro-ondas (MW), televisão (TV) e máquina de lavar roupa (WM). Na Tabela 3 podemos observar o número de objetos API recolhidos das experiências previamente realizadas pelo orientador, e que me foram fornecidos.

	Eletrodoméstico								
Algoritmo	*DW*	*FF*	*FR*	*FZ*	*KT*	*MW*	*TV*	*WM*	*Total*
CO	15	14	7	11	14	16	19	19	115
DSC	15	14	7	11	14	16	19	19	115
FHMME	15	16	7	11	14	16	19	19	117
Hart85	15	14	7	11	14	16	19	19	115
Mean	15	14	7	11	14	16	19	19	115
RNN	0	0	7	11	0	0	0	0	18
Total	75	72	42	66	70	80	95	95	595

Tabela 3: Experiências obtidas previamente ao cálculo das métricas.

2.2.2. Métricas de Estimação de Energia

Métricas de desempenho na categoria de estimação de energia não são mais do que simples fórmulas matemáticas que representam numericamente o desempenho de um determinado algoritmo a prever o consumo de energia de um determinado eletrodoméstico, como tal, estas métricas podem ser descritas num algoritmo simples que apoia os estudantes no cálculo das mesmas.

Foi-me fornecido pelo orientador um algoritmo previamente programado contendo um conjunto de métricas de desempenho não só sobre a categoria de estimação de energia (Tabela 2) como também de deteção de eventos, claro que o que me foi relevante foram as métricas de estimação de energia. Este algoritmo, por sua vez, calcula cada métrica com base em dois parâmetros, uma lista de valores de consumos reais e outra de consumos previstos.

2.3. Preparação de Dados

Para poder avançar com a fase final do projeto proposto, a análise empírica, os dados fornecidos tiveram de ser trabalhados e passados por uma série de processos para poder obter uma perspetiva visual que nos ajudaria a fundamentar a resposta ao problema proposto. No diagrama abaixo podemos observar em quantas fases foi separado o processamento dos dados para no fim obter perceções sobre os dados.

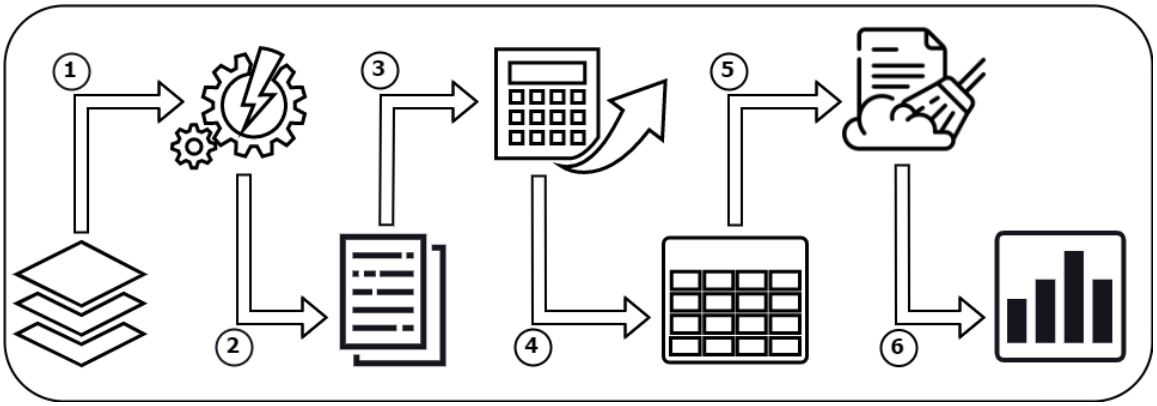


Figura 4: Diagrama do processo de preparação dos dados.

2.3.1. Tratamento dos ficheiros comprimidos (1, 2)

Para efetuar o cálculo das métricas de estimação de energia, o algoritmo fornecido pelo orientador recebe dois parâmetros, como mencionado anteriormente, recebe uma lista com os valores reais de consumo e outra com os valores previstos de consumo. Estes valores, são valores que são resultantes da realização de uma experiência e como tal os mesmos encontram-se dentro dos objetos API fornecidos, objetos esses que foram fornecidos num formato comprimido. Estes objetos comprimidos foram fornecidos numa hierarquia de diretórios onde no primeiro diretório encontramos uma pasta para cada algoritmo de desagregação de energia

utilizado e dentro de cada uma delas existia uma nova pasta para cada eletrodoméstico desagregado que continha os objetos API comprimidos de cada casa do conjunto de dados.

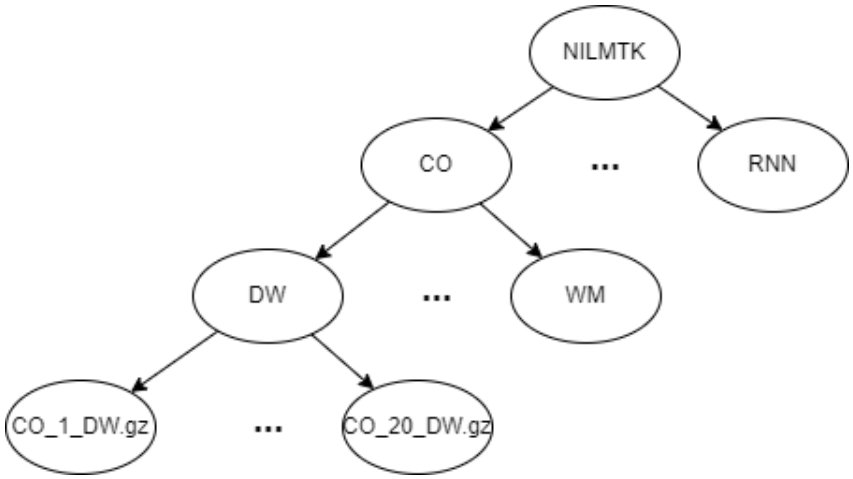


Figura 5: Diretórios dos ficheiros partilhados.

Para conseguir descomprimir estes objetos API e obter os valores de consumo real e consumo previsto foi desenvolvido um algoritmo (Figura 7) que segue a hierarquia de diretórios e utilizando a biblioteca compress-pickle, descomprime cada experiência e obtém da mesma a lista de valores de consumo real e a lista de valores de consumo previsto e coloca esta informação num dicionário de Python, que tem como chave o nome do objeto API, por exemplo, CO_1_DW.gz e como valor um dicionário com as chaves 'gt' e 'pred' e como valores a lista de consumos reais e a lista de consumos previstos respetivamente.

```
{ API name : {
                'gt' : DataFrame,
                'pred' : DataFrame
              } ,
 ... ,
 API name : {
                'gt' : DataFrame,
                'pred' : DataFrame
              }
}
```

Figura 6: Estrutura do dicionário para armazenar os valores de consumo reais e previstos.

Após ter obtido o dicionário com todas as API's existentes nos ficheiros partilhados, o mesmo foi novamente comprimido utilizando a biblioteca compress-pickle e guardado

localmente para poder ser utilizado a qualquer altura numa fase posterior do processamento dos dados.

O código desenvolvido (Figura 7) não teve em conta a otimização nem a complexidade dos dados, o objetivo foi apenas conseguir obter os valores de cada experiência para posteriormente poder calcular as métricas de estimação de energia. Este processo foi bastante demorado, pois cada objeto API tem uma grande dimensão, pois estamos a falar de um objeto que tem como dados 4 meses de observações energéticas a cada 10 minutos, como tal esta extração dos valores reais e previstos de consumo demorou cerca de 4 dias ser obtida, tendo em conta também que a máquina utilizada não é especializada para processamento.

```
#%%
import compress_pickle as cp
import os
#%%
#get all files in directory
methods_path = ['CO/', 'FHMME/', 'Hart85/', 'Mean/', 'RNN/']
appliances_path = ['DW/', 'FF/', 'FR/', 'FZ/', 'KT/', 'MW/', 'TV/', 'WM/']
path = '../experiments/NILMTK/'
experiments_gt_and_pred_df = {}
experiments_error = []
#%%
for method in methods_path:
    if method == 'RNN/':
        files = os.listdir(path + method)
        files = [file for file in files if '.gz' in file]
        for f in files:
            try:
                api = cp.load(path + method + f)
                experiments_gt_and_pred_df.update({f: {'gt': api.gt_overall,
                                                       'pred': api.pred_overall[api.classifiers[0][0]]}})
            except:
                experiments_error.append(f)
    else:
        for appliance in appliances_path:
            files = os.listdir(path + method + appliance)
            files = [file for file in files if '.gz' in file]
            for f in files:
                try:
                    api = cp.load(path + method + appliance + f)
                    experiments_gt_and_pred_df.update({f: {'gt': api.gt_overall,
                                                           'pred': api.pred_overall[api.classifiers[0][0]]}})
                except:
                    experiments_error.append(f)
#%%
cp.dump(experiments_gt_and_pred_df, '../experiments/NILMTK/experiments_gt_and_pred_df.gz', compression='gzip')
```

Figura 7: Código desenvolvido para obter os valores de consumo reais e previstos de todas as experiências.

Durante a obtenção destes dados houve um algoritmo de desagregação que não foi contemplado, que foi o DSC, pois era possível carregar as experiências devido a uma incompatibilidade de versões de pacotes Python instalados.

2.3.2. Cálculo de métricas (3, 4)

Com os valores de consumo reais e previstos extraídos de todos os objetos API, já estávamos em condições de calcular as métricas de estimação de energia, previamente fornecidas pelo orientador em prol da execução deste projeto.

O algoritmo fornecido tinha um conjunto de funções que permitiram agilizar o processo do cálculo das métricas, onde numa função principal podíamos dar como parâmetros uma lista de valores de consumo reais, uma lista de valores de consumos previstos e uma lista de métricas para calcular, por sua vez esta função devolvia uma matriz com todas as métricas calculadas para cada objeto API de onde foram retirados os valores de consumo reais e previstos.

Desenvolvi um algoritmo que percorria todo o dicionário dos objetos API com os valores de consumo reais e previstos extraídos e para cada objeto API invocava a função do cálculo das métricas de estimação de energia e fornecia os parâmetros pedidos, a lista dos valores de consumo reais e a lista dos valores de consumo previstos do objeto API em questão, bem como, uma lista de métricas que pretendia calcular.

Toda a informação das métricas foi armazenada, à medida que iam sendo calculados por objeto API iam sendo guardados num novo dicionário que tinha como chave o nome do objeto API e guardava como valores a lista das métricas calculadas com os respetivos valores do cálculo. Este dicionário, após o cálculo de todas as métricas para todos os objetos API, foi comprimido e guardado localmente para utilizar no seguimento do projeto.

No total foi-nos possível calcular 20 métricas de estimação de energia para cada um dos 480 objetos API fornecidos, logo ficamos com um total de cerca de 9600 valores para serem analisados. Tendo já uma base sólida para avançar com uma análise era necessário apenas limpar estes dados para podermos por fim criar visualizações destes dados para poderem ser analisados mais facilmente.

2.3.3. Limpeza dos dados (5)

Uma vez calculadas as métricas, foi necessário efetuar uma limpeza nos dados obtidos, pois existiam, por exemplo, alguns valores que não eram reconhecidos e como tal impossibilitava a construção de visualizações gráficas.

Comecei por transformar o dicionário criado com os cálculos das métricas num DataFrame, que é uma estrutura de dados da biblioteca Pandas, uma biblioteca para fácil manipulação de dados com várias ferramentas bastante úteis, para poder limpar valores NaN (Not a Number) e expandir algumas métricas que agrupavam um conjunto de resultados em listas nas devidas colunas com o respetivo nome da métrica.

Finalizado o processo da limpeza dos dados foi possível obter uma estrutura de dados limpa sem incongruências. O resultado foi um DataFrame, que pode ser visto como uma matriz, onde cada linha era indexada pelo nome do objeto API e cada coluna continha o valor calculado de uma métrica para o respetivo objeto API.

method_house_elec	*abse*	*ae*	*cv_rmsd*	*eav1*	…	*mae*	*mr_micro*
CO_10_DW	359.546402	354.134361	-23.586287	1.929908e-08	…	361.715759	0.066549
CO_11_DW	297.917852	296.587850	-126.949265	1.159018e-34	…	297.197723	0.015442
CO_13_DW	257.083493	256.243163	-23.904680	2.603702e-07	…	259.605865	0.083097
…	…	…	…	…	…	…	…
RNN_6_FZ	24.295816	6.048592	-0.948912	1.232127e-01	…	24.249762	0.227045
RNN_7_FRI	14.680895	2.049950	-0.955050	2.092576e-01	…	14.677900	0.317567
RNN_8_FZ	31.876781	0.861289	-0.883591	1.311890e-01	…	31.795506	0.169935

Tabela 4: Exemplo do DataFrame depois da limpeza dos dados.

2.3.4. Visualização dos Dados (6)

Com uma estrutura de dados limpa e correta, pude então tratar da visualização dos dados para poder analisar o nosso problema. Para dar resposta ao problema proposto optei por utilizar uma análise por agrupamento hierárquico que consiste em criar uma hierarquia de agrupamentos de métricas, o que nos permite avaliar quais grupos de métricas mais demonstram um bom resultado em avaliar o desempenho do algoritmo na estimação de energia.

Decidimos ainda dar um passo à frente e analisar quais métricas que mais se assemelham quando se trata de avaliar o desempenho do algoritmo na estimação de energia para um determinado tipo de aparelho, por exemplo, qual a melhor métrica para avaliar a estimação de energia do micro-ondas? Com isto formulámos a nossa hipótese para uma análise mais detalhada, que foi:

O aparelho influência a escolha da métrica.

Para explorar a hipótese comecei então por criar as respetivas matrizes de correlação para cada aparelho, utilizando o método Spearman no cálculo das mesmas para podermos ter em conta também relações não lineares entre as métricas. Para termos um ponto de comparação, calculei também a matriz de correlação de todos os dados obtidos para termos também uma comparação entre cada aparelho e o conjunto total dos aparelhos.

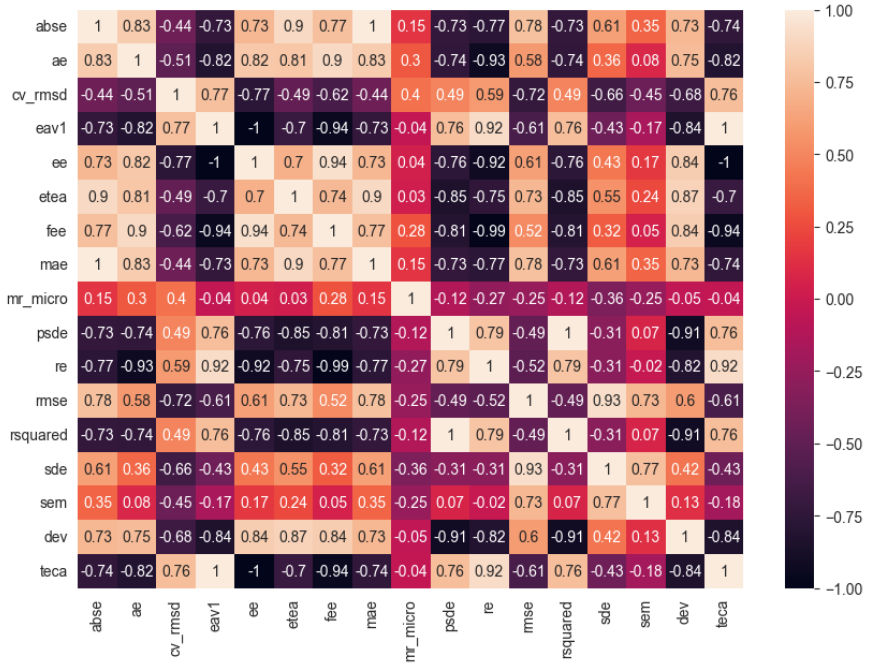


Figura 8: Mapa de calor obtido através da biblioteca Seaborn de todas as métricas calculadas.

Foi utilizada a biblioteca Pandas para calcular todas as matrizes de correlação, sendo que esta biblioteca conta com um método predefinido para o cálculo de matrizes de correlações, e para obter uma perceção mais visual sobre estas matrizes foi utilizada a biblioteca Seaborn, especializada na visualização de dados e baseada na Matplotlib, e que nos permitiu obter uma visualização muito mais explicita sobre as matrizes de correlação calculadas utilizando para tal uma representação em mapa de calor (Figura 8).

Após obtidas as matrizes de correlação, comecei então a produzir os nossos gráficos de agrupamento hierárquico. Para construir este agrupamento hierárquico apenas devemos ter em conta uma das partes da matriz de correlação, então transformei as matrizes obtidas em triangulares inferiores e calculei o absoluto da mesma para termos distâncias corretas entre as respetivas relações entre métricas.

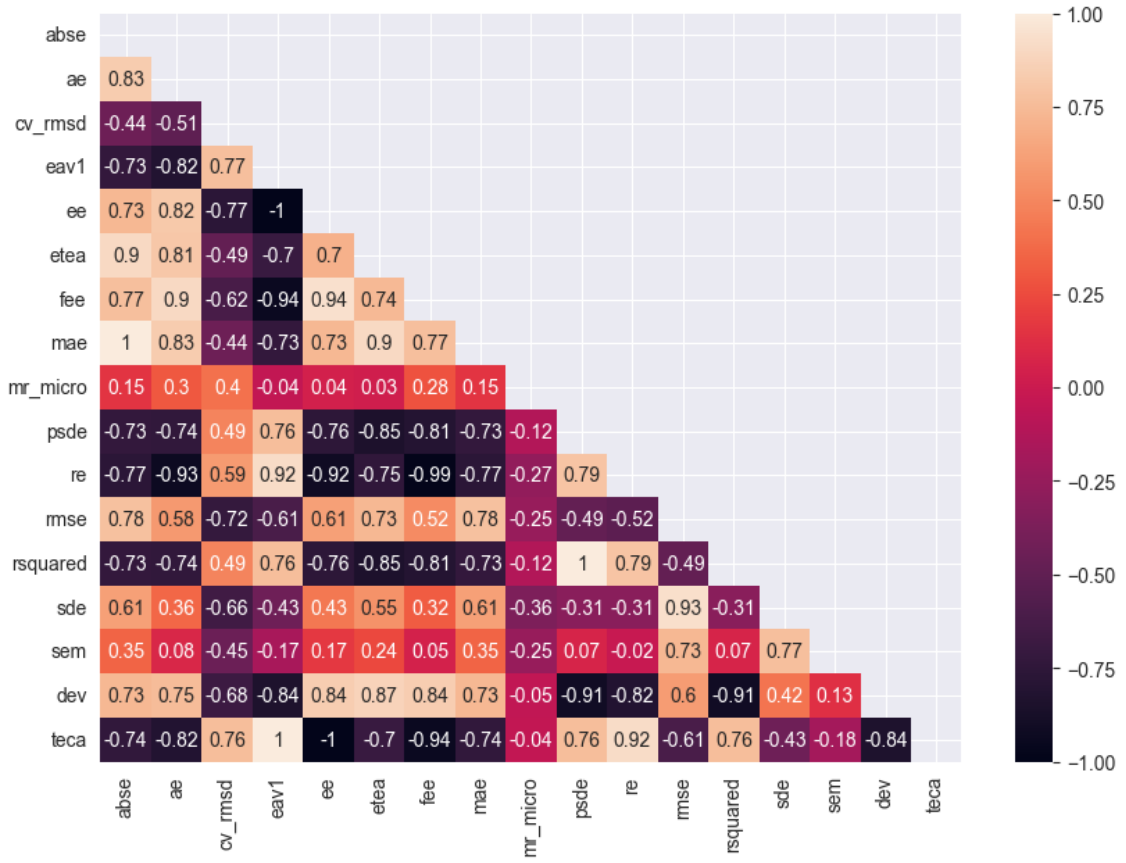


Figura 9: Mapa de calor obtido através da biblioteca Seaborn da matriz triangular inferior.

Neste momento as matrizes triangulares inferiores estavam prontas para serem transformadas em ligações e para isto foi utilizada uma ferramenta da biblioteca Scipy, que é uma biblioteca focada em algoritmos para resolução de problemas de computação científica, foi utilizada a ferramenta que faz o cálculo das ligações para depois podermos analisar as

mesmas utilizando para tal um dendrograma. No cálculo das ligações foi utilizado o método Ward que utiliza a métrica Euclidiana para o cálculo das distâncias dos pontos.

Depois de criadas as ligações Ward, representei as mesmas graficamente por meio de dendrogramas, utilizando também a biblioteca Scipy para a visualização. Os dendrogramas permitem visualizar como as métricas de agrupam tendo em conta as distâncias entre as suas correlações.

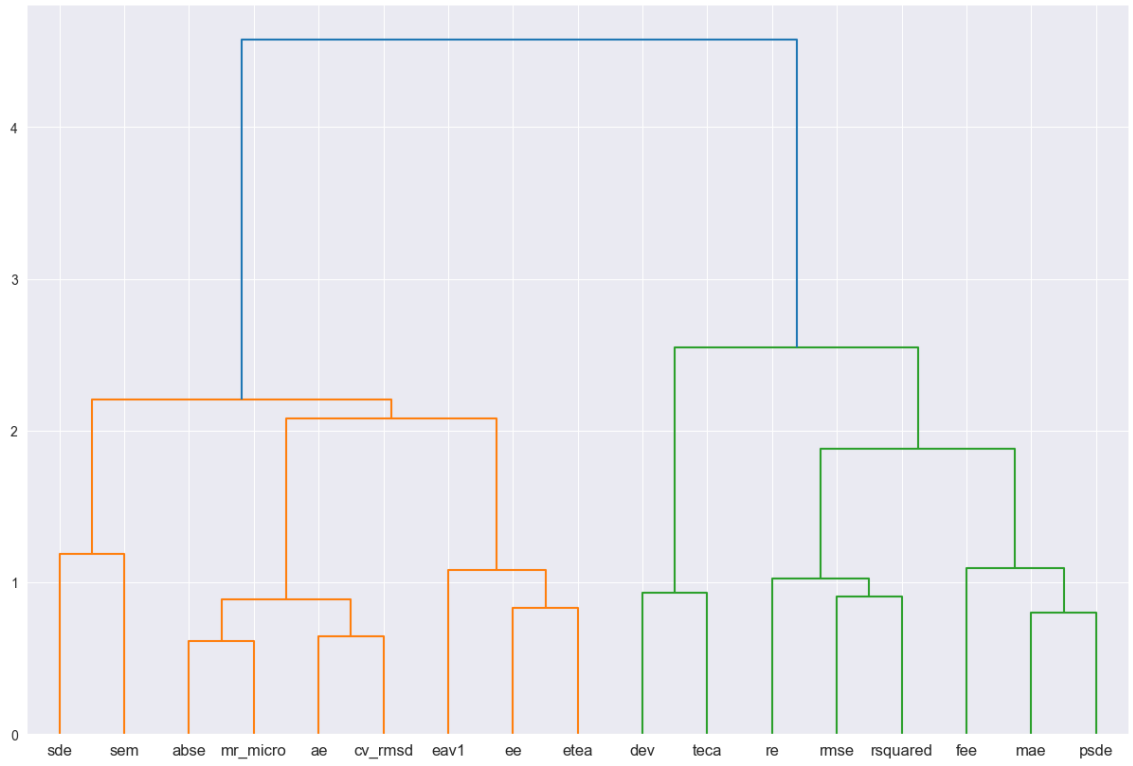


Figura 10: Dendrograma das métricas de estimação de energia para todos os objetos API.

Após toda a informação ter sido apresentada de forma visual, todos os gráficos obtidos foram extraídos para poder efetuar uma análise empírica sobre o problema apresentado e tentar provar a nossa hipótese.

2.4. Análise Empírica

Uma vez obtida uma perceção visual sobre as métricas de estimação de energia obtidas das experiências realizadas sobre o nosso conjunto de dados, estavam reunidas as condições para realizar uma análise empírica sobre os dados e explorar a nossa hipótese que foi se um determinado aparelho influenciaria a escolha da métrica de estimação de energia.

Para esta análise foram utilizados os dendrogramas obtidos dos diferentes aparelhos, onde decidi cortar o dendrograma em y = 2, onde na maioria obtínhamos um interessante agrupamento de métricas, era possível observar na maioria entre 4 e 5 agrupamentos de métricas.

De modo a termos um ponto mais geral sobre toda a informação, decidi realizar um dendrograma sobre todas as métricas calculadas de todos os objetos API, onde também este foi cortado em y = 2. Nele (Figura 11) pude verificar que, considerando o universo todo de dados obtidos, existem algumas métricas que mostram alguma semelhança entre si, tendo em conta o ponto escolhido para efetuar o corte do dendrograma observei 5 agrupamentos de métricas distintos, começando pela esquerda podes observar um agrupamento de duas métricas a SDE e a SEM que tendo em conta a natureza das duas faz sentido que se assimilem neste contexto, pois ambas estão associadas com erros da média. Logo a seguir observamos outro agrupamento com 4 métricas, a ABSE, a MR_Micro, a AE e a CV_RMSD, que dos agrupamentos obtidos são as que mais se assimilam, isto, pois estão todas baseadas no erro obtido entre o valor de consumo real e o valor previsto, daí a sua semelhança. O agrupamento seguinte, são 3 métricas que são a EAV1, a EE e a ETEA. Estes 3 primeiros agrupamentos são semelhantes entre si, pois trabalhos todos com valores médios entre os consumos reais e os consumos previstos. Na parte direita do dendrograma observamos apenas 2 agrupamentos, um com duas métricas e outro com seis métricas que se assimilam entre sim. Estes agrupamentos fazem-nos entender que existem certas métricas que podem ser substituídas por outras, pois, na verdade, iram retornar valores bastante semelhantes em relação ao que queremos avaliar numa experiência.

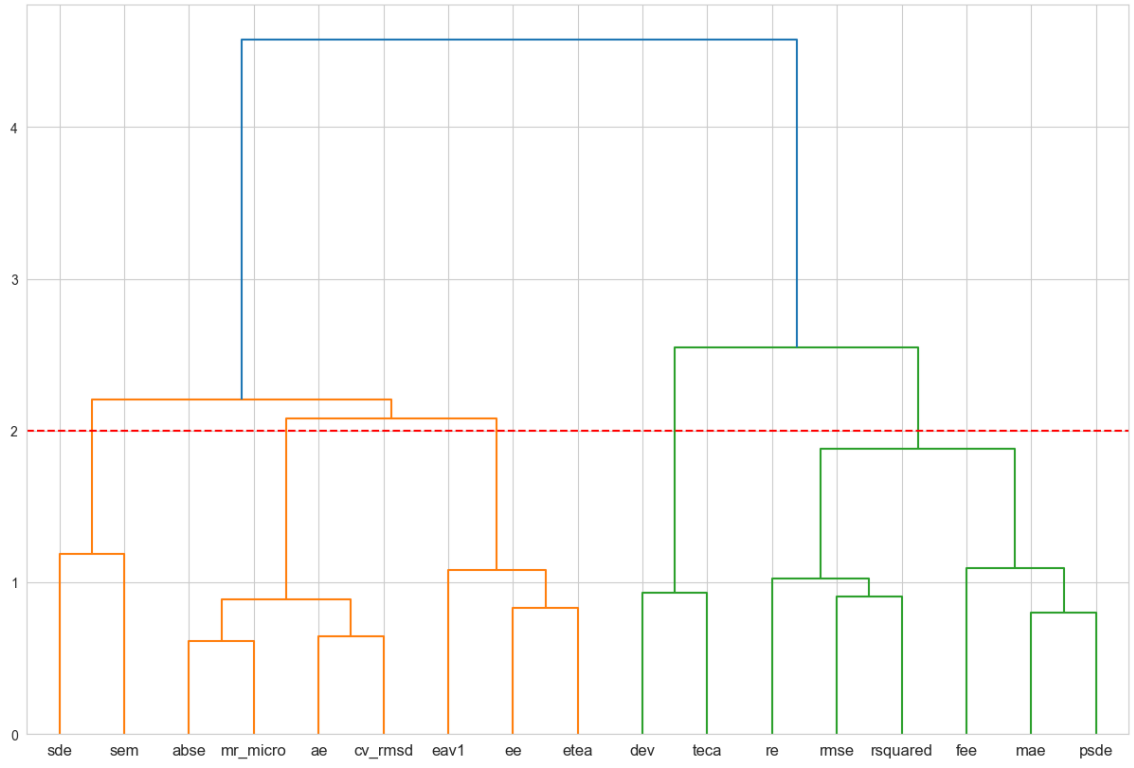


Figura 11: Dendrograma das métricas calculadas de todos os objetos API com um corte feito em y=2.

Assim que comecei a analisar os dendrogramas de cada aparelho notei algumas mudanças nos agrupamentos de métricas. Ao analisar o dendrograma dos objetos API das máquinas de lavar louça (Figura 12), pude observar que logo no primeiro agrupamento da esquerda, houve uma mudança, enquanto no universo geral de todos os objetos API's as métricas SDE e SEM tinham uma grande semelhança, no universo de API's das máquinas de lavar louça esta semelhança havia sido transformada numa dissemelhança, pelo que houve logo uma suspeita que um aparelho realmente pode influenciar a escolha de uma métrica de desempenho de estimação de energia.

Enquanto continuei a minha análise notei mais e mais diferença entre os agrupamentos de métricas de cada aparelho e do universo total de dados e até mesmo diferenças nos agrupamentos entre aparelhos. Uma conclusão retirada foi que existe mais semelhança entre os agrupamentos de métricas de aparelhos utilizados ocasionalmente, por exemplo, chaleira e micro-ondas, e uma dissemelhança quando comparamos um aparelho que está sempre ligado, como um frigorífico, o que me leva a questionar se além do aparelho escolhido, se também o tempo em que este está ligado influencia a seleção da métrica de desempenho.

Agrupei todos os dendrogramas na Tabela 5 onde é possível visualizar estas diferenças e tirar mais conclusões acerca dos diferentes agrupamentos de métricas dos diferentes aparelhos.

Após analisar os dendrogramas apercebi-me que realmente o aparelho escolhido deve influenciar a escolha da métrica, pois diferentes aparelhos agrupam diferentes métricas, e este agrupamento é por vezes tão dissemelhante de aparelho para aparelho que nos leva a entender que nem sempre a mesma métrica de desempenha na estimação de energia serve para medir se a energia estimada de um aparelho teve um tão bom desempenho como noutro aparelho. Se, por exemplo, quisermos saber o desempenho de um algoritmo na estimação de energia de uma chaleira, poderíamos utilizar a métrica PSDE ou a MR_MICRO, pois ambas demonstram ter uma grande semelhança (Figura 16), mas o mesmo já não é verdade se quisermos fazer o mesmo cálculo para um congelador, pois observando o dendrograma do congelador (Figura 15) vemos que estas duas métricas têm uma dissemelhança considerável.

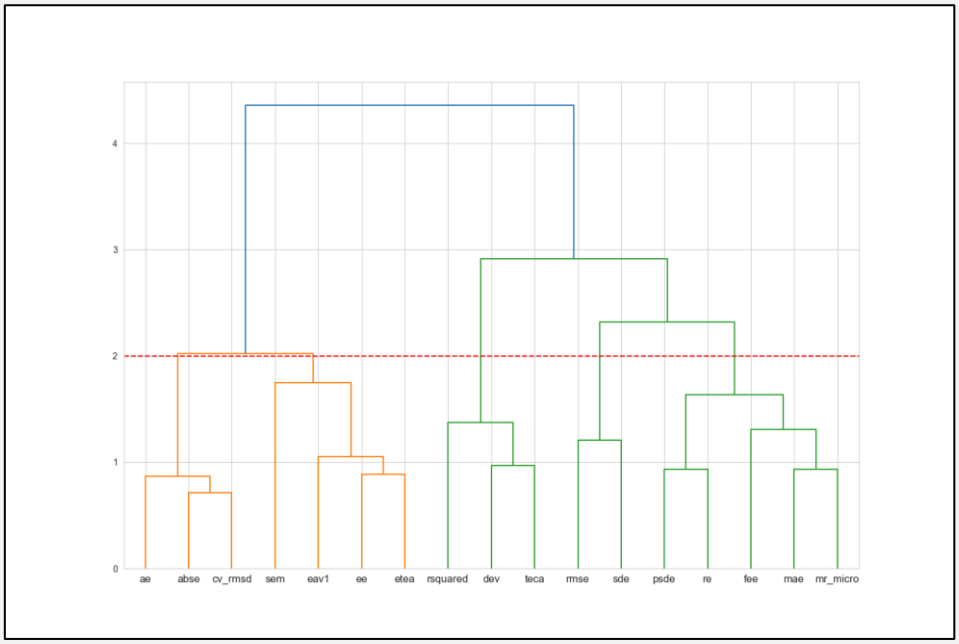
Aparelho	*Dendrograma*
DW	

Figura 12: Dendrograma das métricas de estimação de energia calculadas sobre todos os objetos API de máquinas de lavar louça.

FF

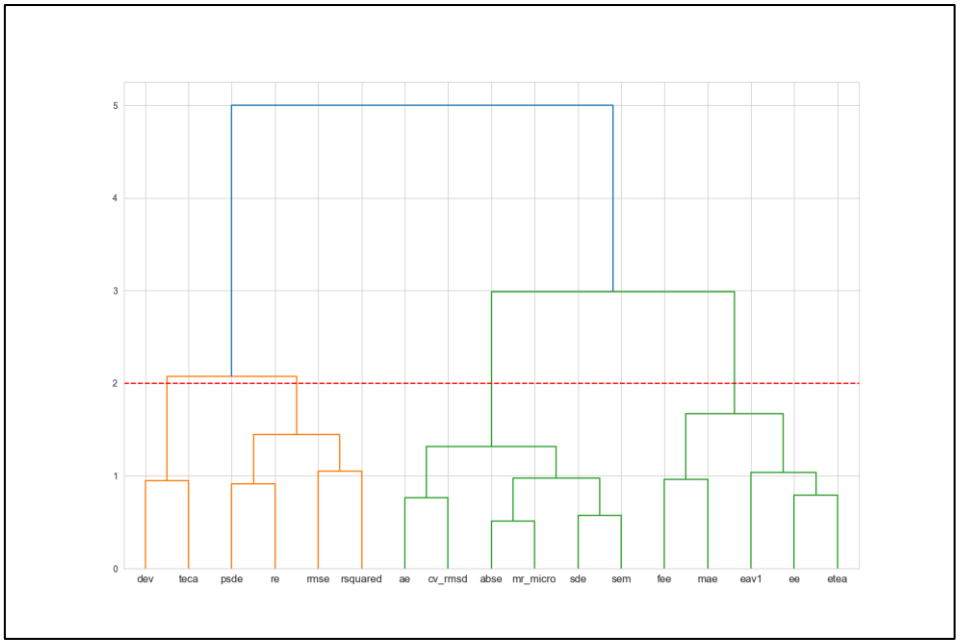


Figura 13: Dendrograma das métricas de estimação de energia calculadas sobre todos os objetos API de frigoríficos com congelador.

FR

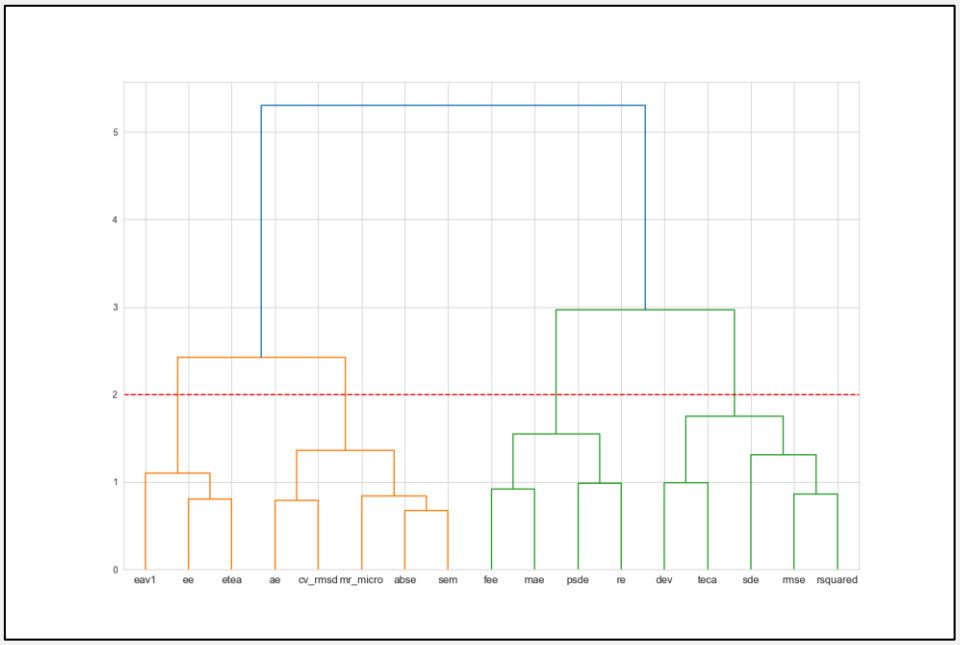


Figura 14: Dendrograma das métricas de estimação de energia calculadas sobre todos os objetos API de frigoríficos.

FZ

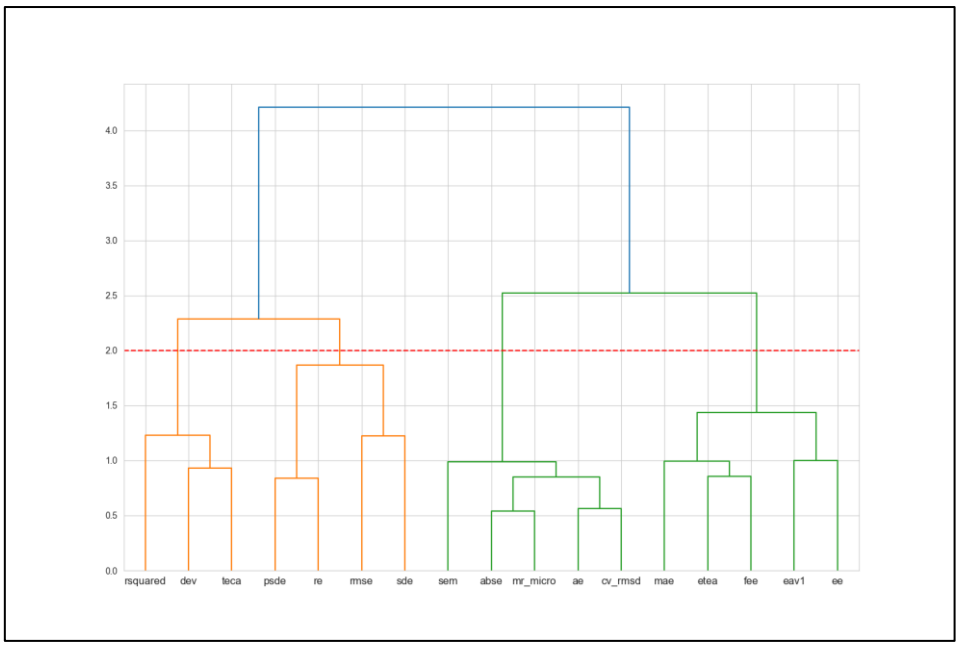


Figura 15: Dendrograma das métricas de estimação de energia calculadas sobre todos os objetos API de congeladores.

KT

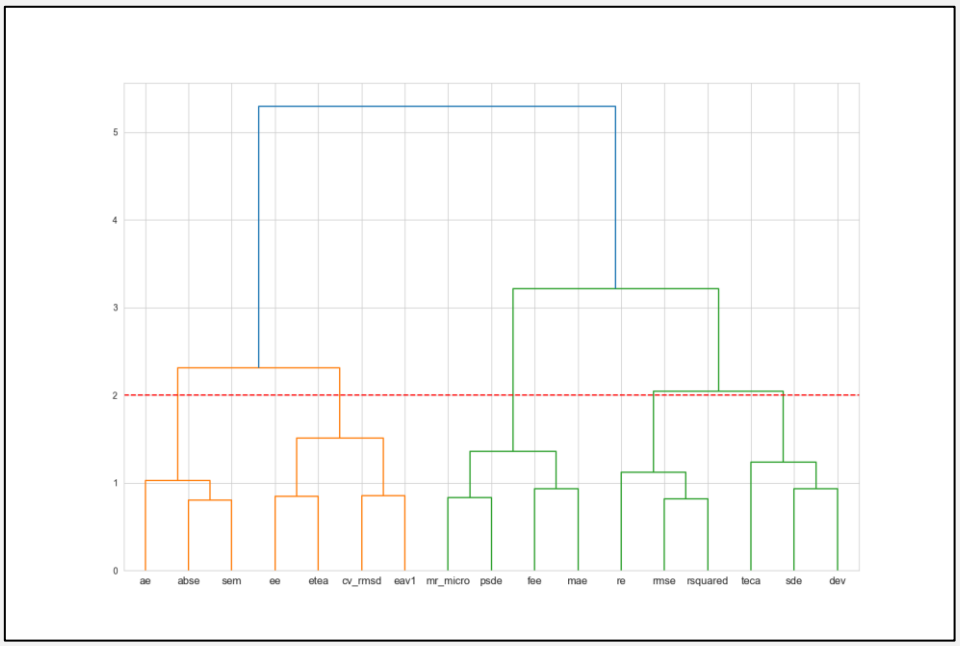


Figura 16: Dendrograma das métricas de estimação de energia calculadas sobre todos os objetos API de chaleiras.

MW

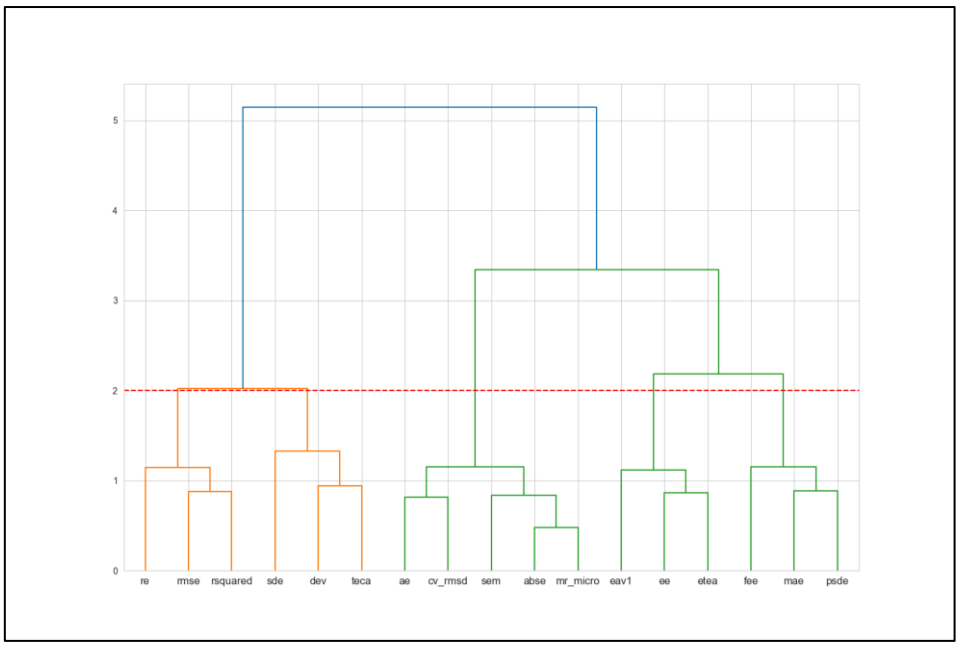


Figura 17: Dendrograma das métricas de estimação de energia calculadas sobre todos os objetos API de micro-ondas.

WM

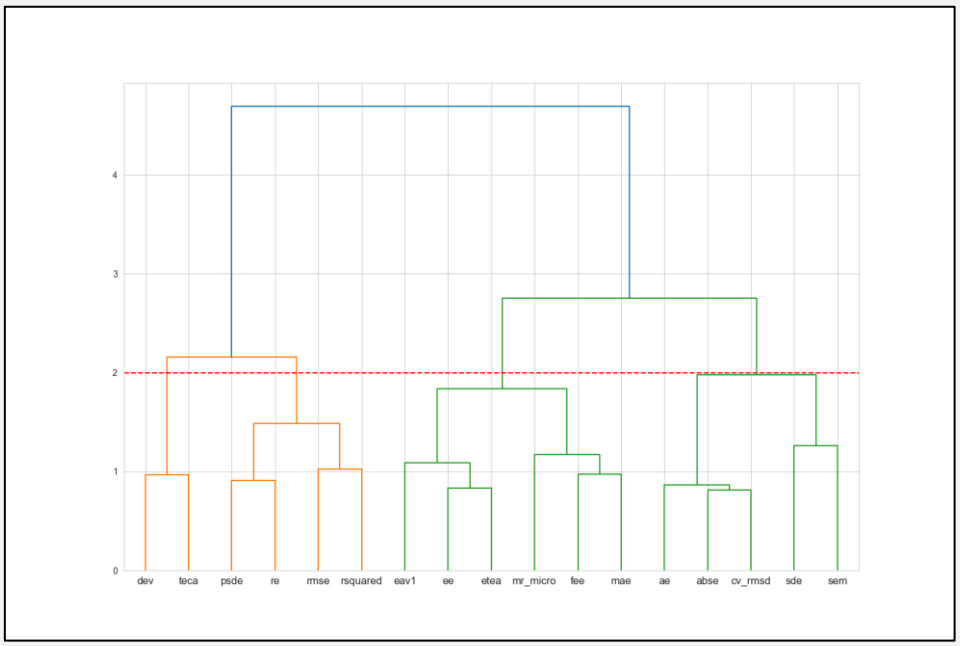


Figura 18: Dendrograma das métricas de estimação de energia calculadas sobre todos os objetos API de máquinas de lavar roupa.

TV

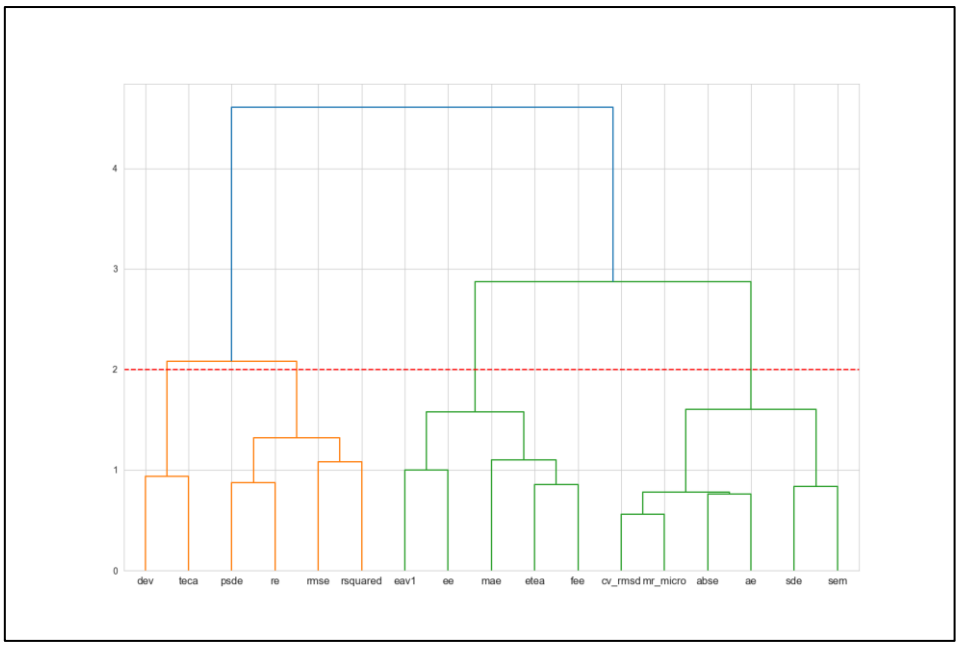


Figura 19: Dendrograma das métricas de estimação de energia calculadas sobre todos os objetos API de televisões.

Tabela 5: Dendrograma por aparelho.

Após tirada esta conclusão pude afirmar que realmente o aparelho influencia a seleção da métrica de desempenha na estimação de energia.

Capítulo III – Conclusão e Trabalho Futuro

3.1. Objetivos

Após a análise sobre os dados recolhidos posso afirmar que foram concluídos os objetivos, onde consegui provar a hipótese proposta, que um aparelho realmente influencia na seleção da métrica de desempenho para estimação de energia.

Os objetivos do projeto proposto foram também atingidos com sucesso, ganhei conhecimento sobre o NILM e sobre as diferentes ferramentas, métricas e conjuntos de dados do mesmo.

3.2. Limitações e Direção para Trabalho Futuro

Para simplificação deste estudo apenas foi considera a utilização de apenas uma métrica para todo o período e não diferentes métricas, como tal seria bom de futuro validar se a utilização de mais que uma métrica para o período da experiência pode influenciar esta hipótese.

As experiências utilizadas contam com um número limitado de algoritmos de desagregação de energia, e não incluem novos algoritmos de desagregação baseados em redes neurais profundas, como é o exemplo do Seq2Point ou Seq2Seq, como tal seria bom aprofundar este estudo utilizando mais algoritmos para obter mais informação sobre as métricas de desempenho estudadas.

Nesta análise foi apenas utilizado o método de análise de agrupamento hierárquico e seria bom ter mais algum método, por exemplo, a utilização do método de t-SNE (incorporação de vizinhos estocásticos t-distribuídos), para poder de outra forma poder validar a hipótese apresentada e se calhar avançar um passo mais e verificar se o algoritmo de desagregação também pode influenciar a escolha da métrica consoante o aparelho verificado.

Num trabalho futuro que deve ser realizado é entender qual a melhor forma de normalizar uma métrica, para percebermos como as mesmas devem ser consideradas e ordenadas. Além disso perceber quais as métricas adequadas para avaliar a estimação de energia de cada aparelho.