Universidad Politécnica de Madrid

Escuela Técnica Superior de Ingenieros Informáticos

Grado en Ingeniería Informática

Trabajo Fin de Grado

Aplicación de Reconocimiento de Imágenes para Monitorización de Carga No Intrusiva

Autor: Hernán Calvo Aguiar
Tutor(a): Esteban Garcia Cuesta

Madrid, Junio 2024

Este Trabajo Fin de Grado se ha depositado en la ETSI Informáticos de la Universidad Politécnica de Madrid para su defensa.

Trabajo Fin de Grado
Grado en Ingeniería Informática

Título: Aplicación de Reconocimiento de Imágenes para Monitorización de Carga No Intrusiva

Junio 2024

Autor: Hernán Calvo Aguiar
Tutor: Esteban Garcia Cuesta
Departamento de Inteligencia Artificial (DIA)
Escuela Técnica Superior de Ingenieros Informáticos
Universidad Politécnica de Madrid

Resumen

Aquí va el resumen del trabajo. Extensión máxima 2 páginas.

La Monitorización No Intrusiva de Carga, es una técnica que busca administrar los consumos de diferentes dispositivos eléctricos sin la necesidad de medir directamente cada uno de ellos.

El objetivo de este proyecto es desarrollar la pieza clave de este sistema. Una Inteligencia Artificial que permita identificar los dispositivos conectados y consumiendo a una red eléctrica, como podría ser una casa, un restaurante o una oficina.

El modelo a diseñar deberá ser capaz de identificar los dispositivos conectados a la red eléctrica, y estimar el tiempo de uso de cada uno. Para ello, se ha decidido hacer uso de CSPNet, un modelo potente, computacionalmente barato de entrenar

Este trabajo

Abstract

Abstract of the Final Degree Project. Maximum length: 2 pages.

Tabla de contenidos

Capítulo 1

Introducción

1.1. Motivación y Aplicaciones

Las técnicas de Monitorización No Intrusiva de Carga (en inglés NILM o Non Intrusive Load Monitoring) extraen el consumo de electrodomésticos a partir de una curva de consumo agregada. [1, pág. 1]. Dicho de otra forma. El monitoreo de carga no intrusivo es un sistema que infiere los consumos de diferentes dispositivos sobre una curva de consumo en la toma de corriente principal. Por ejemplo en la toma de corriente de una casa.

En Europa, un tercio de la energía eléctrica se ha generado mediante energías no renovables(12 % o 333TWh carbón, 17 % o 569TWh gas natural)[2] [3]. Aún siendo un descenso récord, para poder continuar reduciendo el consumo de energías no renovables, debemos desarrollar herramientas que nos permitan hacer un uso más inteligente de la generación de energía. Herramientas como la monitorización de carga no intrusiva pueden ayudarnos a detectar patrones de uso[4, pág. 11], para entonces poder desarrollar políticas que permitan optimizar la generación de energía.

Si, hipotéticamente, se redujese un 2 % la generación de electricidad a través de la combustión de carbón. Supondría una reducción de emisiones de 96.6 Millones de Toneladadas de CO2[1].

Realizar un monitoreo de carga no intrusivo es un problema complejo. Actualmente el área de soluciones NILM se encuentra en la fase de investigación y desarrollo. Por lo tanto. Mi objetivo para este trabajo es entrenar y analizar el rendimiento de una inteligencia artificial especializada en la clasificación de imágenes, transformando los datos de entrenamiento de series temporales a imágenes. Detallado en el **Capítulo 3**. Quiero ver cómo modificar el enfoque al problema afecta al rendimiento de la técnica en comparación con otras técnicas del estado del arte.

Siguiendo este objetivo y con el asesoramiento de Esteban García Cuesta, se de-

[1] $333 * 10^3 \text{GWh} * 290 \frac{tCO_2}{GWh}$ [5]

cidió en hacer uso de datasets clasificados para un entrenamiento supervisado; además de una arquitectura potente, compacta y computacionalmente baja en costes de entrenamiento.

1.2. Objetivos

Previamente al comienzo del desarrollo del proyecto, se establecieron una serie de objetivos a cumplir, dividiendo el objetivo final en pasos coherentes y concretos.

1. Investigación del problema NILM (Monitorización de Carga No Intrusiva) y soluciones actuales
2. Elección de dataset con el que se entrenará a la inteligencia artificial.
3. Análisis y calibración del dataset.
4. Desarrollo codificador para transformar las series de datos a imágenes siguiendo la codificación GAF
5. Preprocesado de dataset a través del codificador, para generar tanto las imágenes como sus etiquetas asociadas.
6. Instalación darknet CSPNet, el framework que contiene a CSPNet, la inteligencia artificial que se entrenará.
7. Entrenamiento del modelo. Una prueba de concepto del 10 % del dataset y modelo completo, con todo el dataset dedicado a entrenamiento.
8. Análisis del rendimiento del modelo con NILMTK, un toolkit que facilita la comparación del rendimiento con otros modelos.
9. Comparación del rendimiento con otros modelos

1.3. Alcance del proyecto

Al finalizar este proyecto, se busca tener un modelo capaz de realizar una monitorización de carga no intrusiva, además de una serie de progamas y herramientas útiles a la hora de realizar trabajos futuros en el área.

1.4. Estructura de la Memoria

La memoria de esta investigación tiene los siguientes capítulos.

- **Introducción:** Se introduce el tema, sus potenciales aplicaciones, motivación, alcance...
- **Fundamentos y Panorama Actual:** Este capítulo sirve para dar al lector una base de la teoría detrás de la tecnología, algoritmos y conceptos sobre los que se desarrolla el trabajo de fin de grado. Además de informar sobre el estado del arte en el área.

- **Desarrollo:** En este capítulo se detallan los pasos tomados para el desarrollo del modelo y las herramientas que utiliza. Entrenamiento, codificación, evaluación del modelo, etc.
- **Análisis de Impacto:** Este apartado recoge los resultados del modelo. Se compara su rendimiento con otros modelos y se habla brevemente del funcionamiento de los modelos contra los que se compara.
- **Conclusiones y Trabajo Futuro:** En este capítulo se recogen las conclusiones del trabajo y el trabajo futuro que puede hacerse para mejorar el modelo y sus partes.
- **Bibliografía:** Se recogen todos los recursos bibliográficos citados durante el transcurso de la investigación.
- **Anexo:** Detalla código e información de interés que no tiene cabida en el cuerpo del texto.

1.5. Planificación

Esta sección recoge la planificación realizada y aprobada por el tutor previa al comienzo del trabajo de fin de grado. Además de la planificación modificada en la entrega de la memoria de seguimiento.

1.5.1. Planificación Original

La planificación original contiene las siguientes tareas:

- **Dataset**
  - Investigación Dataset - Inicio: 23-feb-2024, Fin: 27-feb-2024
  - Elección Dataset - Inicio: 28-feb-2024, Fin: 29-feb-2024
  - Descarga Dataset - Inicio: 01-mar-2024, Fin: 01-mar-2024
  - Listado características Dataset - Inicio: 01-mar-2024, Fin: 01-mar-2024
  - Limpieza de casos no válidos - Inicio: 02-mar-2024, Fin: 06-mar-2024
- **Codificador**
  - Formación Desarrollo Kernel GPUs - Inicio: 02-mar-2024, Fin: 09-mar-2024
  - Elección lenguaje de desarrollo - Inicio: 10-mar-2024, Fin: 10-mar-2024
  - Diseño codificador - Inicio: 11-mar-2024, Fin: 18-mar-2024
  - Elección entorno de codificado - Inicio: 22-mar-2024, Fin: 24-mar-2024

- Preparación del entorno de desarrollo - Inicio: 22-mar-2024, Fin: 24-mar-2024
- Programación Codificador - Inicio: 22-mar-2024, Fin: 31-mar-2024
- Codificación - Inicio: 25-mar-2024, Fin: 01-abr-2024
- Sanity Checks - Inicio: 02-abr-2024, Fin: 04-abr-2024

- **Modelo**
  - Instalación Darknet Framework - Inicio: 05-abr-2024, Fin: 07-abr-2024
  - Porting de Darknet a ZLUDA - Inicio: 08-abr-2024, Fin: 10-abr-2024
  - Entrenamiento modelo del 10 % de los datos - Inicio: 11-abr-2024, Fin: 11-abr-2024
  - Entrenamiento modelo completo - Inicio: 10-may-2024, Fin: 18-may-2024
- **Análisis**
  - Análisis de rendimiento del 10 % - Inicio: 12-abr-2024, Fin: 19-abr-2024
  - Análisis de rendimiento del modelo completo - Inicio: 19-may-2024, Fin: 26-may-2024
  - Investigación otros modelos - Inicio: 27-may-2024, Fin: 04-jun-2024
  - Comparación otros modelos - Inicio: 20-abr-2024, Fin: 26-abr-2024
- **Memoria**
  - Entrega seguimiento - Inicio: 30-abr-2024, Fin: 09-may-2024
  - Redacción seguimiento - Inicio: 20-abr-2024, Fin: 29-abr-2024
  - Redacción Memoria - Inicio: 31-mar-2024, Fin: 29-may-2024
  - Preparación Presentación - Inicio: 30-may-2024, Fin: 06-jun-2024

1.5.2. Planificación Modificada

La planificación tuvo que modificarse. Las causas son una combinación de motivos externos y una perspectiva algo ingenua del tiempo que iba a ser necesario para realizar tanto la memoria como una serie de tareas. Lo primero que se ha hecho ha sido borrar todas las subtareas pertinentes al modelo del 10 %; estas tareas se refieren a un modelo entrenado con el 10 % de los datos, una pequeña prueba de concepto. El principal motivo para estas modificaciones ha sido la dificultad para programar un codificador en C++ y OpenCL, debido a la curva de dificultad que representa aprender un lenguaje para una nueva arquitectura y los retos que presenta interactuar con una base de datos con un motor relativamente lento. El segundo motivo, que no busca excusar el retraso en los objetivos,

si no dar una clara representación de los hechos. Es debido a mi contrato laboral a tiempo completo, que me resta gran parte del día, limitando el tiempo y energía que puedo dedicarle a este trabajo. Sin embargo, espero que el lector pueda apreciar que mi compromiso sigue siendo serio. Abajo puede observarse la lista de tareas y subtareas a partir de la cual se genera el diagrama de Gantt. Por último, se han corregido algunas fechas que: no cuadraban con las fechas de entrega, no cuadran con el trabajo realizado.

- **Dataset**
  - Investigación Dataset - Inicio: 23-feb-2024, Fin: 27-feb-2024
  - Elección Dataset - Inicio: 28-feb-2024, Fin: 29-feb-2024
  - Descarga Dataset - Inicio: 01-mar-2024, Fin: 01-mar-2024
  - Listado características Dataset - Inicio: 01-mar-2024, Fin: 01-mar-2024
  - Limpieza de casos no válidos - Inicio: 02-mar-2024, Fin: 06-mar-2024
  - Parseo de dataset a BBDD - 06-mar-2024
- **Codificador**
  - Formación Desarrollo Kernel GPUs - Inicio: 02-mar-2024, Fin: 09-mar-2024
  - Elección lenguaje de desarrollo - Inicio: 10-mar-2024, Fin: 10-mar-2024
  - Diseño codificador - Inicio: 11-mar-2024, Fin: 18-mar-2024
  - Elección entorno de codificado - Inicio: 22-mar-2024, Fin: 24-mar-2024
  - Preparación del entorno de desarrollo - Inicio: 22-mar-2024, Fin: 24-mar-2024
  - Programación Codificador - Inicio: 22-mar-2024, Fin: 30-abr-2024
  - Codificación - Inicio: 30-abr-2024, Fin: 10-may-2024
  - Sanity Checks - Inicio: 30-abr-2024, Fin: 12-may-2024
- **Modelo**
  - Instalación Darknet Framework - Inicio: 05-abr-2024, Fin: 07-abr-2024
  - Porting de Darknet a ZLUDA - Inicio: 08-abr-2024, Fin: 10-abr-2024
  - Entrenamiento modelo completo - Inicio: 12-may-2024, Fin: 18-may-2024
- **Análisis**

  - Análisis de rendimiento del modelo completo - Inicio: 19-may-2024, Fin: 26-may-2024
  - Investigación otros modelos - Inicio: 08-abr-2024, Fin: 14-may-2024
  - Comparación otros modelos - Inicio: 20-abr-2024, Fin: 26-abr-2024
- **Memoria**
  - Entrega seguimiento - Inicio: 21-abr-2024, Fin: 21-may-2024
  - Redacción seguimiento - Inicio: 01-abr-2024, Fin: 21-abr-2024
  - Redacción Memoria - Inicio: 08-abr-2024, Fin: 03-may-2024
  - Preparación Presentación - Inicio: 03-jun-2024, Fin: 10-jun-2024

Capítulo 2

Fundamentos y Panorama Actual

Capítulo dedicado a describir los fundamentos y el panorama actual del trabajo.

2.1. Monitorización de Carga No Intrusiva

Este concepto fue inventado por George W. Hart, Ed Kern y Fred Schweppe en el Instituto Tecnológico de Massachussets, en los años ochenta [6]. Fundados por el Electric Power Research Institute, podemos encontrar el proceso básico descrito en la patente estadounidense 4,858,141.

Una breve bibliografía de George W. Hart: Actualmente ejerce como escultor desde 2017. Tiene esculturas en exposición en Berkeley, Princeton, Cambridge, Duke, Brown... Cofundó el Museo de las Matemáticas y se ha retirado recientemente, trabajó en ciencia computacional, matemáticas, educación e investigación.

2.1.1. Modelado y Teoría

La técnica, descrita por Hart, Kern y Schweppe en su publicación modela los consumos en lo que nombran como "Modelo Total de Carga". El modelo total de carga depende de que dispositivos estén encendidos en un momento dado. Por lo que definen un 'proceso de cambio' $a(t)$ donde a es un dispositivo de consumo eléctrico en un instante de tiempo t. Suponiendo n dispositivos,$a(t)$ será un vector Booleano de n componentes:

$$a_i(t) = \begin{cases} 1, \text{si dispositivo } i \text{ encendido en } t \\ 0, \text{si dispositivo } i \text{ apagado en } t \end{cases}$$

Continúan modelando el sistema de potencia total $P(t)$ para un sistema de corriente alterna como el estado de un dispositivo a_i por su consumo P_i con un pequeño ruido o error e

$$P(t) = \sum_{i=1}^{n} a_i(t)P_i + e(t)$$

El criterio para determinar el valor de cada una de los dispositivos es entonces:

$$\hat{a}(t) = \arg\min_{a} \left| P(t) - \sum_{i=1}^{n} a_i(t) P_i \right|$$

Esto nos lleva a un problema computacionalmente imposible de resolver salvo mediante la fuerza bruta[6, pág. 4] y poco margen de mejora en el modelado para simplificar su resolución. De manera resumida, el problema trata en encontrar el número de dispositivos y el consumo de cada uno. Este es un problema que se beneficia de hacer uso de modelos heurísticos.

El concepto de NILM desde su introducción ha sido el método preferido por ingenieros e investigadores para la disgregación de consumo desde su introducción, debido a sus ventajas económicas y prácticas.[7, pág. 2, pár. 4]

En cuanto al origen de este método, se buscó categorizar los dispositivos dentro de tres grandes grupos, para poder estructurar los diferentes dispositivos y generalizar el modelado de dispositivos específicos. Estos grupos son:

1. **Señales de estados fijos** Siguen la arquitecturea de una máquina de estados finita, las transiciones entre estados son consumos exactos, al igual que los estados
2. **Señales transitorias** Esto son señales que no se encuentran en un espacio discreto que permita modelar su comportamiento como una máquina de estados. Un ejemplo podría ser un calefactor automático, que regula su generación de calor en función de la diferencia de la temperatura ambiente y la temperatura objetivo.
3. **Otras** Señales que no pueden categorizarse en ninguno de los dos casos anteriores. Por ejemplo, los cambios de dirección del motor de una lavadora en el proceso de lavado.

Cabe mencionar que los propios investigadores reconocieron que en el futuro los dispositivos eléctricos que siguiesen un diseño de máquinas de estados perderían precedencia. Ya que habría más 'dispositivos inteligentes'. Actualmente, una calefacción programable o un aire acondicionado programable están al alcance de cualquiera. En 2021, en España el 50 % de las viviendas en alquiler constan de aire acondicionado[8].

Y esto es sólo un dispositivo. Habrá cientos de modelos de cada dispositivo, decenas de marcas, cada dispositivo con sus características específicas. Un modelado exhaustivo sería prácticamente imposible de crear y mantener, ya que sólo para el mercado doméstico hay una cantidad desorbitante de dispositivos, combinaciones de dispositivos, ect.

Un ejemplo de sistema NILM

Un sistema NILM consta de las siguientes partes: Una fuente de datos (acometida principal). Un dispositivo de recogida de datos (puede ser de Potencia Activa Y

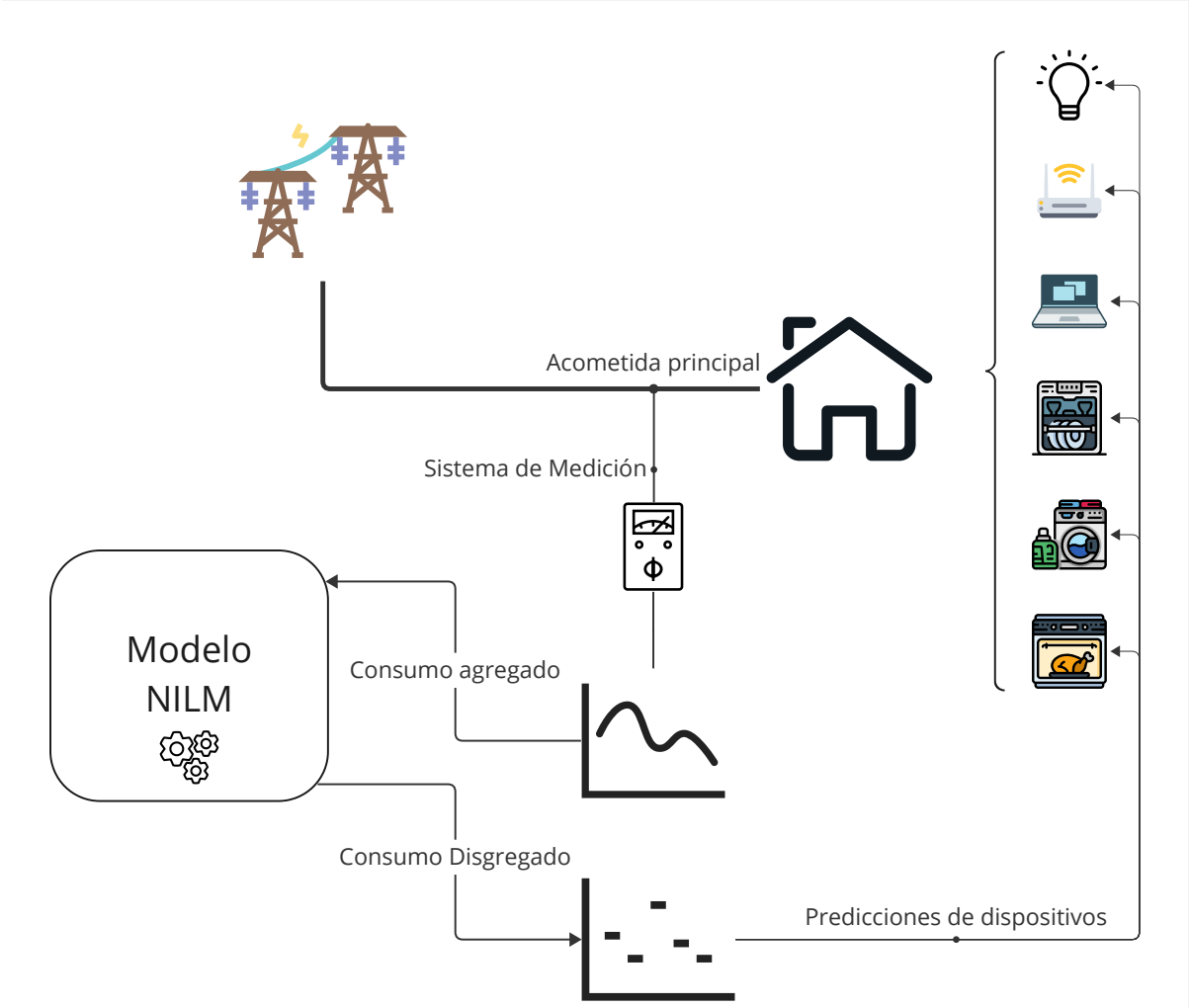


Figura 2.1: Un diagrama simple de un sistema NILM

reactiva, Voltaje e Intensidad, ect). El modelo predictivo. Y el sistema de consulta de datos, que se ha omitido en la Figura 2.1.

Debido a estos argumentos, la investigación de la monitorización no intrusiva se inclinó a los modelos heurísiticos. De los que voy a destacar los Modelos Ocultos de Markov, el Sparse Coding y finalmente las Redes Neuronales. Siendo estas últimas el foco principal del trabajo, y de este capítulo.

2.2. Modelos Ocultos de Markov

Si se sabe la teoría detrás de una cadena de markov, los modelos de Markov Ocultos resultan una dirección intuitiva a tomar, ya que estos representan la probabilidad de transición de un estado a otro. En esta subsección daremos una breve introducción a la teoría detrás de este modelo probabilístico.

La Cadena de Markov

Formalmente, una cadena de Markov se representa por los siguientes componentes:

- Un conjunto Q de N estados:

$$Q = q_1 q_2 ... q_n$$

- Una matriz de probabilidad de transición A donde para cada a_{ij} representando la probabilidad de transicionar desde el estado j al estado i.

$$A = a_{11} a_{12} ... a_{N1} ... a_{NN}$$

  Cumpliendo A la propiedad: $\sum_{j=1}^{n} a_{ij} \forall i$

- Una distribución de probabilidad inicial π sobre los est ados. Siendo π_i la probabilidad de que la cadena de Markov comience en el estado i.

$$\pi = \pi_1, \pi_2, ..., \pi_N$$

  Si un estado j tuviera $\pi_j = 0$ no podría ser un estado inicial (teniendo probabilidad 0). Además,π debe cumplir $\sum_{i=1}^{N} \pi_i = 1$

[9]

Identificando Datos Ocultos Con Cadenas de Markov

Una cadena de Markov resulta útil cuando tenemos que modelar un comportamiento en base a una secuencia de eventos observables. Estos eventos observables son elementos conocidos, pero la secuencia de estados del modelo para esos eventos observables son desconocidos.

El problema, como en NILM, es que los estados no son directamente observables. Estan *ocultos* y por tanto debemos ajustar la cadena para incluir una secuencia de posibles similitudes observadas.

Continuando la explicación de [9] en la subsección anterior:

- Para un elemento i se expresa como $b_i(o)$ donde o es derivado de un conjunto de posibles valores observables V.
- Dado:

$$Q = q_1 q_2 ... q_n$$

- Calculamos la probabilidad de una cadena de estados S de longitud T

$$S = s_1, s_2, ..., s_T$$

- Dado un conjunto de valores observados

$$O = o_1, o_2, ..., o_T$$

  para T observaciones.

Resumiendo: dados una serie de eventos observables, podemos inferir los estados que causaron estos eventos observables. Calculando la probabilidad de una cadena de eventos observables dada la probabilidad de una cadena de estados del conjunto, para todas las posibles combinaciones de la cadena los estados del conjunto.

Texto no definitivo, revisar

Siendo la fórmula:

$$\underset{O=o_1,o_2,...,o_T}{\arg\max}\ P(O = o_1, o_2, ..., o_T | S = s_1, s_2, ..., s_T)$$

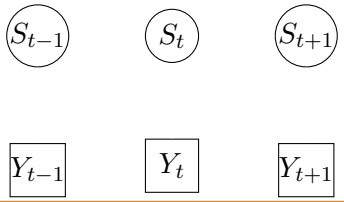
Este es el modelado que nos permite inferir de manera probabilística (fórmulas en detalle podrán encontrarse en el **Apéndice B**)

se añadirá el anexo para la entrega de la memoria final

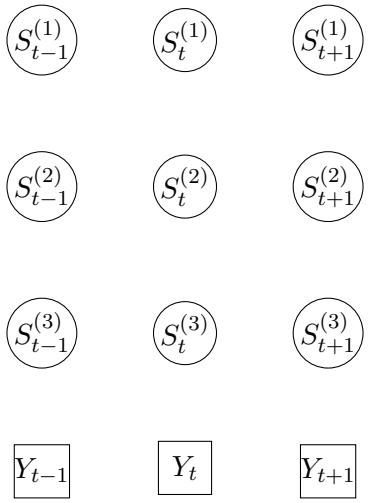
Modelos Ocultos de Markov

Explicadas brevemente las bases de los Modelos Ocultos de Markov, pasamos a la aplicación de este modelo estadístico en la disgregación de consumos.

Dado que un modelo de markov oculto infiere probabilísticamente una serie de estados, esto nos resulta útil para la disgregación de dispositivos. La forma de codificar diferentes estados de diferentes dispositivos se basa aumentando el ancho del modelo. Teniendo una matriz de estados S_i^j donde S es un conjunto de estados. Y las observaciones dependen de todos los estados en un momento dado. Podemos observar esto en el siguiente diagrama:



pendiente hacer que los vértices aparezcan



pendiente hacer que los vértices aparezcan

Observando el diagrama b) no resulta muy complicado imaginar la utilidad de este modelo. Si cada cadena representa un dispositivo modelado en diferentes estados, resulta un diseño muy apropiado para NILM. Los Modelos de Markov Ocultos se han tratado en la literatura como efectivos en comparación con otras técnicas de monitorización cuando el entrenamiento supervisado presenta una baja tasa de muestreo en el eje temporal de la obtención de datos [10]. Otras ventajas que presenta es su bajo coste computacional en el entrenamiento. Pero el uso de este tipo de elementos lo vuelven altamente susceptible a máximos locales[1].

En el caso de los Modelos Factoriales de Markov Ocultos (con siglas FHMM en inglés), presentan ciertas ventajas en comparación con los HMM [1]: La complejidad y el coste computacional de los los algoritmos de aprendizaje e inferencia son menores para los HMM. Sin embargo, tanto FHMM como HMM presentan limitaciones al volumen máximo de dispositivos simultáneos, incluso en casos como [11] donde se propone un nuevo algoritmo no supervisado. El algoritmo desarrollado (AFMAP) no presenta los problemas de alta complejidad y máximos locales, pero requiere de etiquetación manual después de la disgregación. Y presenta bajo rendimiento para dispositivos electrónicos y de cocina [1, pág. 5].

2.3. Redes Neuronales

Dado este tercer "verano de la Inteligencia Artificial". Este tercer verano ocurre en la década de 2010. En 2012 ocurre un avance técnico, unos años antes se creó el reto ImageNet. Un repositorio de más de un millón de imágenes de objetos a lo largo de más de mil categorías diferentes. [12] Resulta lógico que el número de artículos científicos proponiendo modelos que permitiesen resolver los retos que presenta NIML aumentase. Y, encontrar

En esta sección se tratan tres conceptos de las Redes Neuronales; se dará una introducción y se expondrán las aplicaciones pertinentes al tema del proyecto.

[1] Hidden Markov Models, o en español los Modelos Ocultos de Markov

Por motivos de brevedad, se asume que el lector tiene una cierta base en inteligencia artificial. En caso contrario, sugiero consultar las citas y el Apéndice A

2.3.1. Sparse Coding

Breve Introducción

El Sparse Coding consiste en reducir la activación de la capa oculta de las neuronas, forzando a la red neuronal a realizar un aprendizaje [13]. Su objetivo e encontrar un conjunto de vectores ϕ_i tal que podamos representar un vector x como una combinación lineal de estos vectores:

$$x = \sum_{i=1}^{k} a_i \phi_i$$

Este es un concepto que está fuertemente relacionado con los autoencoders y el aprendizaje PCA.

Que tienen como objetivo es que su salida sea igual a la entrada, pero limitando la capa oculta para forzar un aprendizaje. [14] Por ejemplo, supongamos que buscásemos utilizar un autoencoder para comprimir una imagen para enviarla por internet. Si no implementásemos restricciones, dado que el objetivo de la red neuronal es reducir la pérdida de información a 0; la salida del autoencoder sería la imagen, completamente idéntica. Para evitar esto en el caso de los autoencoders se introduce una capa oculta de menor tamaño en comparación a la capa de entrada como restricción para forzar un aprendizaje, al tener una capa oculta, la capa de salida (de igual tamaño a la de entrada) debe reconstruir"la entrada [14] .

Volviendo al Sparse Coding. Al tener un conjunto de vectores sobrecompleto, los coeficientes a_i ya no se determinan únicamente por el vector de entrada x. Luego se define una función de escasez[2]. Definimos esta función de escasez como: tener los menos componentes (a_i) distintos de cero o el mínimo de componentes no cerca de cero.

Aplicaciones

Tanto los autoencoders como el sparse coding han sido usados en la literatura NILM. sparse coding:[15]

Pendiente continuar investigando

[2]De ahí el "Sparse"; escasez en Español

2.3.2. Clasificadores de imágenes

Breve Introducción

Aplicaciones

2.3.3. Redes Recurrentes

Breve Introducción

Aplicaciones

2.4. Teoría detrás de la propuesta

2.4.1. Datasets para entrenamiento supervisado en NILM

2.4.2. CSPNet

2.4.3. Codificación GAF

2.4.4. Validación de Modelos: NILMTK

Capítulo 3

Desarrollo

TODOS LOS DIAGRAMAS SE HARAN CON MIRO, EXPORTADOS A SVG E INTEGRADOS

3.1. REFIT Dataset

3.1.1. Pros y contras del dataset

3.1.2. Características principales

3.1.3. Recogida de datos

Hablar de REFIT, sus características, los datos y los metadatos. Especial hincapié en cómo se recogieron

3.2. Parseado de REFIT a MYSQL

3.2.1. Motivación

3.2.2. Inconvenientes

3.2.3. Solución

Hablar de porqué lo parseamos a mysql, los inconvenientes (zona horaria e interrupciones en el parseo) y el diseño realizado para resolver estos problemas, además de los programas desarrollados (el primero que se rompía y el segundo que es traversecsv.cpp)

3.3. Codificación GAF

3.3.1. Justificación del uso de OpenCL

3.3.2. Inconvenientes

3.3.3. Solución

Hablar de porqué se decidió usar opencl, porqué se retrasó tanto esta parte del proyecto el diseño que era muy bueno pero muy complicado de implementar

3.4. Entrenamiento

3.4.1. ZLUDA y Darknet Framework

3.4.2. Monitorización

3.4.3. Inconvenientes

Apartado pendiente

Capítulo 4

Análisis de impacto

4.1. Personal

Mayor control de gasto, menos privacidad. Analisis de patrones de gastos y por tanto hábitos personaless/de la vivienda

4.2. Empresarial

industria: mayor control y más barato. (explicar pq) del gasto energético

4.3. Económico

mejora de la gestión energética lleva a mayor eficiencia, optimización y a mejor aprovechamiento de las energías renovables

4.4. Medioambiental

pro: aprovechar el consumo/optimizar los patrones de gasto teniendo mayor granularida y un modelo inferente como el que se ha desarrollado permite construir en un futuro herramientas predictivas de gastos que por tanto pueden aportar info valiosa para modelos más generales (usando estos y otros datos como los metereológicos, mercantiles, mercados, etc) para estimar la demanda esperada y el volumen de energía a generar.

Se recomienda analizar también el potencial impacto respecto a los Objetivos de Desarrollo Sostenible (ODS), de la Agenda 2030, que sean relevantes para el trabajo realizado (ver enlace)

Capítulo 5

Conclusiones y trabajo futuro

5.1. Conclusiones

5.2. Trabajo a Futuro

5.3. Alegato Final

Bibliografía

[1] A. Faustine, N. H. Mvungi, S. F. Kaijage y M. Kisangiri, «A Survey on Non-Intrusive Load Monitoring Methodies and Techniques for Energy Disaggregation Problem», *CoRR*, vol. abs/1703.00785, 2017.

[2] *Shedding light on energy in Europe - 2024 edition - Eurostat*. DOI: 10.2785/88627. dirección: https://ec.europa.eu/eurostat/web/interactive-publications/energy-2024#about-publication.

[3] S. Brown y D. Jones. (2024). European Electricity Review 2024, Ember, dirección: https://ember-climate.org/insights/research/european-electricity-review-2024/.

[4] J. Revuelta Herrero, Á. Lozano Murciego, A. Barriuso, D. Hernández de la Iglesia, G. Villarrubia, J. Corchado Rodríguez y R. Carreira, «Non Intrusive Load Monitoring (NILM): A State of the Art», jun. de 2018, págs. 125-138, ISBN: 978-3-319-61577-6. DOI: 10.1007/978-3-319-61578-3_12.

[5] J. C. Romero Mora, *Anyone know a scientific reference about converting kwh electricity to CO2 emission?*, dic. de 2014. dirección: https://www.researchgate.net/post/Anyone-know-a-scientific-reference-about-converting-kwh-electricity-to-CO2-emission#:~:text=In%202013%2C%20it%20was%200%2C29%C2%A0Kt%20CO2/GWh.

[6] G. Hart, «Nonintrusive appliance load monitoring», *Proceedings of the IEEE*, vol. 80, n.º 12, págs. 1870-1891, 1992. DOI: 10.1109/5.192069.

[7] C. Nalmpantis y D. Vrakas, «Machine learning approaches for non-intrusive load monitoring: from qualitative to quantitative comparation», *Artificial Intelligence Review*, vol. 52, n.º 1, págs. 217-243, jun. de 2019, ISSN: 1573-7462. DOI: 10.1007/s10462-018-9613-7. dirección: https://doi.org/10.1007/s10462-018-9613-7.

[8] *El 36% de las casas en España tiene aire acondicionado*, Idealista News, Accessed: 2023-04-15, jul. de 2021. dirección: https://www.idealista.com/news/inmobiliario/vivienda/2021/07/15/791442-el-36-de-las-casas-en-espana-tiene-aire-acondicionado.

[9] D. Jurafsky y J. H. Martin, *Speech and Language Processing*. 2023, cap. A. Hidden Markov Models, Copyright © 2023. All rights reserved.

[10] N. Miquey y E. Grover-Silva, «Non-Intrusive Load Monitoring of Single and Aggregated Profiles with a Hidden Markov Model», en *ENERGY 2021*, Valencia, Spain, mayo de 2021. dirección: https://minesparis-psl.hal.science/hal-03520223.

[11] J. Z. Kolter y T. Jaakkola, «Approximate Inference in Additive Factorial HMMs with Application to Energy Disaggregation», en *Proceedings of the Fifteenth International Conference on Artificial Intelligence and Statistics*, N. D. Lawrence y M. Girolami, eds., ép. Proceedings of Machine Learning Research, vol. 22, La Palma, Canary Islands: PMLR, abr. de 2012, págs. 1472-1482. dirección: https://proceedings.mlr.press/v22/zico12.html.

[12] H. Kautz, «The Third AI Summer: AAAI Robert S. Engelmore Memorial Lecture», *AI Magazine*, vol. 43, n.º 1, págs. 105-125, mar. de 2022. DOI: 10.1002/aaai.12036. dirección: https://ojs.aaai.org/aimagazine/index.php/aimagazine/article/view/19122.

[13] S. University. (). Sparse Coding. Accessed: 2024-04-20, dirección: http://ufldl.stanford.edu/tutorial/unsupervised/SparseCoding/.

[14] ——, (). Autoencoding Coding. Accessed: 2024-04-20, dirección: http://ufldl.stanford.edu/tutorial/unsupervised/AutoEncoders/.

[15] M. Xue, S. Kappagoda y D. K. A. Mordecai, *Energy Disaggregation with Semi-supervised Sparse Coding*, 2020. arXiv: 2004.10529 [eess.SP].

Anexos

Apéndice A

Glosario

Lista de términos y abreviaciones utilizados en el trabajo. Su sentido es facilitar y familiarizar al lector la comprensión de las abreviaciones.

- NILM: Non Intrusive Load Monitoring - Monitorización de Carga No Intrusiva.
- HMM: Hidden Markov Models - Modelos Ocultos de Markov.
- Sparse Coding: Codificación de escasez.
- CSPNet: Cross Stage Partial Network
- PCA: Principal Component Analysis - Análisis de Componentes Principales

Apéndice B

Fórmulas Matemáticas

B.1. Cadenas de Markov y Modelos Ocultos de Markov

Añadir todas las fórmulas relevantes para poder dejar constancia de la teoría matemática

B.2. Modelos Factoriales Ocultos de Markov

B.3. Sparse Coding

Apéndice C

Repositorios

C.1. Código Fuente

aquí irá mi repositorio, con el readme del repositorio

C.2. Repositorios externos

aquí irán los repositorios que he usado. Actualmente solo he usado el SafeMap