

UNIVERSIDAD NACIONAL DE SAN AGUSTÍN DE AREQUIPA

ESCUELA DE POSGRADO

**UNIDAD DE POSGRADO DE LA FACULTAD DE INGENIERÍA DE
PRODUCCIÓN Y SERVICIOS**



**“HERRAMIENTAS DE MINERÍA DE DATOS DE GRAN
ESCALA PARA APOYAR INVESTIGACIONES EN CAMBIO
CLIMÁTICO EN LA AMAZONIA PERUANA”**

Tesis presentada por el Bachiller
Carlos Eduardo Arbieto Batallanos,

Para la obtención del Grado de Maestro en
Ciencias: Informática
con mención en Tecnologías de Información.

Asesor: Mg. Alexander Victor Ocsa Mamani.

AREQUIPA - PERÚ
2018

Resumen

En el Perú existen diversas instituciones que dentro de sus actividades se encargan de recolectar información del clima a través de estaciones que cuentan con diversos sensores, estos miden los procesos naturales en diferentes escalas numéricas que nos permiten comprender la intensidad, frecuencia, volumen, etc. de estos fenómenos, este es el caso del IIAP (Instituto de investigaciones de la Amazonía Peruana) [IIAP, 2018], entidad que ha colaborado a esta investigación brindando una base de datos climatológica de 36 dimensiones, de una estación ubicada en el Departamento de Ucayali, y que presenta mediciones por dimensión cada 30 minutos, del año 2012 al 2016.

Esta base de datos reconstruida, ha sido almacenada en un repositorio de la amazonía peruana y publicada en la siguiente dirección ¹ donde se espera colaborar con otros investigadores que puedan descargar la información y publicar sus resultados; siendo a su vez fue tomada por este trabajo como caso de estudio para realizar una demostración de eficiencia de técnicas de minería de datos escalables sobre series temporales climáticas, con el objetivo de optimizar su desempeño en procesamiento en tiempo computacional y aprovechar las nuevas plataformas de hardware existentes, buscando acelerar el proceso de minería de datos a través del uso de consultas kNN (búsqueda del vecino más cercano) masivamente paralelos en GPU (Unidad de procesamiento gráfico).

Específicamente, se tomaron algoritmos de identificación de patrones frecuentes y datos irregulares (*outliers*) basados en consultas kNN y se re-implementaron en GPU usando consultas en GPU de kNN masivamente paralelos. Estas implementaciones en GPU (Unidad de procesamiento gráfico) permitieron ejecutar estudios experimentales sobre esta base de

¹www.repositorio.netlify.com

datos y así validar su desempeño usando un servidor(*workstation*) con GPU(Unidad de procesamiento gráfico) GTX1080 que resultó en un aumento de desempeño de hasta 10 veces.

Estas técnicas de minería de datos serían de utilidad para apoyar estudios en climatología, o para ser utilizada en otros análisis de series temporales. Este trabajo forma parte de los resultados del círculo de investigación de cambio climático que desarrolla la Universidad Nacional de San Agustín de Arequipa, la Universidad La Salle de Arequipa, la Pontificia Universidad Católica del Perú y la Universidad de la Amazonía.

Palabras clave: Minería de Datos, Computación de Alto desempeño, Recuperación de información, Big data, Patrones climáticos, Climatología, series temporales, GPU.

Abstract

In Peru there are several institutions that are responsible for collecting climate information through stations that have different sensors, they measure natural processes in different numerical scales that allow us to understand the intensity, frequency, volume, etc. of these phenomena, this is the case of the IIAP (Research Institute of the Peruvian Amazon) cite iiap, an entity that has collaborated in this research by providing a 36-dimensional climatological database of a station located in the Department of Ucayali, and that presents measurements by dimension every 30 minutes, from the year 2012 to 2016.

This reconstructed database has been stored in a repository of the Peruvian Amazon and published at the following address ² where it is expected to collaborate with other researchers who can download the information and publish their results; At the same time, this work was taken as a case study to perform an efficiency demonstration of scalable data mining techniques on climate time series, with the aim of optimizing its processing performance in computational time and taking advantage of new hardware platforms existing, seeking to accelerate the process of data mining through the use of massively parallel kNN queries in GPUs.

Specifically, algorithms for identifying frequent patterns and irregular data (outliers) based on kNN queries were taken and re-implemented in GPU using queries in massively parallel kNN GPUs. These implementations in GPU allowed to execute experimental studies on this database and thus validate its performance using a server (workstation) with GPU GTX1080 that resulted in an increase in performance of up to 10 times.

²www.repository.netlify.com

These data mining techniques would be useful to support studies in climatology, or to be used in other analyzes of time series. This work is part of the results of the climate change research circle developed by the Universidad Nacional de San Agustín of Arequipa, the Pontificia Universidad Católica of Perú and the Universidad de la Amazonía Peruana.

Listado de Símbolos

S	Conjunto de datos
N	Número de objetos en el conjunto de datos
$d()$	función de distancia
q	Objeto de consulta
r	Radio de consulta
k	Número de objetos más próximos en una consulta kNN
ε	Error relativo en la busca por similaridad aproximada
$(1 + \varepsilon)$	El factor de aproximación
$R_q(q, r)$	Consulta por rango
$kNN(q, k)$	Consulta a los k-vecinos más próximos
$AllkNN(k)$	Consulta a todos los k-vecinos más próximos
$(1 + \varepsilon) - R_q(q, r)$	Consulta por rango aproximada
$(1 + \varepsilon) - kNN(q, k)$	Consulta aproximada a los k-vecinos más próximos
$(1 + \varepsilon) - AllkNN(k)$	Consulta aproximada a todos los k-vecinos más próximos
$T = (T_1, T_2, \dots, T_n)$	Serie temporal

Agradecimientos

El presente trabajo va dirigido con una expresión de gratitud para mi asesor Alexander Victor Ocsa Mamani, maestros y compañeros de la maestría en ciencias informáticas de la escuela de posgrado de ingeniería de sistemas, que con nobleza y entusiasmo, influyeron con su conocimiento y ejemplo en mi; asimismo agradecer a mis padres, amigos, Alejandra Marquez y al centro de investigación CiTeSoft, por su apoyo incondicional, CONCYTEC, FONDECYT y CIENCIACTIVA por el apoyo económico y de materiales como también a la Universidad Nacional de San Agustín que hicieron posible la realización de este trabajo.

Índice

Resumen	II
Abstract	IV
Lista de Símbolos	VI
1. Introducción	1
1.1. Definición del problema	3
1.2. Antecedentes del problema	5
1.3. Propuesta	9
1.4. Justificación	10
1.5. Motivación	11
1.6. Objetivos de la investigación	13
1.6.1. Objetivo general	13
1.6.2. Objetivos específicos	13
1.7. Aportes	13
1.8. Estructura de la Tesis	14
2. Estado del Arte	15
2.1. Técnicas de Minería de Datos	16
2.1.1. Búsqueda de Patrones Frecuentes	16
2.1.2. Búsqueda de Datos Irregulares (Outliers)	18
2.2. Minería de Datos aplicada a datos climatológicos	20
2.2.1. Patrones Frecuentes	21
2.2.2. Datos Irregulares (<i>Outlier</i>)	22

2.2.3. Predicción sobre Datos climatológicos	23
2.3. Estudios climáticos	25
3. Marco teórico	28
3.1. Bases de datos	28
3.1.1. Series Temporales	29
3.1.2. Base de datos de NOAA	31
3.1.3. Base de datos de IIAP	35
3.2. Proceso de Minería de datos	36
3.3. Consultas por Similitud	42
3.3.1. Dominio de datos	43
3.3.2. Espacios Métricos	43
3.3.3. Espacios Multidimensionales	44
3.3.4. Tipos de Consultas de Similitud	46
3.3.5. Alta Dimensionalidad	52
3.4. Minería de series temporales	52
3.4.1. Representación de Series Temporales	53
3.4.2. Patrones frecuentes(motifs) y Datos Irregulares(outliers)	58
3.5. GPU en CUDA	62
3.5.1. Arquitectura CUDA	64
3.5.2. Aplicaciones	65
3.5.3. LSH	67
4. Desarrollo	69
4.1. Caso de estudio	69
4.1.1. Base de Datos de Ucayali	69
4.1.2. Repositorio de la Amazonia Peruana	85
4.2. Experimentos	91
4.2.1. Materiales y Métodos	92

4.2.2. Métricas para la evaluación del desempeño	93
4.2.3. Experimento 1: Descubrimiento de Patrones Frecuentes	94
4.2.4. Experimento 2: Descubrimiento de Outliers	97
5. Conclusiones	101
5.1. Principales Contribuciones	102
5.2. Discusión	104
5.3. Trabajos Futuros	104
Bibliografía	104
Anexos	114

Capítulo 1

Introducción

Los procesos naturales que se presentan en nuestra atmósfera conocidos como clima muestran un carácter dinámico y aleatorio, lo que los convierte en fenómenos complejos de comprender y cuantificar. Para estudiar los efectos que estos producen en nuestra realidad, es necesario caracterizar en detalle los cambios que se dan en las variables atmosféricas a diferentes escalas medibles, ya que es muy importante caracterizar las componentes de viento, precipitación, oleaje ,caudal etc. Estos procesos son necesarios para realizar diseños de infraestructuras, planes agrónomos, proyectos de prevención o para la definición de tecnologías de gestión de riesgo.

Estos información climatológica generalmente tienden a relacionarse con el tiempo, por lo que se definen como series de tiempo o series temporales, y para su estudio regularmente se emplean de modelos numéricos estructurados a partir de métodos numéricos o estadísticos para estudiar el clima en distintas escalas, se ha evidenciado en diversos estudios de investigación.

En el Perú existen diversas instituciones que dentro de sus actividades se encargan de recolectar esta información a través de estaciones que cuentan con diversos sensores, estos miden los procesos naturales en diferentes escalas numéricas que nos permiten comprender la intensidad, frecuencia, volumen, etc. de estos fenómenos, este es el caso del IIAP(Instituto de investigaciones de la Amazonia Peruana) [IIAP, 2018], entidad que ha

colaborado a esta investigación brindando una base de datos climatológica de 36 dimensiones, de una estación ubicada en el Departamento de Ucayali, y que presenta mediciones por dimensión cada 30 minutos, del año 2012 al 2016 con 61,417 registros.

Ya que es comprensible que se presenten situaciones atmosféricas, de energía o fallas, algunos sensores dejan de funcionar por periodos de tiempos, convirtiendo complicada la labor de recolectar información continuamente, este fue el caso que se presentó al recibir la información del IIAP, por lo que se debía presentar una solución a esta situación, en efecto, existen variadas técnicas y herramientas para completar conjuntos de datos actualmente, este es el caso de Pandas, una librería de Python utilizada pero el análisis e interpolación, de bases de datos, ya que es una de las herramientas más utilizadas y completas se decidió, hacer eso de esta para realizar la interpolación de la información faltante de la base de datos, se llevaron a cabo numerosas pruebas para su completación siendo seleccionada la interpolación lineal, ya que era la técnica que presentaba resultados más cercanos a la realidad, dado que en la interpolación de orden 2 o 3 deformaban de manera significativa la distribución de la data.

Esta base de datos reconstruida, ha sido almacenada en un repositorio de la amazonía peruana y publicada en la siguiente dirección ¹ donde se espera colaborar con otros investigadores que puedan descargar la información y publicar sus resultados; Siendo a su vez fue tomada por este trabajo como caso de estudio para realizar una demostración de eficiencia de técnicas de minería de datos escalables sobre series temporales climáticas, con el objetivo de optimizar su desempeño en procesamiento en tiempo computacional y aprovechar los nuevas plataformas de hardware existentes, buscando acelerar el proceso de minería de datos a través del uso de consultas kNN masivamente paralelos en GPU.

Específicamente, se tomaron algoritmos de identificación de patrones frecuentes y da-

¹www.repository.netlify.com

tos irregulares(outliers) basados en consultas kNN y se re-implementaron en GPU usando consultas en GPU de kNN masivamente paralelos. Estas implementaciones en GPU permitieron ejecutar estudios experimentales sobre esta base de datos y así validar su desempeño usando un servidor(workstation) con GPU GTX1080 que resultó en un aumento de desempeño de hasta 10 veces.

Estas técnicas de minería de datos serían de utilidad para apoyar estudios en climatología, o para ser utilizada en otros análisis de series temporales. El caso de estudio en sí fue el de encontrar patrones frecuentes y datos irregulares(outliers) en una base de datos interpolada de la Amazonia peruana, exactamente de Ucayali del año 2012 al 2016 brindada por el IIAP.

Este trabajo forma parte de los resultados del círculo de investigación de cambio climático que desarrolla la Universidad Nacional de San Agustín de Arequipa, la Universidad La Salle de Arequipa, la Pontificia Universidad Católica del Perú y la Universidad de la Amazonia.

Esta investigación es parte de las actividades del equipo de alto desempeño del proyecto : “Círculo de Investigación en Cambio Climático” financiado por Ciencia Activa, que busca proveer de herramientas de análisis de datos y aprendizaje profundo a las investigaciones que se realizarán con referencia al clima y al cambio climático como consecuencia de la contaminación ambiental.

1.1. Definición del problema

Una de las mayores preocupaciones a nivel mundial es la influencia que ha tenido y tendrá el cambio climático sobre diversos sectores socio-económicos y sistemas ecológicos,

El cambio climático incrementará la frecuencia de eventos climatológicos extremos, como las olas de calor, precipitaciones intensas, sequías, temperaturas extremas, entre otros.

El Perú es uno de los países más vulnerable al cambio climático por presentar repercusiones de fenómenos relacionados con el Niño, etc. Además, la situación se agrava porque el 34 % de la población tiene bajos niveles de ingreso y las economías regionales dependen en gran medida de actividades económicas sensibles a los cambios climáticos.

La investigación para el análisis del clima en la actualidad es un tema prioritario, nuestro país es uno de los últimos en sumarse a estas iniciativas, dado a lo costoso y complicado que es implementar estaciones de medición como conseguir profesionales especializados, en realidad son cuantificables las instituciones nacionales que se encargan de recolectar y organizar la información, que no se dan a basto y no consiguen documentar, dar seguimiento y analizar, todos estos fenómenos como sería idóneo.

El IIAP(Instituto de investigaciones de la Amazonía Peruana) [IIAP, 2018] con mucho esfuerzo ha logrado colaborar con esta investigación brindando una base de datos con información climática del departamento de Ucayali, del año 2012 al 2016, que aunque se encuentre incompleta se ha buscado llenar la información faltante con técnicas de interpolación óptimas para este tipo de casos. Con el objetivo de que se puedan implementar técnicas de identificación de patrones frecuentes y data irregular para analizar su integridad y descubrir información.

Cuando se trata de grandes cantidades de datos de alta dimensionalidad se pueda identificar como obstáculo el volumen de datos que se tendrían que procesar las soluciones en GPU han demostrado que pueden ser bastante óptimas, dependiendo de la situación, que otras secuenciales, dado a que se dividirá el trabajo de un solo núcleo(CPU) a miles de núcleos que propone la unidad gráfica de proceso(GPU), pero para lograr esta solución se

tendrían que re-implementar las técnicas buscando parallelizar sus operaciones, por este motivo no solo se busca que la técnica sea la más eficiente sino también sea la más simple, por lo que se busca que las técnicas cumplan ambas, dado a que mientras más compleja sea la solución más compleja será su implementación

Es así que para este caso de estudio identificamos como principal problema, la eficiencia de técnicas de minería de datos de identificación de patrones frecuentes y data irregular(outliers) sobre series temporales climáticas, sabiendo que al tratarse de un gran conjunto de datos nos enfrentamos a la situación de los tiempos de respuesta del desempeño de estas técnicas, ya que durante una experimentación un investigador requiere realizar varias pruebas, por lo que se busca demostrar que se puede optimizar la respuesta del tiempo computacional a través del uso de consultas kNN masivamente paralelos en GPU, por lo que se requiere re-implementar estas técnicas en GPU.

1.2. Antecedentes del problema

Villar [VILLAR et al., 2010] en su estudio “*Variabilidad espacio-temporal de las lluvias en la cuenca amazónica y su relación con la variabilidad hidrológica regional. Un enfoque particular sobre la región andina*” hace un análisis de la variabilidad de las precipitaciones en la cuenca amazónica en el período 1964-2003. basándose en 756 estaciones pluviométricas distribuidas en todos los países de la cuenca incluyendo datos de Bolivia, Perú, Ecuador y Colombia.

Aprovechando la disponibilidad de datos de precipitación de los países andinos hace posible completar estudios anteriores, subraya el impacto de la cordillera de los Andes sobre las lluvias. La mayores precipitaciones son observadas en regiones bajas expuestas a los vientos húmedos del este, y las de menor lluvia son registradas en las estaciones de al-

tura que se encuentra protegidas por las primeras montañas de los Andes de los vientos húmedos del este. Además, los regímenes de las precipitaciones son más diversificados en las regiones andinas que en el llano amazónico. logrando un análisis de la variabilidad espacio-temporal de las precipitaciones es estudiada mediante un Análisis de Componentes Principales.

Enfrentándose así a la escasez de fuentes de información en nuestro país, hace uso de otras fuentes de bases de datos para tentar analizar las precipitaciones en nuestra región, obteniendo los siguientes resultados siendo la variabilidad a largo plazo muestra una precipitación decreciente desde 1980 que es predominante en los meses de junio-julio-agosto y en septiembre-octubre-noviembre. Durante la temporada más lluviosa, la principal variabilidad se da en la escala de tiempo decadal e interanual. La variabilidad interdecadal está relacionada con los cambios a largo plazo en el Océano Pacífico, mientras que la variación decadal, que opone al noroeste y sur.

Vergara [Vergara et al., 2016] en su trabajo “*Eventos Hidrológicos Extremos en la Amazonía Peruana: Sistema de Alerta para la Previsión*” presenta un informe mensual, elaborado en el marco del observatorio ORE-HYBAM y es posible gracias al convenio inter-institucional entre la Autoridad Nacional del Agua y el Instituto Geofísico del Perú. Esta cooperación inter-institucional tiene como objetivo la elaboración e implementación del estudio en mención, con la finalidad de contar con un sistema estacional que permita prever los impactos de los eventos hidrológicos extremos en la sociedad de la Amazonía peruana.

Concluyendo que Durante el mes de abril 2016, se observó convergencias de flujo de humedad en el norte de la cuenca Amazónica y el suroeste de la misma (cuencas del Mambe de Dios y del Beni). Por otro lado, se observaron divergencia de flujos de humedad en la Amazonía peruana y en la región sureste de la cuenca amazónica. Hasta fines del mes de abril 2016, según lo mostrado por la fuente de datos del TRMMRT, se presentaron

anomalías positivas de precipitación la Amazonía boliviana (anomalías de 7 mm/día) y la parte alta de la cuenca de Madre de Dios. Por otro lado, gran parte de la Amazonía peruana presentaron anomalías de precipitación negativas que alcanzaron en promedio de -3 mm/día.

Siendo una de las pocas iniciativas de recolección de información y su vez presentando sus análisis en un repositorio nacional de resultados pluviales ubicado en la siguiente dirección www.http://repositorio.igp.gob.pe sin la posibilidad de descargar las bases de datos con las que se obtuvieron los resultados, esta solución se vuelve inaccesible para continuar analizando los datos y descubrir otro tipo de información relevante.

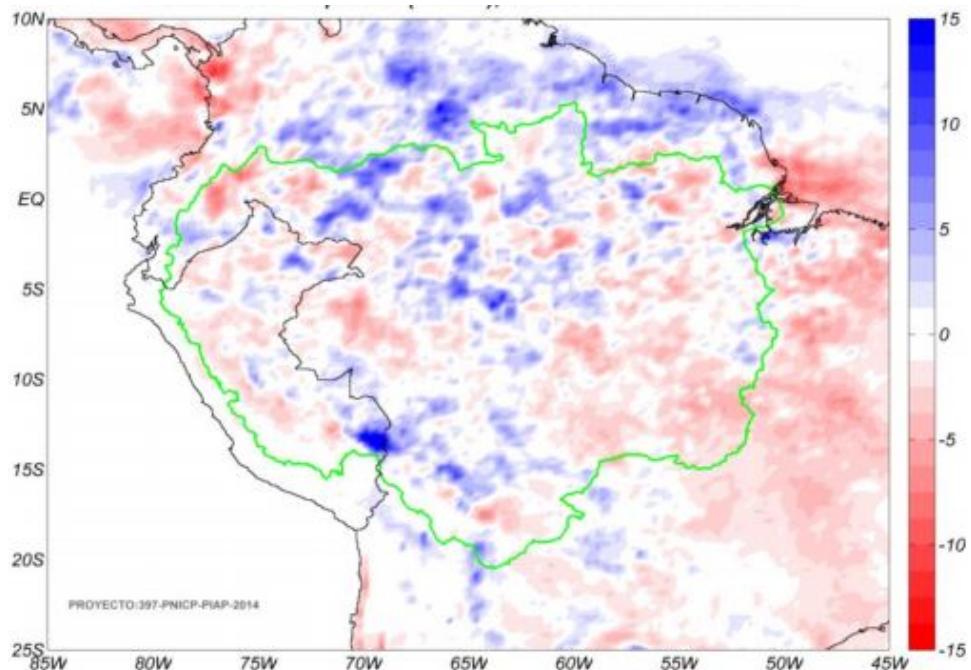


Figura 1.1: Anomalías de Precipitación Abril 2016 [Vergara et al., 2016]

Casas [Casas Luna and Mejía Marcacuzco, 2017] en su trabajo “*“MODELO AUTO-REGRESIVO DE PRIMER ORDEN APLICADO A LA PREDICCIÓN ANUAL DE CAUDALES EN LA AMAZONIA PERUANA: CUENCA DEL RÍO MAYO”* presenta como principal problema la frecuente insuficiencia de datos, sean de precipitación o, de manera más común, de caudales en nuestro País, en la mayoría de casos, se asume que el

futuro es estadísticamente similar al pasado y es sostenido por la hidrología.

siendo el objetivo de su trabajo desarrollar un modelo auto-regresivo de primer orden aplicado a la predicción anual (100 años) de caudales del río Mayo, mediante el análisis de datos históricos, generación de series temporales de caudales con distribución comparada normal, log normal y gamma; con el fin de lograr una predicción de caudales confiables.

Este trabajo es uno de los pocos que ya se encuentra proponiendo analizar series temporales climáticas para la predicción en nuestro país, siendo un precedente de que en nuestra región poco a poco se unen iniciativas con el objetivo de descubrir información valiosa a través de una cultura de recolección de información y un uso de técnicas algorítmicas adecuadas.



Figura 1.2: Localización del área de estudio y estaciones hidrológicas [Casas Luna and Mejía Marcacuzco, 2017]

Paredes [Paredes et al., 2015] en su artículo “*PATRONES TEMPORALES Y ESPACIALES DE LA PRECIPITACIÓN EN ARAGÓN DESDE 1950*” se analizan los patrones espaciales y temporales de la precipitación en Aragón desde 1950 y se analizan las tendencias del periodo 1981-2010 respecto a 1951-1980. Para el análisis se obtuvo una

serie regional de precipitaciones entre 1950 y 2010. Se comprueba la existencia de diferentes fases secas y húmedas, además de un descenso pluviométrico generalizado, utilizando técnicas de extracción de patrones temporales centrándose mas en los resultados que en las técnicas de descubrimiento de información concluye mostrando una clara tendencia hacia el descenso año tras año con porcentajes de cambio moderados y pocas diferencias territoriales.

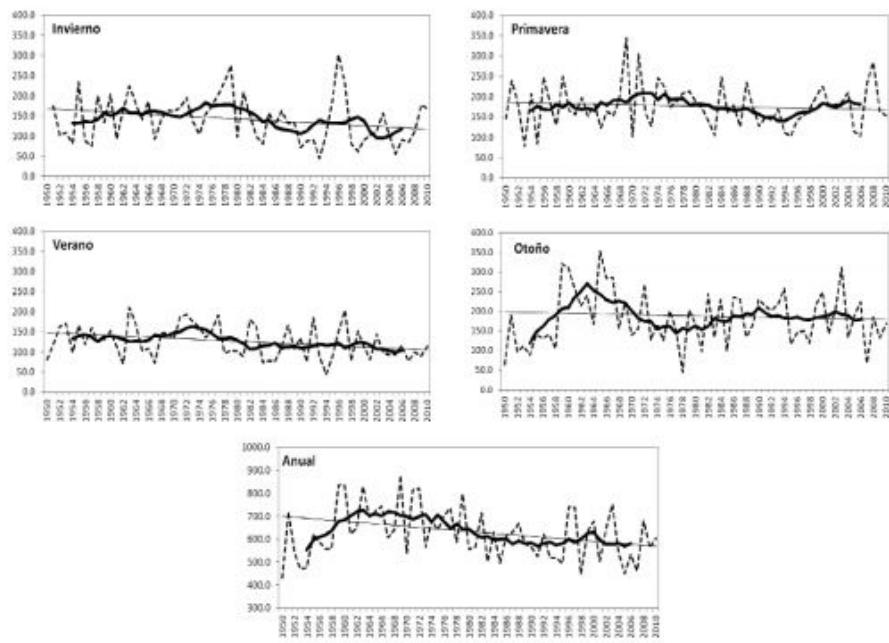


Figura 1.3: Evolución de las cantidades estacionales y anuales de precipitación. [Paredes et al., 2015]

1.3. Propuesta

En los procesos de minería de datos el costo computacional es una de las tareas mas importantes donde las investigaciones han prestado sus esfuerzos, siendo quizás la barreira mas tediosa a cruzar para lograr identificar patrones, buscar similitudes o encontrar información oculta.

Este proyecto de investigación propone demostrar la eficiencia de integrar la técnica LHS-

CUDA para optimizar técnicas de minería de datos de descubrimiento de *motifs* y *outliers*, a través de una comparación de su deficiencia en un workstation.

Para validar estos experimentos, se completo una base de datos climática como caso de estudio del departamento de Ucayali-Perú, de alta dimensionalidad y que presenta mediciones de 3 años, cada media hora.

1.4. Justificación

El desarrollo de esta investigación comprende una serie de actividades de un proceso de minería de datos de series temporales, comprendiendo todas sus partes, desde el pre-procesamiento que comprendió la interpolación, limpieza y ordenamiento de la base de datos, el análisis que comprendió en el descubrimiento de información de las series temporales, hasta la visualización, que comprende en una vista en ejes de coordenadas.

Para la interpolación se utilizo Pandas que es una biblioteca de software escrita como extensión de *NumPy* para manipulación y análisis de series temporales para el lenguaje de programación Python. En particular, ofrece estructuras de datos y operaciones para manipular tablas numéricas y series temporales.

Para el proceso de análisis de series temporales se utilizaron las técnicas de extracción y descubrimiento de patrones frecuentes y datos irregulares(*outliers*) dado a que desde su primera introducción, muchos estudios resaltan haber contribuido en su optimización dado a que esto asegura una extracción de información eficiente en un proceso de minería de datos. [Han et al., 2000, Agrawal and Srikant, 1995, Mannila et al., 1997, ?]

Estas técnicas aplican recuperación de información por similitud, por lo general crea

un universo de objetos y mide la distancia entre dos objetos, a través de diferentes tipos de consultas por similitud en este caso consultas por radio ($rangequeryRq(q, r)$) que tiene como objetivo recuperar los objetos similares a q que están dentro del rango de la consulta r , consultas de vecinos más cercanos KNN(búsqueda del vecino más cercano), que tiene como objetivo recuperar los k objetos más cercanos al objeto de búsqueda q o más formalmente los k vecinos más cercanos, y también nombrando las consultas de los k vecinos más cercanos que son consultas que buscan recuperar los k objetos más cercanos a todos los objetos de búsqueda.

Luego para demostrar sus tiempos computacionales del análisis de información en nuestro conjunto de datos establecidos en nuestro caso de estudio, se utilizó un servidor de aplicaciones de alta demanda (*workstation*) adquirido por el círculo de investigación de cambio climático, con las siguientes características un procesador INTEL CORE i7-6800K, tarjetas de vídeo NVIDIA TITAN X, CUDA CORES GPU TX1080, 16 GB de memoria RAM, y una motherboard ASUS RAMPAGE V EXTREME, este equipo especializado, nos asegura realizar eficientemente nuestros experimentos que consistieron en utilizar técnicas de patrones frecuentes y datos irregulares en conjuntos de datos de tiempo con el objetivo de demostrar la eficiencia de estas técnicas con diferentes arquitecturas para el procesamiento de datos en este caso, se realizaron pruebas en procesamiento secuencial CPU y en procesamiento paralelo GPU.

1.5. Motivación

“En la naturaleza al ser parte del universo se aprecia la existencia de diversos sistemas con diferentes niveles de complejidad, así al estudiar esta en el Planeta Tierra se encuentran ejemplos que demuestran esta afirmación : las interrelaciones entre la litosfera, la atmósfera, hidrosfera y la biosfera ; las interrelaciones entre sus componentes internos:

los suelos , las plantas y los animales; las propias interrelaciones existentes entre los organismos en la biosfera, entre otros. Todas las personas estamos implicadas en ella. Y es que todos comemos, nos vestimos, viajamos y nos comunicamos por medio de los productos de la ciencia. La conducta de las personas está marcada por el proceso de la ciencia. El proceso científico influye en el grado de autonomía y de confianza en sí misma que la persona posee, y en el grado en que persigue y alcanza soluciones racionales y eficaces a los problemas que se le presentan.” [Suárez and Pérez, 2017].

El desarrollo de la ciencia y la tecnología han traído no solo efectos positivos, sino también negativos para la sociedad, ha crecido el consumo de energías no renovables, lo cual hace que se aproxime el agotamiento de las fuentes convencionales como la madera, el carbón y el petróleo, deteriorando así el medio ambiente y contaminando el aire, el agua y las tierras, se destruyendo los bosques y reservas mundiales.

Nuestro país no es ajeno a esta situación, un tercio de viviendas utilizan combustibles para cocinar o para calefacción [sociodemográfico del Perú, 2008]. Los contaminantes producidos por la combustión tienen efectos negativos en la salud [Astete et al., 2009]. El aire exterior en Perú también está contaminado, a un alto nivel comparado con otros países latinoamericanos, según un reporte reciente de la Organización Mundial de la Salud (OMS). Se sabe que la materia particulada en el aire exterior está asociada con varias enfermedades crónicas. El Perú es también un país netamente minero, por lo cual está expuesto a la contaminación ambiental producida por la explotación minera [Astete et al., 2009]; esto implica un importante efecto negativo tanto de la contaminación de aire de exteriores y de interiores.

Teniendo en cuenta todos estos factores es importante resaltar la importancia del estudio de los efectos climatológicos en nuestro ambiente académico, sin dejar de mencionar que es un área prioritaria de investigación en nuestro país, por lo que se pretende contribuir

con una demostración de una optimización de herramientas de minería de datos basados en técnicas de extracción de patrones frecuentes y datos irregulares en bases de datos climáticas con el objetivo de apoyar investigaciones en cambio climático a expertos especializados en estudios del clima, brindando herramientas e información relevantes. [Han et al., 2011]

1.6. Objetivos de la investigación

1.6.1. Objetivo general

Comparar la eficiencia de técnicas de minería de datos basado en consultas kNN(búsqueda del vecino mas cercano) de series temporales en CPU y GPU para identificar patrones frecuentes y datos irregulares(*outliers*) en una base de datos de la Amazonia Peruana

1.6.2. Objetivos específicos

- Completar y Utilizar una base de datos climatológica de la Amazonia Peruana obtenida de IIAP (Instituto de investigaciones de la Amazonia Peruana) como caso de estudio
- Utilizar técnicas de identificación de patrones frecuentes y datos irregulares(*outliers*) en CPU y GPU basados en consultas KNN.

1.7. Aportes

Este trabajo tiene las siguientes contribuciones:

- Una base de datos de series temporales de 36 dimensiones interpolada de Ucayali.
- Una Comparación de la eficiencia de las técnicas de minería de datos en CPU y GPU.
- Un Repositorio con la base de datos de series temporales climatológica y las técnicas.
- Una publicación que se realizó durante la investigación.

1.8. Estructura de la Tesis

El presente trabajo se encuentra dividido en tres capítulos lo que van hacer explicados a continuación:

- El primer capítulo trata de las **consideraciones generales**, del trabajo que se tomó para el desarrollo de la investigación, así también como la problemática y los objetivos que se plantearon.
- En el segundo capítulo se desarrolla el **Estado del Arte** donde se da una revisión histórica del artículos y libros relevantes que ayudaron con su contenido a la realización de la investigación.
- En el tercer capítulo se especifica el **Marco Teórico** donde se explican los conceptos más importantes acerca de las técnicas y herramientas que se utilizaron en esta investigación.
- En el cuarto capítulo se va contemplar la **Experimentación** de acuerdo al caso de estudio y las técnicas seleccionadas.
- Y por último en el quinto capítulo se expone las **Conclusiones** donde se va a recopilar los resultados, análisis y discusión de los mismos.

Capítulo 2

Estado del Arte

En esta sección hablaremos sobre los trabajos previos de la literatura sobre el área de minería de datos de series temporales que contienen mediciones del clima.

Dado que el objetivo de esta tesis es la minería de patrones frecuentes en series temporales climáticas y de la detección de datos irregulares, decidimos dividir ésta sección en dos partes. La primera parte en la sección 2.1, hablaremos sobre la minería de datos para la extracción de patrones frecuentes en general, lo cuál se logra mediante las búsquedas por similitud exacta y aproximada, de las cuales hablaremos al respecto. Luego, hablaremos de algunos trabajos relacionados a la detección de datos irregulares.

La segunda parte que se encuentra en la sección 2.2, será dedicada a trabajos previos en la detección de patrones frecuentes y datos irregulares(*outliers*) pero específicamente en bases de datos del clima y/o en series temporales climáticas.

Finalmente en la sección 2.3, será dedicada a trabajos relacionados a los reciente estudios climáticos basados en análisis de series temporales.

2.1. Técnicas de Minería de Datos

La minería de datos es el proceso de extraer distintos tipos de información sobre un conjunto de datos. La extracción de información puede darse de manera automática o semi-automática, y consiste en extraer patrones mediante el análisis de *clustering*, detección de *outliers*, y minería de patrones secuenciales.

Las series temporales son una colección de observaciones en orden cronológico, que son largas en tamaño, altas en dimensionalidad y son actualizadas con frecuencia. Éste es el caso de los datos climatológicos, pues abarcan observaciones temporales. En este ámbito se investiga analizar la similitud entre series, la búsqueda de sub-secuencias o la búsqueda de patrones frecuentes. A diferencia de casos tradicionales, en series temporales se busca hacer consultas por similitud de manera aproximada y no exacta [Fu, 2011].

El objetivo final de las búsquedas por similitud, ya sean exactas o aproximadas es la minería de los datos, es decir el descubrimiento de información oculta o conocimiento de las series temporales. El descubrimiento de patrones frecuentes es el caso de minería más común y la *clusterización* es el método de agrupamiento más común. Otros tipos de minería incluye clasificación, minería de reglas y resúmenes.

A continuación vamos a hablar sobre dos tipos de minería o extracción de información: La búsqueda de patrones frecuentes en la data (2.1.1) y de la detección de *outliers* (2.1.2).

2.1.1. Búsqueda de Patrones Frecuentes

Como mencionamos anteriormente, la búsqueda de patrones frecuentes es una tarea demandada en la minería de información, y muchas trabajos se han basado en técnicas de agrupamiento o clusterización para lograrlo. En [Fu, 2011], se presentan una compilado

de técnicas para el descubrimiento de patrones frecuentes o también llamados “motifs”.

Se muestran diversos abordajes para resolver este problema, por ejemplo técnicas de clusterización automáticas como Mapas auto-organizables (SOM) [Van Hulle, 2012], otras técnicas basadas en el *fuzzy c-means* [Bezdek et al., 1984] (FCM), en donde la similitud entre dos series se basa en comparar su formas, las cuales cambian según el cambio de amplitud y el cambio temporal. También existen modelos ARMA (Media móvil autoregresiva) [Marple and Marple, 1987] que usan *k-means* para modelar su data. Usando un proceso de recorte o “clipping”, la data se discretiza en secuencias binarias según la media. Por otro lado, los Modelos Ocultos de Markov [Eddy, 1996] (HMM) se usan también para el agrupamiento de series temporales. Para el descubrimiento de patrones periódicos podemos utilizar técnicas para el procesamiento de señales como la Transformada Rápida de Fourier (FFT) [Brigham and Brigham, 1988]. Existe un sub-problema en el descubrimiento de *motifs*, el cuál busca encontrar los motifs que se muestran más frecuentemente en una serie temporal P , los cuales toman el nombre de “motifs más significativos”. El algoritmo para hacer esto toma el nombre de “K-motif” [Chiu et al., 2003a].

Existen técnicas actuales para la búsqueda de patrones frecuentes que se basan en algoritmos genéticos como en [Koshti and Patel, 2018]. Los autores mencionan que una de las ventajas de trabajar con algoritmos genéticos es que para descubrir patrones desarrollan búsquedas globales en la data y su complejidad es menor que la de otros algoritmos, ya que realiza un enfoque de búsquedas “greedy” o codiciosas, que si bien no siempre obtienen las mejores soluciones, son mucho más rápidas.

Existen también enfoques para la búsqueda de patrones frecuentes mediante la indexación de la información en una estructura de datos, tal es el caso de [Borah and Nath, 2018] que menciona alguna aplicaciones y mejoras al FP-Tree. Esta estructura se considera muy importante en el área de minería de patrones frecuentes. Una de las características del FP-Tree es que es una estructura de datos compacta, ya que evita que se tenga que

recorrer repetidamente su estructura y se basa en una metodología conocida en algoritmos llamada “divide and conquer”, que en español significa: “divide y vencerás”. El FP-Tree contiene en su estructura todo lo que necesitamos saber de la base de datos que estamos indexando para poder minar los patrones frecuentes.

En algunos trabajos como el de [Kohonen, 2008], comparan el FP-Tree con otro tipo de técnica muy estudiada para patrones frecuentes llamada Apriori [Perego et al., 2001]. Para ellos el FP-Tree es significativamente más eficiente que el algoritmo Apriori. Entre algunas comparaciones que hacen entre ambos algoritmos se dan las siguientes conclusiones: El algoritmo Apriori trabaja bien excepto cuando existen muchos patrones frecuentes, y cuando los patrones son muy largos. En éste caso el FP-Tree trabaja bien en éstos escenarios.

2.1.2. Búsqueda de Datos Irregulares (Outliers)

El problema de detección de *outliers* ha sido estudiado desde hace tiempo y diversos libros y artículos(*papers*) han sido publicados clasificando las distintas técnicas y abordajes que se han tenido para este problema [Ramaswamy et al., 2000, Ben-Gal I., 2005, Domingues et al., 2018].

En [Kantardzic, 2011], se propone dividir las técnicas para detección de *outliers* en: técnicas gráficas o de visualización, técnicas estadísticas, técnicas basadas en distancias y técnicas basadas en modelos.

- Las técnicas de visualización incluyen los “*Boxplot*”, “*Scatter plot*” y el “*Spin plot*” en 1, 2 y 3 dimensiones respectivamente, los cuales nos dan una muestra visual de la distribución de la data.
- Los métodos estadísticos por su parte tratan de crear un modelo que se adapte a

toda la data y marca como *outliers* aquellos elementos que se desvían del modelo. Algunas métricas que se usan son media y la desviación *estandard*. En método de *Grubb*, se mide un valor “Z”, la cuál es la diferencia entre el valor medio de la data y el valor en consulta, y si ya diferencia es mayor a un rango, es considerado *outliers*.

- Los métodos basados en distancias no necesitan un conocimiento previo de la distribución de la data. Solo necesitamos saber la distancia de un punto a sus vecinos para decidir si se trata de un dato irregular. Por ejemplo, si estuviéramos tratando con números, calcularíamos todas las distancias entre ellos, y tomaríamos como *outliers* aquellos que superen un límite determinado.
- Las técnicas basadas en modelos, simulan la manera en que los humanos detectan datos irregulares. Podemos mencionar algunas técnicas como *k-nearest Neighbor*, agrupamiento espacial basado en densidad, etc, las cuales son técnicas de agrupamiento pero que pueden adaptarse para resolver el problema de los *outliers*.

Posteriormente, observamos que el trabajo de [Kriegel et al., 2010], brinda una clasificación de métodos más extensa a la anteriormente mencionada y le agrega algunas otras técnicas como son: técnicas basadas en profundidad y técnicas basadas en densidad.

- Las técnicas basadas en profundidad consisten en crear distintas capas de “convex hull” hasta abarcar toda nuestra data, de tal manera que los *outliers* se encontrarían en las capas mas externas. Algunos algoritmos que se basan en esta idea son: ISODEPTH y FDC.
- Otras técnicas basadas en densidad, busca comparar la densidad alrededor de un punto con la densidad de sus vecinos. Una de éstas técnicas es el “*Local Outlier Factor*” [Chen et al., 2010], el cuál tiene otras variantes como el “*Local outlier correlation integral*”, el “*Connectivity-based outlier factor*” o el “*Influenced Outlierness*”.

En trabajos actuales como [Domingues et al., 2018], se mencionan algunos métodos probabilísticos para la detección de *outliers* como análisis probabilístico de componentes principales [Tipping and Bishop, 1999] (PPCA) y Detección de anomalías por mínimos cuadrados [Quinn and Sugiyama, 2014].

A pesar de que el aprendizaje de máquina es un área que se encuentra de moda en la actualidad, existen soluciones para la detección de *outliers* desde hace muchos años utilizando éste enfoque. Por ejemplo, en [Williams et al., 2002], se propone una interesante uso de una RNN (Red neuronal replicadora) para detección de *outliers*. Aquí las entradas de la red son iguales que las salidas, entonces la red trata de reproducir los patrones de los datos de entrada. Por consiguiente, aquellos patrones que representan *outliers* tendrían un mayor error de reproducción. A su vez en [Marsland et al., 2002], se hace el entrenamiento de una red *Grow When Required* (GRW) que es adaptativa ya que nuevos nodos y aristas son introducidas para ajustarse mejor a los datos.

2.2. Minería de Datos aplicada a datos climatológicos

Cómo vimos en la sección anterior, existen muchas técnicas de minería de datos, las cuales tienen un gran rango de aplicabilidad para la detección de patrones en cualquier tipo de data ya sea, imágenes, documentos, videos, etc. Sin embargo, existen muchos retos sobre el clima que pueden ser resueltos con minería de datos (*data mining*), haciendo algunas variaciones sobre las técnicas espaciales y espacio-temporales. Además en datos climatológicos, se deben considerar los cambios estacionales, que son ignorados en técnicas tradicionales [Ganguly and Steinhaeuser, 2008]. Un trabajo que resalta la diferencia entre las técnicas tradicionales de minería de datos sobre técnicas para el análisis de datos espacio-temporales en series temporales de clima es el de [Faghmous and Kumar, 2014].

En esta sección hablaremos de los trabajos sobre minería de datos en datos climáticos,

dividiéndolos en tres grupos: Técnicas para la detección de patrones frecuentes (2.2.2), técnicas para la detección de datos irregulares o “*outliers*” (2.2.2) y predicción (2.2.3) basada en datos anteriores.

Dentro de los trabajos que se dedican al estudio de la data climática podemos encontrar muchas variantes, por ejemplo, algunos se dedican al análisis de datos ambientales como la lluvia, precipitación, humedad, temperatura, etc. Algunos otros estudios se centran en analizar la data de las emisiones humanas como el CO₂, entre otros.

2.2.1. Patrones Frecuentes

Desde hace muchos años se han propuesto técnicas para descubrir patrones. Algunos trabajos iniciales comienzan proponiendo cosas simples, como el caso de la propuesta de [Singh and Stuart, 1998]. Se busca hacer predicción de nuevos registros haciendo búsquedas por similitud de datos anteriores en la base de datos de la Competencia en Series Temporales de Santa Fé. La premisa es que las estructuras actuales deben ser relacionadas con estructuras antiguas para generar una predicción. Lo interesante, es que para hacer búsquedas rápidas se convirtieron las series temporales a datos binarios donde el 0 y el 1 representaban la dirección de cambio en el valor de la serie temporal.

En el caso de [Kumar et al., 2001] se usa k-means, la cual es una técnica de agrupamiento que se usó para descubrir relaciones espacio-temporales entre datos de temperatura y precipitación de “Earth Science”. Antes del agrupamiento, se realizó un proceso para eliminar cambios estacionales fuertes producidos por el fenómeno del niño o tendencias de calentamiento global. En [Steinbach et al., 2003], también se usan métodos de agrupamiento de índices de clima. Métodos más actuales como el de [Sridevi et al., 2018], utilizaron una técnica más refinada de *k-means* llamada *K-Harmonic means* (KHM) para el agrupamiento, el cual es menos sensible a la inicialización y sirve para encontrar secuencias de patrones frecuentes, tanto en el conjunto de datos de electricidad de Australia

como en el conjunto de datos climatológico de Tamilnadu.

Anteriormente ya mencionamos la técnica de *k-Nearest Neighbour* (kNN), y este es un método de aprendizaje de máquina que no necesita suposiciones de la data, además no necesita de mucho entrenamiento, y sirve para buscar los k elementos más cercanos dentro de un conjunto de datos, dado un elemento de búsqueda. En trabajos recientes [Jakhar et al., 2018], se muestran que técnicas como kNN(*k-Nearest Neighbour*), bosques aleatorios y algoritmos *Adaboost*, se usan en análisis de datos climatológicos.

2.2.2. Datos Irregulares (*Outlier*)

En ésta sección hablaremos sobre la detección de datos irregulares en la data climática. Un dato irregular es también llamado *outlier*, y en el resto de éste proyecto se les llamarán indistintamente por los dos nombres. Estos representan datos que son muy desviados de los valores frecuentes o comunes de nuestra data. Es necesario saber identificar un dato irregular, ya que, tomando como ejemplo los datos climáticos que estamos presentando, un *outlier* podría representar una medición errónea producto de un error en los instrumentos de medición y es bueno eliminarlos para quitar el “ruido” de nuestros datos.

En [Estévez et al., 2018] se hace un estudio sobre la calidad de data metereológica ya que es importante para garantizar la confiabilidad de las aplicaciones que usan esta data como variable de entrada. Dentro de la calidad de la data, se puede decir que es importante la detección y eliminación de datos irregulares para asegurar la calidad de la data. La manera que este trabajo utiliza para la detección de estos datos es mediante la comparación de los datos de una estación específica con sus estaciones vecinas para validar los propios. Las estimaciones son hechas usando distintos pesos para darle mayor o menor importancia a cada estación dependiendo de la distancia entre las estaciones o del coeficiente de correlación.

Este tipo de técnica de validación se le llama en la literatura como “*tests* de consistencia espacial”. Los experimentos se corrieron sobre los datos de la Red de Información Agroclimática de Andalucía (sur de España).

Uno de los trabajos recientes en detectar datos espacio-temporales irregulares del CO₂ es el de [He et al., 2018]. Los autores analizan una base de datos del 2009 al 2015 del sistema “*Carbon Tracker*” y hacen un pre-procesamiento de la data para eliminar cambios continuos y extremos en las mediciones usando una función que es el resultado de una función lineal y una función periódica. Luego, mediante el cálculo del $Z \text{ score}(i, j, t)$ espacio-temporal, el cual utiliza la media y la desviación estándar, aquellos valores que se muestran fuera del rango normal de los valores con alta probabilidad, son considerados como irregulares.

2.2.3. Predicción sobre Datos climatológicos

Otra de las áreas de interés dentro de la minería de datos climáticos es la predicción. Las predicciones se hacen utilizando datos recolectados del pasado, y mediante distintas técnicas logramos predecir datos no vistos anteriormente. Los estudios de predicción son importantes, ya que ayudan a prevenir cambios climáticos para poder actuar con anticipación ante ellos.

En algunos trabajos como en el de [Olaiya and Adeyemo, 2012] se utiliza la inteligencia artificial para predicción del clima para predecir temperatura, lluvia, evaporación y velocidad del viento sobre la data del aeropuerto de Ibadan de 10 años. En este caso sólo fue necesaria una red neuronal artificial (ANN) simple en conjunto con un árbol de decisión. En un árbol de decisión, cada nodo representa un test sobre un atributo, y cada rama la salida para cada test. Finalmente, los nodos hoja representan las clases finales.

Por otros lado, como mencionamos anteriormente, la técnica kNN(*k-nearest neighbor*)

bor), no sólo se utiliza para la detección de patrones frecuentes, pero también es ser utilizado para predicción en datos climáticos como veremos a continuación. Tal es el caso de [Aftab et al.,], que hace una comparación de técnicas predictivas de minería de datos tales como Máquinas de Vectores Soporte (SVM), Naive Bayes (NB),kNN , Árboles de Decisión (J48) y Perceptrón Multicapa (MLP), para hacer la predicción de lluvia, basado en datos pluviales. Como es común en minería de datos, los autores utilizaron técnicas de pre-procesamiento como la media aritmética para los datos faltantes, y normalización para limitar todos los valores dentro de un intervalo específico. Los autores utilizaron las implementaciones de algoritmos de minería de datos de Weka [Mark Hall, 2009].

En [Sun and Trevor, 2017], una combinación de varios modelos kNN se utilizan para la predicción de hielo en el río “Smoky River at Watino.^{en} Canadá. Y en [Bannayan and Hoogenboom, 2008], se intenta predecir el clima diariamente usando la media de los *k* días más cercanos para el día que se quiere predecir usando también kNN.

Usualmente la media aritmética mencionada en el trabajo anterior para la completación de datos faltantes, no es muy eficiente cuando la cantidad de datos es muy grande. Esto suele ser algo común en datos climáticos, ya que en nuestro proceso de búsqueda de bases de datos sobre las cuales elaborar este proyecto de tesis, encontramos que muchas de las bases de datos climáticas tenían meses con mediciones faltantes. Para esto otros trabajos exploran alternativas para la completación de datos faltantes antes de hacer la predicción. Cabe resaltar que la mayoría de algoritmos de predicción funcionan sobre bases de datos completas, entonces el pre-procesamiento es vital.

En [Gad et al., 2017], los autores dejan atrás las técnicas comunes de completación de datos faltantes y exploran técnicas de aprendizaje automático como: regresiones linear y *kernel ridge*, Bosques aleatorios, y procedimientos de imputación con Máquinas de Soporte Vectorial y kNN. Las técnicas son aplicadas sobre la base de datos temporales de Administración Atmosférica y Oceánica (NOAA) [NOAA, 2018].

2.3. Estudios climáticos

El término “clima” tiene una varias definiciones que dependen del interés de la investigación. Generalmente se asocia los términos “clima” y “climatología” con temperatura [McGuffie and Henderson-Sellers, 2005], a pesar de que está compuesta de muchos mediciones y variables (precipitación, temperatura, humedad, etc.). Una definición presentada por varias investigaciones se podría definir en “Un conjunto factores químicos y físicos no vivientes en el medio ambiente, que afectan a los ecosistemas” [Asif et al., 2018]. Sin embargo, esta definición es muy general y no se presta para nuestros propósitos. la definición mas acertadas para nuestra investigación podría ser “[...] las estadísticas que describen la atmósfera y océano determinados sobre un intervalo de tiempo definido (estaciones, décadas o más), calculadas para el planeta en su totalidad o posiblemente para una región seleccionada” [McGuffie and Henderson-Sellers, 2005]. Con esta definición se puede definir el cambio climático como las variaciones en las estadísticas en un lapso de tiempo determinado. Algo importante es que estas variaciones emergen a partir de las interacciones entre diferentes componentes propios del sistema climático y factores externos.

*El avance de la tecnología ha permitido al ser humano recolectar información climática de diversas formas. Estas se dan mediante instrumentos fijos (sensores) hasta remotos (imágenes satelitales). Actualmente, los datos climáticos disponibles se pueden clasificar en cuatro grupos: *in situ*, de percepción remota (remote sensing), salidas del modelo matemáticos y paleo climáticos [Faghmous and Kumar, 2014]. Los mecanismos de percepción remota ofrecen información a gran resolución espacial (imágenes satélites) pero esta forma es muy reciente, datos paleo climáticos brindan información de largos periodos de tiempo (sedimentos glaciales, cortezas de árboles, etc.) pero no son de alta resolución y los modelos matemáticos generan información temporal y espacial de alta resolución pero necesitan de datos reales para inicializar sus parámetros. Al parecer los datos *in situ* son*

una buena fuente de información, pero solo se conoce los registros a partir de mediados de 1800 [Hudson and Quade, 2013].

Se pueden abordar los estudios climáticos de diversas formas. Algunos investigadores prefieren usar métodos tradicionales basados en la estadística, mientras otros aprovechan los grandes avances que ha tenido las técnicas computacionales. Casi todos los trabajos abordan un aspecto específico de los datos climáticos, aunque algunos de ellos proponen técnicas para problemas generales y muy pocos están usando técnicas de *aprendizaje de maquina* para estudiar los datos climáticos.

Balzter [Balzter et al., 2015] presenta, *Multi-Scale Entropy (MSE)*, un método para identificar las escalas temporales de variabilidad y su cambio en el tiempo en series temporales climáticas. MSE se obtiene al calcular la entropía de las series temporales particionadas a diferentes escalas temporales. Los resultados en anomalías de temperatura mensual de Europa central (CRUTEM4v) muestran que las escalas temporales del clima (1960-2014) son diferentes del promedio a largo plazo (1850-1960), debido a que las escalas temporales mayores a 12 meses, tienen mayor entropía que el promedio a largo plazo.

A partir de 1961 los patrones de temperatura mensual de aire son menos regulares a escalas temporales mayores que 12 meses. Estos resultados nos sugieren que, en estas escalas de tiempo interanuales, la variabilidad se ha convertido menos predecible que en el pasado.

Mudelsee [Mudelsee et al., 2014] presenta métodos estadísticos que tiene en cuenta la incertidumbre de los procesos climáticos para estimar sus parámetros. Se sabe que los intervalos de confianza más estrechos garantizan un entendimiento más preciso de los datos (menor incertidumbre). Para construir intervalos de confianza, hacen uso de enfoques clásicos y adaptaciones de métodos clásicos (*bootstrap*). Presentan una variedad de técni-

cas tales como: análisis de regresión lineal y no lineal, el análisis espectral y el análisis de series temporales de valores extremos en el caso de análisis de series de tiempo climáticas uni-variadas, correlación, así como otras variantes de la regresión para las series temporales bi-variadas.

Hennemuth [Hennemuth et al., 2015] presenta una variedad de métodos estadísticos que se pueden utilizar en estudios de investigación. Se muestran las diferentes aplicaciones de estos métodos para documentar el rango de posibilidades de su uso. Estos métodos presentados abarcan: estadísticos generales, distribuciones de frecuencia, análisis de series de tiempo, corrección de sesgo, pruebas de significancia, regionalización (*downscaling*, interpolación), análisis de valor extremo (método de selección, estimación de parámetros, métodos empíricos, métodos de análisis de valores extremos), índices (medidas de evaluación del modelo, índices estadísticos climáticos), métodos espacio-temporales, *ensemble analysis*.

Capítulo 3

Marco teórico

3.1. Bases de datos

Las bases de datos de hoy en día son esenciales en ambientes de negocios, administración, e investigación y se utilizan para mantener registros internos de usuarios, clientes, etc; en un ambiente de negocios ,como también información recogida por investigadores como temperatura, humedad, altitud, etc; en el caso de climatólogos o meteorólogos. El poder de las bases de datos resulta de un conjunto de conocimientos y tecnologías que se han ido desarrollando durante varias décadas. [Garcia-Molina, 2008].

Las capacidades que una Base de datos puede proveer al usuario son las de [Garcia-Molina, 2008]:

- Persistencia en el tiempo: como un archivo del sistema, dado que puede soportar almacenar grandes cantidades de datos que pueden existir independientemente de procesos que estén procesando los datos
- Interfaz de programación: permite al usuario o una aplicación acceder y modificar a través de un poderoso lenguaje de búsqueda, aquí es donde podemos ver la ventaja de usar una BD sobre un sistema de archivos ya que la flexibilidad de manipular datos almacenados en muchas formas complejas es más ventajoso que solo tener la

posibilidad de leer y escribir archivos.

- Gestión de transacciones: puede soportar accesos de datos concurrentes ya que diferentes procesos tienen la capacidad de acceder simultáneamente a los datos al mismo tiempo, y para evitar consecuencias que no deseamos de un acceso simultaneo, es capaz de soportar aislamiento permitiendo que las transacciones se ejecuten de una en una sola vez y atomicidad, requiriendo que la transacción se ejecute completa sino no, también agrega la capacidad de recuperarse de fallos y errores guardando históricas.

3.1.1. Series Temporales

Una serie temporal o cronológica es una secuencia de datos, observaciones o valores, medidos en determinados momentos y ordenados cronológicamente [Lin et al., 2007] . Los datos pueden estar espaciados a intervalos :

- Iguales como la temperatura en un observatorio meteorológico en días sucesivos al mediodía
- Desiguales como el peso de una persona en sucesivas mediciones en el consultorio médico, la farmacia, etc.

Uno de los usos más habituales de las series de datos temporales es su análisis para predicción y pronóstico así se hace por ejemplo con los datos climáticos, las acciones de bolsa, o las series de datos demográficos.

En cada punto de medición en el tiempo, pueden ser monitorizados uno o más atributos, y la serie temporal resultante es llamada uni-variada o multi-variada, respectivamente.

En muchos casos, una secuencia de símbolos puede ser usada para representar una serie

temporal.

Siendo un conjunto ordenado de n valores reales. Las series temporales pueden ser largas, a veces conteniendo miles de millones de observaciones. Sin embargo, por lo general el interés no está en las propiedades globales de la serie, sino, en las sub-partes de las series, las cuales son llamadas sub-secuencias [Chan and Fu, 1999]

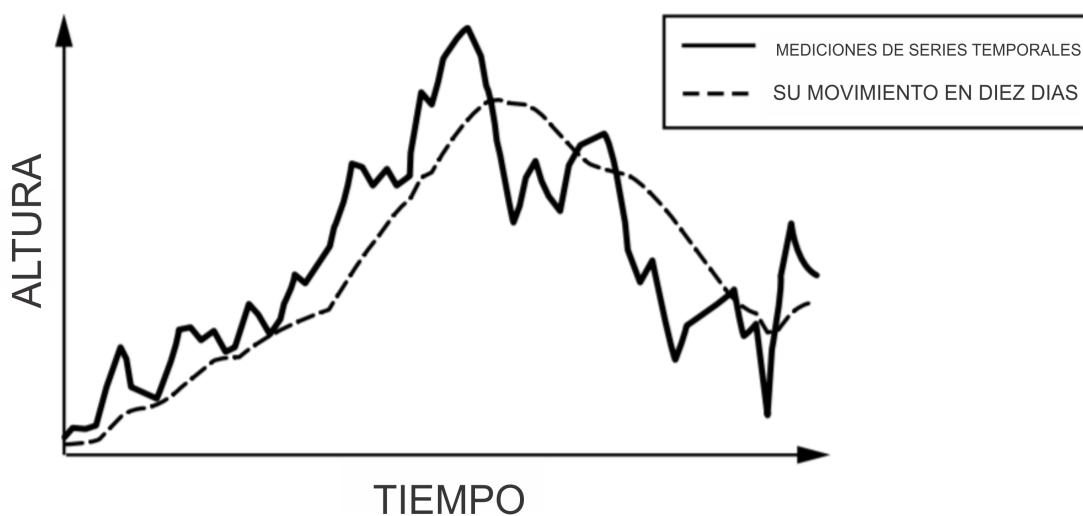


Figura 3.1: Diagrama de tendencia [Han et al., 2011]

Bases de datos de series temporales

Según [Han et al., 2011], una base de datos temporal normalmente almacena datos relacionales que incluyen atributos relacionados con el tiempo. Estos atributos pueden implicar varias marcas de tiempo, cada una teniendo semántica diferente. Una base de datos de secuencias almacena secuencias de eventos ordenados, con o sin un concreto noción de tiempo. Los ejemplos incluyen secuencias de compras de clientes, flujos de *clicks* de Web y secuencias biológicas. La base de datos de series temporales almacena secuencias de valores o eventos obtenidos sobre mediciones repetidas de tiempo (por ejemplo, hora, diaria, semanal). Los ejemplos incluyen datos recogida de la bolsa de valores, control de

inventarios y observación de fenómenos (como la temperatura y el viento).

Las técnicas de minería de datos se pueden utilizar para encontrar las características de la evolución del objeto, o la tendencia de los cambios de objetos en la base de datos. Dicha información puede ser útil en la planificación estratégica. Por ejemplo, la extracción de datos bancarios puede la programación de los cajeros de los bancos de acuerdo con el volumen de tráfico del cliente. Bolsa los datos se pueden extraer para descubrir tendencias que podrían ayudarle a planificar estrategias de inversión (por ejemplo, cuando es el mejor momento para comprar AllElectronics). Tales análisis suelen requerir definiendo la granularidad múltiple del tiempo. Por ejemplo, el tiempo puede ser descompuesto a años fiscales, años académicos o años civiles. Los años se pueden descomponer Trimestres o meses.

3.1.2. Base de datos de NOAA

Las bases de datos de NOAA (*National Centers for environmental information*) son conjuntos de datos terrestres (*in situ*) que se han desarrollado a partir de datos recolectados en los Estados Unidos y en el mundo [NOAA, 2018].

La disponibilidad de datos varía según el tipo de datos y la estación, y algunos tienen períodos de registro de más de un siglo. Todos los productos de datos ordenados de NOAA han sido certificados para el tribunal Americano.

Entre las bases de datos que podemos descargar, se encuentran las siguientes [NOAA, 2018]:

- QCLCD, que son conjuntos de datos climatológicos locales de calidad controlados a través de cuatro formas, por día, por hora, por hora de precipitación, y por hora

remarcada

- COOP, otorgadas por el servicio nacional del clima, que es información que ha sido recolectada por 10,000 voluntarios sobre el clima, en áreas urbanas y suburbanas. estos data set incluyen adicionalmente información hidrológica y meteorológica
- Normales de clima, son datos de mas de 3 décadas, de los diferentes estados de estados Unidos, estos contienen temperatura, precipitación, estos productos son lanzados cada 10 años, y contiene información por diaria y mensual, de la temperatura, las precipitaciones, la caída de nieve, el calentamiento, medidos por grados *Farenheit* y recolectada por mas de 9800 estaciones que opera en servicio nacional del clima.
- información de radiación solar, esta depende de la localidad y medida a través de rayos ultra-violetas y sus diferentes emisiones.

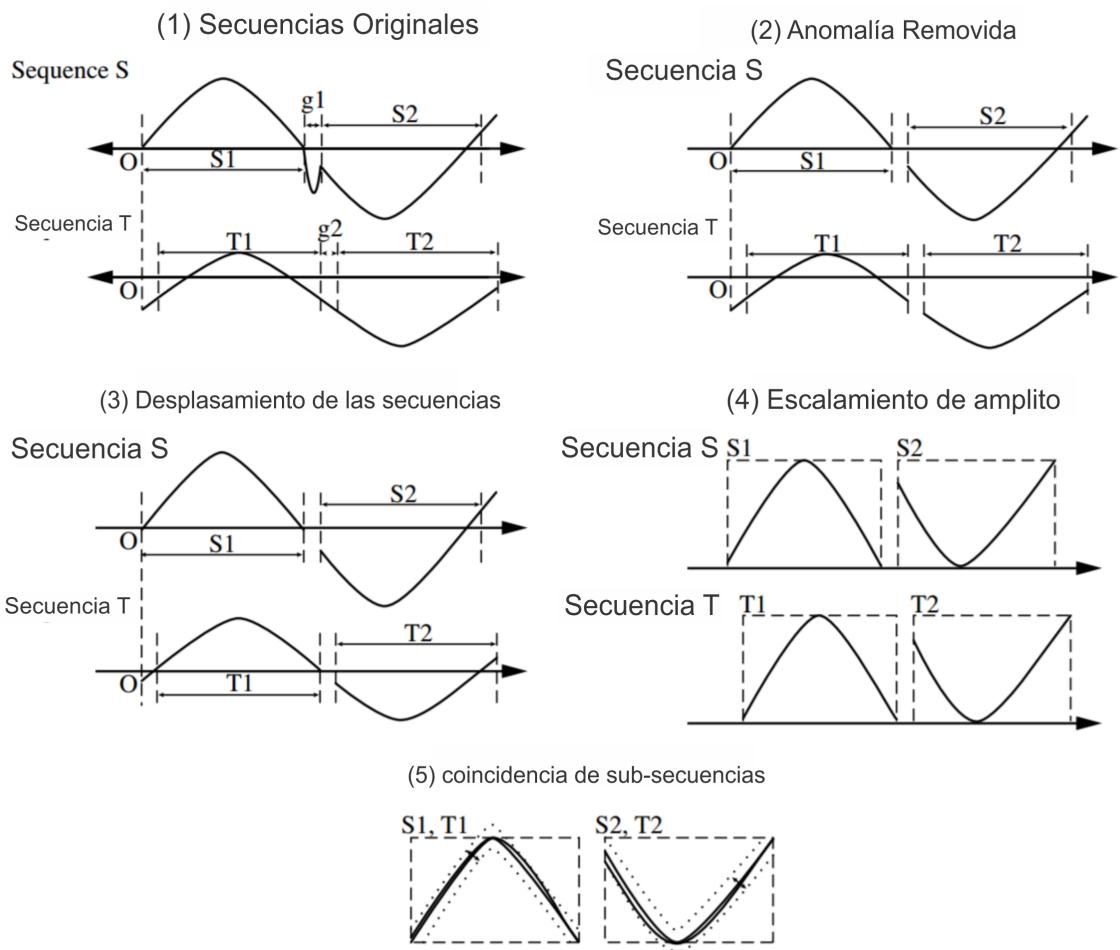


Figura 3.2: Coincidencias de sub-secuencias en series temporales [Han et al., 2011]

Dado que esta base de datos es muy amplia, sera tomada en cuenta como referencia ya que cuenta mediciones, satélites, radares , puntos aéreos y sensores de diversas localidad un ejemplo mostrado en imágenes sera las de la localidad de FLORIDA, USA. [NOAA, 2018].

APALACHICOLA AIRPORT, FL US

[View Station Details](#) | [View Station Report](#)

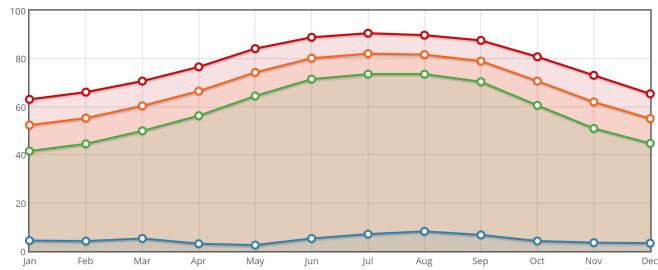


Figura 3.3: Imagen de la base de datos gráfica [NOAA, 2018]

DAY	MIN TEMP (°F)	AVG TEMP (°F)	MAX TEMP (°F)
01	42.1	52.5	62.9
02	42.0	52.4	62.9
03	41.9	52.4	62.8
04	41.8	52.3	62.8
05	41.7	52.2	62.8
06	41.6	52.2	62.7
07	41.5	52.1	62.7
08	41.5	52.1	62.7
09	41.4	52.1	62.7
10	41.4	52.0	62.7
11	41.3	52.0	62.7
12	41.3	52.0	62.7
13	41.3	52.0	62.8
14	41.2	52.0	62.8
15	41.2	52.0	62.8
16	41.3	52.1	62.9
17	41.3	52.1	62.9
18	41.3	52.1	63.0
19	41.3	52.2	63.0
20	41.4	52.2	63.1
21	41.4	52.3	63.2
22	41.5	52.4	63.2
23	41.5	52.4	63.3
24	41.6	52.5	63.4
25	41.7	52.6	63.5
26	41.8	52.7	63.6
27	41.9	52.8	63.7
28	42.0	52.9	63.8
29	42.1	53.0	63.9
30	42.2	53.1	64.0
31	42.3	53.2	64.1

Figura 3.4: Imagen donde se registran las mediciones climáticas de NOAA [NOAA, 2018]

3.1.3. Base de datos de IIAP

El Instituto de Investigaciones de la Amazonia Peruana [IIAP, 2018] en más de 34 años de vida institucional ha producido cuantiosa información útil para la toma de decisiones sobre desarrollo socio-productivo en la Amazonia, y cuenta con un Repositorio Digital que consiste en una plataforma que trabaja como un servicio digital que brinda el IIAP buscando compartir información, producto de la investigación científica y tecnológica sobre los recursos naturales y ambiente amazónicos, posee bases de datos de artículos científicos de diferentes tipos, como el objetivo de estudiar el comportamiento ambiental en esta región, esta entidad nos ha otorgado un conjunto de información perteneciente de una estación de medición en la región de Ucayali.

Por lo tanto se utilizará para su estudio los siguientes datos de la localidad de la Amazonía Peruana, entre las mediciones solicitadas se encuentro.

- Temperatura Máxima
- Temperatura mínima
- Precipitaciones
- Altura geoidal
- Calentamiento

Dado a que en la Amazonía aun son escasas las fuentes de datos [IIAP, 2018], es espera recolectar toda esta información y comparar la eficiencia.

3.2. Proceso de Minería de datos

La minería de datos según [Han et al., 2011] es la etapa de análisis de *Discovery in Databases Knowledge* o KDD, es un campo de la estadística y las ciencias de la computación referido al proceso que intenta descubrir patrones en grandes volúmenes de conjuntos de datos. Utiliza los métodos de la inteligencia artificial, aprendizaje automático, estadística y sistemas de bases de datos, o también es el proceso de detectar la información procesable de los conjuntos grandes de datos. Utiliza el análisis matemático y estadístico para deducir los patrones y tendencias que existen en los datos. Normalmente, estos patrones no se pueden detectar mediante la exploración tradicional de los datos porque las relaciones son demasiado complejas o porque hay demasiado datos.

Estos patrones y tendencias se pueden recopilar y definir como un modelo de minería de datos. Los modelos de minería de datos se pueden aplicar en escenarios como los siguientes [Han et al., 2011]:

- **Pronóstico:** por ejemplo se presenta durante cálculo de las ventas, por ejemplo predicción de las cargas del servidor o del tiempo de inactividad del servidor.
- **Riesgo y probabilidad:** por ejemplo se puede observar elección de los mejores clientes para la distribución de correo directo, determinación del punto de equilibrio probable para los escenarios de riesgo, y asignación de probabilidades a diagnósticos y otros resultados.
- **Recomendaciones:** por ejemplo se presenta determinación de los productos que se pueden vender juntos y generación de recomendaciones.
- **Búsqueda de secuencias:** por ejemplo se da en un análisis de los artículos que los clientes han introducido en el carrito de la compra y predicción de posibles eventos.

- **Agrupación:** por ejemplo se da en la distribución de clientes o eventos en grupos de elementos relacionados, y análisis y predicción de afinidades.

La generación de un modelo de minería de datos forma parte de un proceso mayor que incluye desde la formulación de preguntas acerca de los datos y la creación de un modelo para responderlas, hasta la implementación del modelo en un entorno de trabajo. Este proceso se puede definir mediante los seis pasos básicos siguientes:

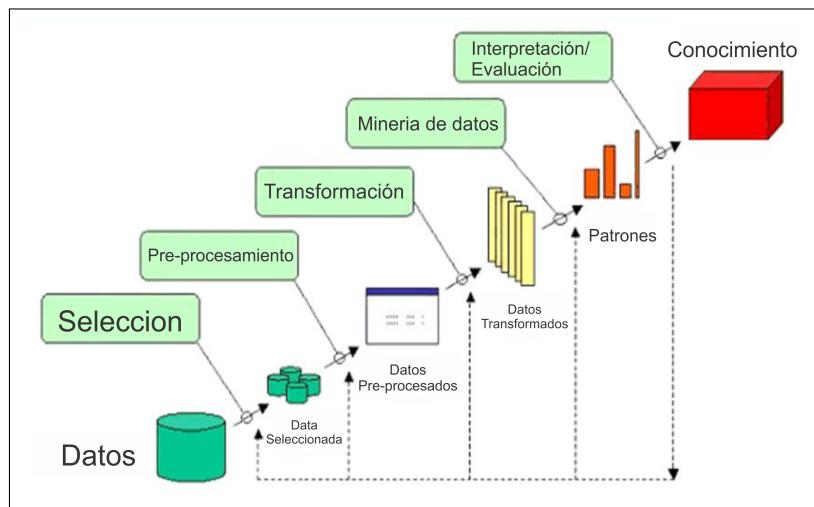


Figura 3.5: Proceso de minería de datos(adaptado al español) [Han et al., 2011]

Almacenamiento

Las fuentes de información para los procesos de minería de datos son variadas, entre sus diversas categorías, puedes tener la data cruda(*raw data*) almacenada en servidores, muchas veces no ordenadas, o también tenerlas indexadas en bases de datos SQL o noSQL, o incluso en servidores montados en la nube, que faciliten el acceso a la información [Han et al., 2011].

Este paso es necesario dado que el proceso de minería de datos es dependiente de información, dado que es la materia prima para que se identifiquen información de la información.

Pre-procesamiento

El “Pre-procesamiento de Datos” o “La Preparación de datos”, engloba a todas aquellas técnicas de análisis de datos que permite mejorar la calidad de un conjunto de datos de modo que las técnicas de extracción de conocimiento/minería de datos puedan obtener mayor y mejor información (mejor porcentaje de clasificación, reglas con más completitud, etc.) [Han et al., 2011].

Partes de la preparación de datos

Es difícil dar una lista exacta de tareas o tópicos, diferentes autores dan diferentes tareas y clasificaciones, por lo que se ha resumido según el proceso mostrado en la figura del proceso de minería de datos [Han et al., 2011].

Se pueden incluir las siguientes tareas o tópicos:

- ***Data collecting and integration:*** Obtiene los datos de diferentes fuentes de información, Resuelve problemas de representación y codificación, Integra los datos desde diferentes tablas para crear, información homogénea,
- ***Data cleaning:*** Resuelve conflictos entre datos, elimina el ruido, Chequea y resuelve problemas de ruido y valores perdidos
- ***Data transformation:*** Los datos son transformados o consolidados de forma apropiada para la extracción de información. Diferentes vías:
 - Sumarización de datos
 - Operaciones de agregación, etc.

- **Data reduction:** Selecciona datos relevantes para la tarea de la minería de datos/extracción de información. Diferentes vías para la Reducción de Datos:
 - Selección de Características
 - Selección de Instancias
 - Discretización

Análisis

Esta etapa, es la mas importante del proceso de minería de datos, dado que para esta, se debe tener definido que tipo de análisis se va a dar a la data preprocesada, utilizando las técnicas de la minería de datos provienen de la inteligencia artificial y de la estadística, dichas técnicas, no son más que algoritmos, más o menos sofisticados que se aplican sobre un conjunto de datos para obtener unos resultados [Han et al., 2011].

Las técnicas más representativas son:

- **Redes neuronales.**- Son un paradigma de aprendizaje y procesamiento automático inspirado en la forma en que funciona el sistema nervioso de los animales. Se trata de un sistema de interconexión de neuronas en una red que colabora para producir un estímulo de salida.

Algunos ejemplos de red neuronal son:

- El perceptrón.
- El perceptrón multicapa.
- Los mapas auto-organizados, también conocidos como redes de Kohonen.

- **Regresión lineal.**- Es la más utilizada para formar relaciones entre datos. Rápida y eficaz pero insuficiente en espacios multidimensionales donde puedan relacionarse más de 2 variables.
- **Árboles de decisión.**- Un árbol de decisión es un modelo de predicción utilizado en el ámbito de la inteligencia artificial y el análisis predictivo, dada una base de datos se construyen estos diagramas de construcciones lógicas, muy similares a los sistemas de predicción basados en reglas, que sirven para representar y categorizar una serie de condiciones que suceden de forma sucesiva, para la resolución de un problema.

Ejemplos:

- Algoritmo ID3.
 - Algoritmo C4.5.
- **Modelos estadísticos.**- Es una expresión simbólica en forma de igualdad o ecuación que se emplea en todos los diseños experimentales y en la regresión para indicar los diferentes factores que modifican la variable de respuesta.
 - **Agrupamiento o Clustering.**- Es un procedimiento de agrupación de una serie de vectores según criterios habitualmente de distancia; se tratará de disponer los vectores de entrada de forma que estén más cercanos aquellos que tengan características comunes.

Ejemplos:

- Algoritmo K-means.
 - Algoritmo K-medoids.
- **Reglas de asociación.**- Se utilizan para descubrir hechos que ocurren en común dentro de un determinado conjunto de datos.

Según [Weiss and Indurkhy, 1998] el objetivo del análisis de los datos, los algoritmos utilizados se clasifican en:

- supervisados
- no supervisados

Donde los **algoritmos supervisados (o predictivos)**: predicen un dato (o un conjunto de ellos) desconocido a priori, a partir de otros conocidos.

y los **Algoritmos no supervisados (o del descubrimiento del conocimiento)**: se descubren patrones y tendencias en los datos.

Presentación

Finalmente, una vez obtenido el o los modelos de conocimiento mediante el uso de las distintas técnicas aplicables, se procede a la validación de los mismos tras compararlos e interpretarlos, y a la elección del más satisfactorio según los resultados obtenidos. Si ningún modelo satisface las expectativas de conocimiento esperadas, el proceso se repite de nuevo cambiando variables y adoptando técnicas distintas a las usadas en los procesos anteriores, hasta obtener un modelo la explotación del cual satisface las necesidades de conocimiento [Han et al., 2011].

La visualización de datos y, en concreto, el uso de las herramientas que cumplen este cometido es fundamental para agilizar el proceso y ahorrar tiempo y esfuerzos a los expertos que deben determinar, con la máxima rapidez y eficiencia, si los modelos obtenidos se ajustan con lo esperado. En esa comparativa entre modelos y su evaluación para determinar si son lo suficientemente satisfactorios es donde entran en juego las herramientas de visualización de datos, que simplifican y agilizan la tarea de los expertos permitiendo

optimizar el proceso del data mining, reduciendo el tiempo empleado para llevarlo a cabo y minimizando los riesgos asociados a una mala interpretación de los resultados obtenidos [Han et al., 2011].

No cabe duda que la minería de datos y, por consiguiente, la visualización de datos están íntimamente relacionados con una correcta gestión de los datos y la información. No olvidemos que los datos representan uno de los valores principales de cualquier organización y que, por lo tanto, su correcta administración es de gran importancia el devenir corporativo [Han et al., 2011].

3.3. Consultas por Similitud

En el área de base de datos, las técnicas de indexación tienen como objetivo auxiliar el almacenamiento y la recuperación eficiente de datos. Muchas de estas técnicas se aplican a tareas de recuperación de información. En general, la organización se lleva a cabo a través de una estructura de indexación, también denominado como Método de Indexación(MI) o Método de acceso(MA).

Sin embargo, debido a la “maldición de la alta dimensionalidad”, el tiempo necesario para llevar a cabo la búsqueda por similitud, como kNN(*K-nearest neighbor*), puede crecer exponencialmente con la dimensionalidad de los datos. Aunque no se conoce ningún método para un buen desempeño en todas las instancias del problema, hay muchos algoritmos para acelerar las consultas, por lo general a través de una aproximación a la respuesta correcta. Así, se definen dos tipos de búsqueda: Métodos de búsqueda exactos y métodos de búsqueda aproximada; sobre los métodos de búsqueda de línea Búsqueda exacta de los métodos de acceso espacial (SAMs) y los métodos de acceso Métrico (MAM) son los más conocidos y utilizados en muchas aplicaciones.

La línea de investigación de los métodos de búsqueda exacta, los Métodos de Acceso Espaciales (MAEs), como *Kd–Tree* [Bentley and Ottmann, 1979], *R–Tree* [Guttman, 1984], *R*–Tree* [Fell et al., 1993], *R+–Tree* [Sellis et al., 1987] y, *X–Tree* [Böhm et al., 2001], describen los datos de entrada como vectores multidimensionales. Estos métodos fueron concebidos para apoyar las operaciones de búsqueda involucrando puntos y objetos geométricos. [Guttman, 1984, Böhm et al., 2001, Ocsa Mamani, 2015].

3.3.1. Dominio de datos

la concepción de un método de búsqueda debe considerar la naturaleza de los datos, a fin de explotar ciertas propiedades del espacio en donde están embebidos.

La gran mayoría de métodos de búsqueda supone que los datos pertenecen a un espacio n-dimensional. Esos espacios están en R^n , lo que permite el uso de propiedades geométricas en la solución.

Por otro lado, algunos métodos tienen exigencias menos rigurosas, como el caso de los métodos basado en espacios métricos, que solo usan las distancias entre los puntos, y ninguna otra propiedad geométrica [Guttman, 1984, Ocsa Mamani, 2015].

3.3.2. Espacios Métricos

Un espacio métrico es definido como $M = \langle S, d \rangle$, donde S es un universo de elementos y d es una función de distancia (o métrica), definida sobre los elementos en S , que mide la disimilitud los objetos y satisfacen las siguientes condiciones, $\forall_{x,y,z} \in S$:

Tabla 3.1: Condiciones

(1)	$d(x, y) \geq 0$	Positividad
(2)	$d(x, y) = 0 \leftrightarrow x = y$	Reflexividad
(3)	$d(x, y) = d(y, x) \geq 0$	Simetria
(4)	$d(x, y) \leq d(x, y) + d(y, z)$	Desigualdad

La desigualdad triangular es considerado una de las propiedades mas importantes, pues es utilizada para establecer los límites del valor de distancia entre dos objetos sin la necesidad del cálculo real de distancia, acelerando así los algoritmos de consulta por similitud. Mas específicamente, dados los valores de distancia $d(x, z)$ y $d(y, z)$ los límites para el valor (desconocido) de $d(x, y)$ son $|d(x, y) - d(y, z)| \leq d(x, y) \leq d(x, z) + d(y, z)$.

No todas las propiedades (1)-(4) son necesarias para todos los métodos. Por ejemplo, se puede substituir (2) por una propiedad mas débil $\forall x \in S, f(x, x) = 0$, tornando el espacio pseudo-métrico. Los métodos que no obedecen las propiedades (3), (4) o ambas son típicamente llamados no métricos [Fell et al., 1993, Ocsa Mamani, 2015].

3.3.3. Espacios Multidimensionales

Si los objetos del dominio S corresponden a los vectores de valores numéricos entonces el espacio es llamado Espacio Multidimensional o Espacio Vectorial con Dimensión Finita. Los objetos de un espacio multidimensional de dimensión n (*o n-dimensional*) son representados por n coordenadas de valores reales x_1, \dots, x_n .

Las funciones de distancia métrica mas común para medir la similitud entre elementos en un Espacio Multidimensional son de la familia L_p , o Minkowski, definidas por:

$$L_\infty((x_1, \dots, x_n), (y_1, \dots, y_n)) = \max_{i=1}^n |x_i - y_i| \quad (3.1)$$

- La distancia L_1 es también conocida como distancia **Manhattan**, y corresponde a una simple suma de las diferencias absolutas de los componentes.
- La distancia L_2 es también conocida como distancia **Euclidiana**, y corresponde a la idea usual de distancia en espacios 2D o 3D.
- La distancia L_∞ y es también conocida como **Chebychev**, y corresponde a la diferencia máxima absoluta entre los componentes, definida por:

$$L_p((x_1, \dots, x_n), (y_1, \dots, y_n)) = \left(\sum_{i=1}^b |x_i - y_i|^p \right)^{1/p} \quad (3.2)$$

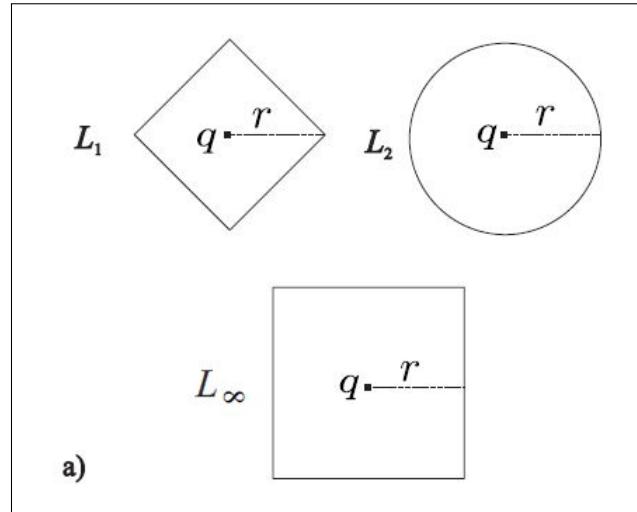


Figura 3.6: Formas geométricas que ilustran la delimitación de una región de búsqueda de acuerdo a la métrica L_p utilizada [Ocsa Mamani, 2015]

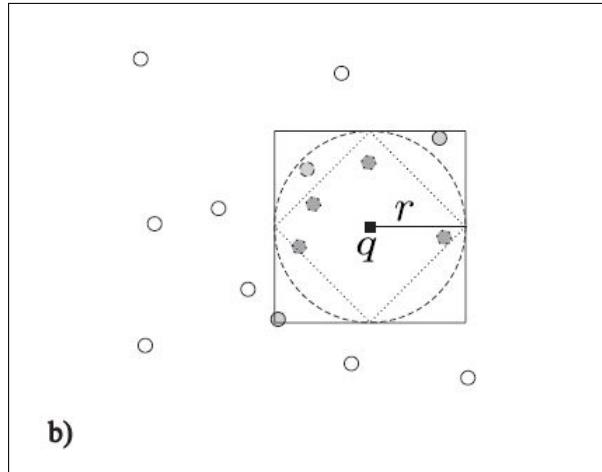


Figura 3.7: Ejemplo de consulta por rango para diferentes métricas L_p [Ocsa Mamani, 2015]

3.3.4. Tipos de Consultas de Similitud

Aplicaciones en la recuperación de información por similitud, por lo general crea un universo de objetos U y una función $d : U \times U \rightarrow R$ que mide la distancia entre dos objetos en U . En los espacios métricos, definimos $S \subseteq U$ como un conjunto finito de objetos, donde la función $d()$ mide la disimilitud entre objetos.

Dado un objeto de consulta $q \in U$, los objetos similares a q se pueden recuperar de acuerdo a los siguientes tipos de consultas por similitud:

Consulta por radio (range query $R_q(q, r)$) consulta que tiene como objetivo recuperar los objetos similares a q que están dentro del rango de la consulta r .

$$R_q(q, r) = \{u \in S \mid d(u, q) \leq r\} \quad (3.3)$$

En la ecuación 3.3 se representa una consulta por rango en un espacio bi-dimensional con la métrica L_2 . Los elementos contenidos por el radio e componen la respuesta. Vale

recordad que no es necesario que el elemento de consulta pertenezca al conjunto de datos de búsqueda, debiendo este pertenecer al mismo dominio de datos.

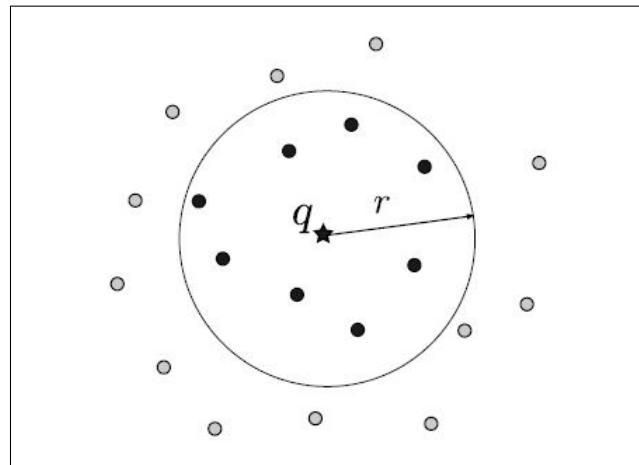


Figura 3.8: Ejemplo de consulta por rango. [Ocsa Mamani, 2015]

Consulta de los k-vecinos más cercanos (k-Nearest neighbor query $kNN(q, k)$)

consulta que tiene como objetivo recuperar los k objetos más cercanos al objeto de búsqueda q . O más formalmente, los k vecinos más próximos definen el conjunto $C = \{s_1, s_2, \dots, s_k\}$ en que:

$$\forall s_i \in C, \forall s_j \in S - C, d(q, s_i) \leq d(q, s_j) \quad (3.4)$$

En la ecuación 3.4 ejemplifica una consulta kNN en un espacio bi-dimensional con la métrica L_2 . En la figura, la consulta tiene como entrada el elemento de consulta q y el valor de k igual a 3. Los elementos conectados a q corresponden al consulto de respuesta, considerando que en el ejemplo el elemento q no pertenece al conjunto de datos.

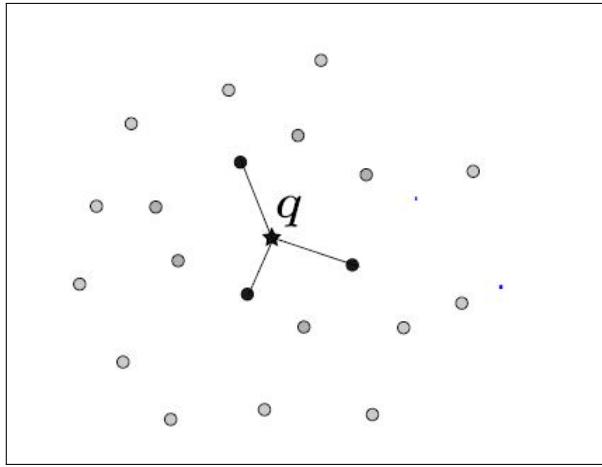


Figura 3.9: Ejemplo de consulta de los k -vecinos más cercanos [Ocsa Mamani, 2015]

Consulta a todos los k -vecinos más cercanos $AllkNN(Q, k)$: Consulta que busca recuperar todos los k objetos más cercanos a todos los objetos de búsqueda $q \in Q$, en donde Q es el conjunto de datos original, o sea $Q = S$.

Tal como la búsqueda a los k -vecinos más cercanos (kNN), este puede ser restringido a la búsqueda de más cercano (NN), este tipo de consulta (AllkNN) puede también ser reducida a un tipo de consulta AllNN. En esta consulta, todos los vecinos más cercanos a los elementos de Q son buscados, y apenas aquellos cuya distancia sea diferente a cero son retornados, o sea, no se puede retornar como conjunto de respuesta elementos iguales al conjunto de objetos de consulta. La siguiente Figura ilustra la noción de consulta AllkNN para un conjunto de datos bi-dimensional

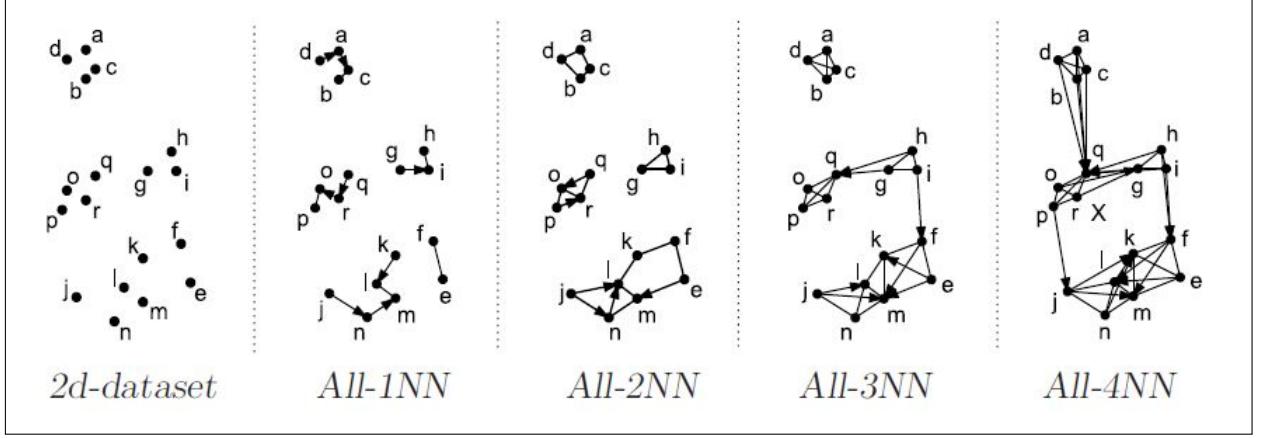


Figura 3.10: Ejemplo de consulta AllkNN para un conjunto de datos bidimensionales [Ocsa Mamani, 2015]

Para las consultas por similitud, de interés en este trabajo, una solución obvia es la búsqueda secuencial, en que cada elemento del conjunto de datos es comparado en relación al objeto de consulta.

Sin embargo, una solución de fuerza bruta apenas es aceptable para pequeñas base de datos, siendo inviable para problemas de gran escala, como por ejemplo la búsqueda de imágenes por contenido en la Web. Así mismo, algunas aplicaciones la búsqueda kNN exacta es tan costosa que, muchas veces, las soluciones aproximadas son aceptables. En esos casos, la utilización de las siguientes definiciones serán usadas.

Consulta por rango aproximada $(1 + \varepsilon)Rq(q, r)$: Consulta que busca recuperar los objetos cercanos a q que se encuentran dentro del radio de consulta $(1 + \varepsilon) \times r$, más formalmente:

$$(1 + \varepsilon)Rq(q, r) = \{u \in S \mid d(u, q) \leq (1 + \varepsilon) \times r\} \quad (3.5)$$

En la ecuación 3.5 ejemplifica una consulta de este tipo para L_2 . Los elementos dentro del radio de consulta $(1 + \varepsilon) \times r$ componen la respuesta aproximada.

Consulta Aproximada a los k-vecinos KNN más cercanos $(1+\varepsilon)-kNN(q, K)$:

Sea r la distancia entre el objeto de consulta q y el elemento más distante entre los k verdaderos vecinos más próximos, esto es,

$$r = \max \{s_j \in S \mid d(q, s_j)\} \quad (3.6)$$

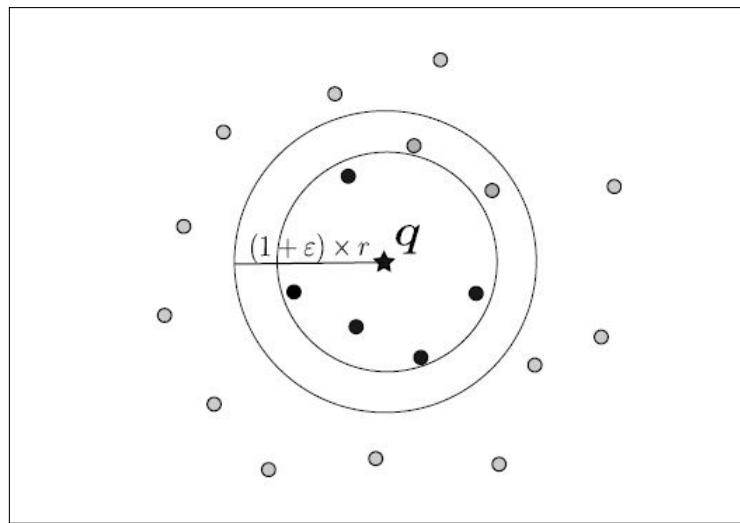


Figura 3.11: Ejemplo de consulta por rango aproximada [Ocsa Mamani, 2015]

La búsqueda $(1 + \varepsilon) - kNN$: para los k elementos mas similares a q consiste en encontrar un conjunto, $C = \{s'_1, s'_2, \dots, s'_k\}$ en donde,

$$\forall s'_i \in C', d(q, s'_i) \leq (1 + \varepsilon) \times r \quad (3.7)$$

El factor $(1 + \varepsilon)$ es usualmente llamado **factor de aproximación**, y indica que el conjunto de la solución C' está dentro de un **error relativo** ε a la respuesta exacta

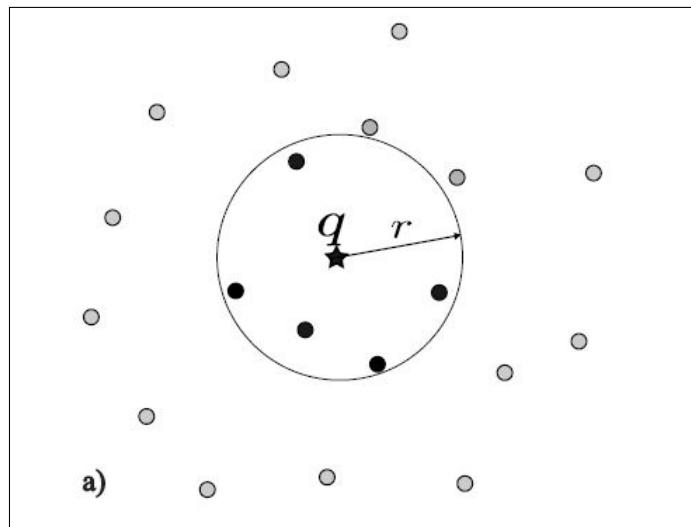


Figura 3.12: Búsqueda aproximada 5-NN, búsqueda exacta [Ocsa Mamani, 2015]

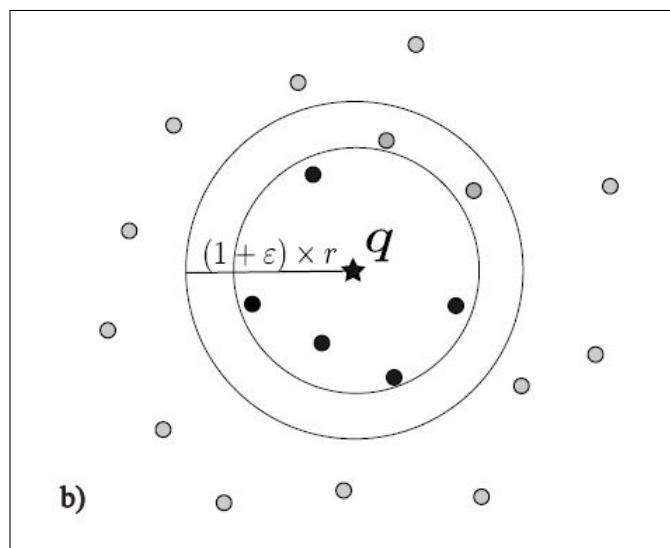


Figura 3.13: Búsqueda aproximada 5-NN, búsqueda aproximada. [Ocsa Mamani, 2015]

Consulta Aproximada a todos los k -vecino AllkNN más cercanos ($1 + \varepsilon - AllkNN(Q, k)$): Este tipo de consulta aproximada está basado en la definición exacta de AllkNN y la consulta aproximada $(1 + \varepsilon)kNN(q, k)$. Así, en este problema, utilizando la definición aproximada de búsqueda k NN, se busca los k -vecinos más cercanos a cada objeto $qinQ$, en donde Q es el conjunto de datos original, ósea $Q = S$.

3.3.5. Alta Dimensionalidad

Definida también como *La Maldición de la alta Dimensionalidad* La eficiencia de los métodos de búsqueda por similitud dependen mucho de la dimensionalidad, aunque el tiempo de búsqueda pueda alcanzar un costo logarítmico en relación al tamaño de los datos, este va crecer exponencialmente con la dimensionalidad de los datos.

Para dimensiones moderadas, la mayoría de los métodos ejecutan las consultas de manera suficientemente eficientes para permitir una solución exacta en un tiempo razonable.

Para dimensiones más altas, se vuelve viable el uso de métodos aproximados, lo que implica en un *trade – off* entre la precisión y la eficiencia. Para dimensiones más altas ese *trade – off* va convertirse progresivamente más relevante, resultando en penalidades mayores en términos de precisión, a fin de obtener una eficiencia aceptable.

El problemas de *la maldición de la alta dimensionalidad* fue estudiado originalmente por Bellman [Bellman, 1962], observando que particiones del espacio de soluciones en problemas de optimización son ineficientes en problema con datos en altas dimensiones. Los efectos de “maldición” en Métodos de Acceso Espaciales (MAEs) y Métodos de Acceso Métricos (MAMs) son discutidos en [Böhm et al., 2001, Blott and Weber, 2008, Ocsa Manani, 2015]

3.4. Minería de series temporales

La Minería de Datos (Data Mining) es un área de investigación dentro en un contexto más amplio llamado Descubrimiento del Conocimiento en Bases de Datos (KDD – Knowledge Discovery in Databases), cuyo objetivo principal es extraer de un conjunto de datos,

el conocimiento a ser utilizado en procesos decisarios. Más específicamente, los principales objetivos de los métodos de minería de datos son una descripción de un conjunto de datos y una predicción de valores futuros de interés basado en conocimiento previo de un banco de datos [Fayyd et al., 1996]

De acuerdo con la literatura del área, las principales tareas en la minería de series temporales son: Identificación de agrupamientos, clasificación, identificación de intrusos, encontrar patrones frecuentes, encontrar reglas de asociación y pronóstico. Aunque algunas de esas tareas sean semejantes a tareas correspondientes a minería de datos, el aspecto temporal coloca algunas inquietudes específicas que son consideradas y/o restricciones impuestas a las aplicaciones correspondientes. Primero, como tareas y algoritmos de minería, pueden estar basados en búsqueda por similitud, se convierte necesaria la utilización de medidas de similitud entre series temporales, esta cuestión es muy importante en la minería de series temporales, pues envuelven un cierto grado de subjetividad que puede afectar el resultado final de tareas y algoritmos que utilizan la búsqueda por similitud como parte del proceso de minería. Una segunda cuestión, relacionada a la medida de similitud, es la representación de series temporales a fin de reducir la dimensión. Como la cantidad de datos puede variar de pocos *megabytes* a *terabytes*, una representación de una serie temporal es necesaria para manipular y analizar los datos de manera eficaz. [Fayyd et al., 1996]

3.4.1. Representación de Series Temporales

Una serie temporal es una colección de observaciones recogidas secuencialmente a lo largo del tiempo. En cada punto de medición en el tiempo, pueden ser monitoreados uno o más atributos, y la serie temporal resultante es llamada univariada o multivariada, respectivamente. En muchos casos, una secuencia de símbolos puede ser usada para representar una serie temporal.

Una serie temporal $T = (T_1, T_2, \dots, T_n)$ es un conjunto ordenado de n valores reales. Las series temporales pueden ser largas, a veces conteniendo miles de millones de observaciones. Sin embargo, por lo general el interés no está en las propiedades globales de la serie, sino, en las sub-partes de las series, las cuales son llamadas sub-secuencias.

La Transformada Discreta de Fourier (DFT) [Agrawal et al., 1993] fue una de las formas de representación propuesta por primera vez en el contexto de minería de datos. DFT transforma una serie temporal a partir de un dominio de tiempo a un dominio de frecuencia. Una Transformada *Discreta Wavelet (Discrete Wavelet Transform DWT)* [Chan and Fu, 1999], transforma una serie temporal en espacio/frecuencia. El Singular Value decomposition (SVD) [Korn et al., 1997] realiza una transformación global, girando el eje del conjunto de datos de tal modo que el primer eje explica una variación máxima, el segundo eje explica el máximo de varianza restante y es ortogonal al primer eje, y así sucesivamente. Piecewise Aggregate Approximation (PAA) [Keogh et al., 2001] divide una serie temporal en segmentos de igual longitud y registra la media de los valores correspondientes de cada segmento.

Para agrupar series temporales similares, con un margen de ajuste variable, se necesita de métodos de aproximación de series temporales. Entre esos, el SAX *Symbolic Aggregate Approximation* (SAX) [Lin et al., 2007] se muestra como una de las mejores técnicas de representación de series temporales para tareas de minería. El SAX usa el PAA como primer paso para la representación y, en seguida, la serie resultante es transformada en una secuencia de símbolos usando las propiedades de distribución de probabilidad normal. Un ejemplo de representación discreta SAX es ilustrado en la Figura La transformación de la representación PAA a una representación discreta genera una secuencia de letras (palabras) para representar la serie temporal. La principal ventaja de SAX es obtener un modelo de representación, que reduzca la dimensión, pero preserve las características de la serie original. A pesar de perder alguna de las informaciones para el cálculo de la media del *Piece-wise Aggregate approximation*(PAA), el SAX es bueno contra el ruido, a no ser

que las variaciones repentinas son inherentes al dominio de aplicación.

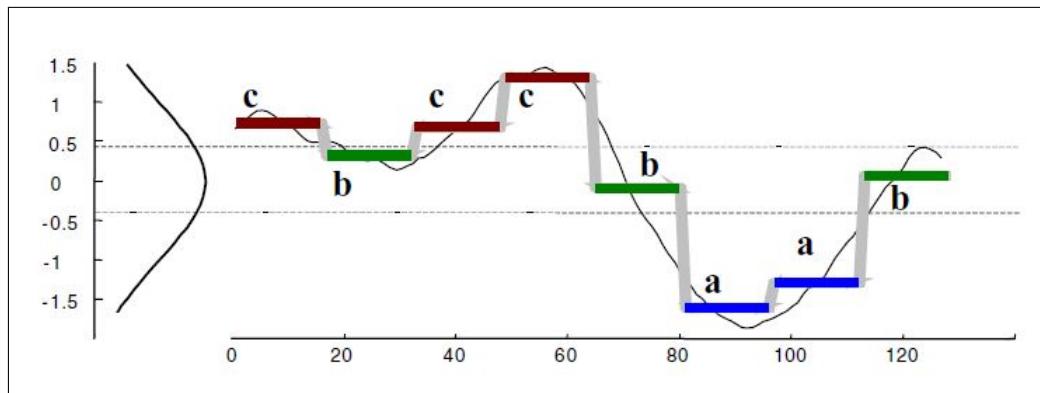


Figura 3.14: Una serie temporal (línea negra), ya normalizada, es discretizada, obteniéndose una representación PPA(línea gris gruesa). Después los coeficientes PPA en letras (en negrita) son mapeados a una representación discreta. En este ejemplo, una serie es discretizada para la palabra cbccbaab [Lin et al., 2007]

Medidas de Similitud en Series Temporales

La definición de nuevas medidas de similitud ha sido una de las áreas más investigadas en el campo de Minería de Series Temporales. Generalmente, las medidas de similitud están fuertemente relacionadas con el esquema de representación aplicada a los datos originales. Sin embargo, existen

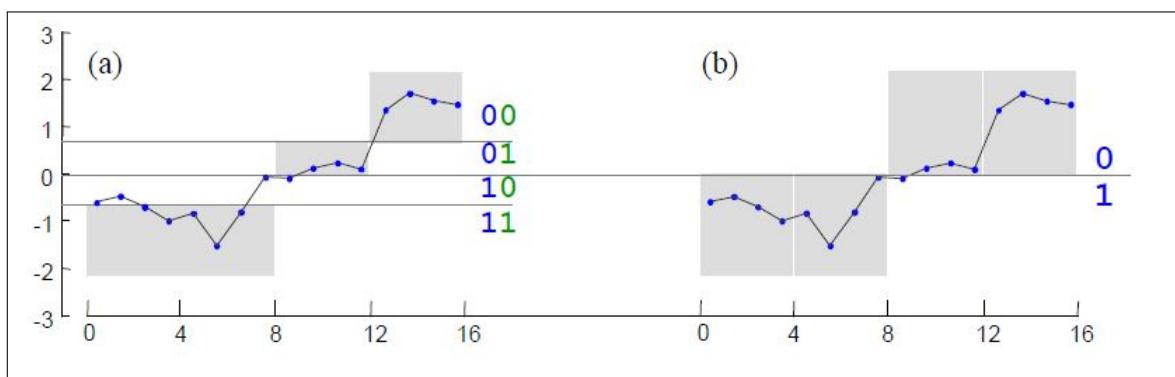


Figura 3.15: Una serie temporal T convertida en palabras SAX. (a) Con cardinalidad 4 $\{11, 11, 01, 00\}$ (b) con cardinalidad $\{1, 1, 0, 0\}$ [Lin et al., 2007, Mueen et al., 2009]

algunas medidas de similitud que aparecen frecuentemente en la literatura. La mayoría

de las opciones de los investigadores se basan en la familia Minkowski L_p , que incluye la distancia euclídea L_2 .

Otra medida de similitud que atrae mucha atención es el *Dynamic Time Warping* (DTW) [Berndt and Clifford, 1994]. La principal ventaja de esa medida es permitir la aceleración - desaceleración de una serie a lo largo de la dimensión temporal (alineamientos no lineales son posibles). Sin embargo, el DTW es computacionalmente caro.

A pesar de ser simple de calcular, la distancia euclídea produce resultados erróneos para series temporales que son similares, pero presentan distorsiones en el eje del tiempo. DTW es una medida de similitud más eficiente en ese caso. Al contrario de la distancia Euclídea, el DTW está basado en la idea de alineamientos no lineales entre series.

A pesar de ser simple de calcular, la distancia euclídea produce resultados erróneos para series temporales que son similares, pero presentan distorsiones en el eje del tiempo. DTW es una medida de similitud más eficiente en ese caso. Al contrario de la distancia Euclídea, el DTW está basado en la idea de alineamientos no lineales entre series.

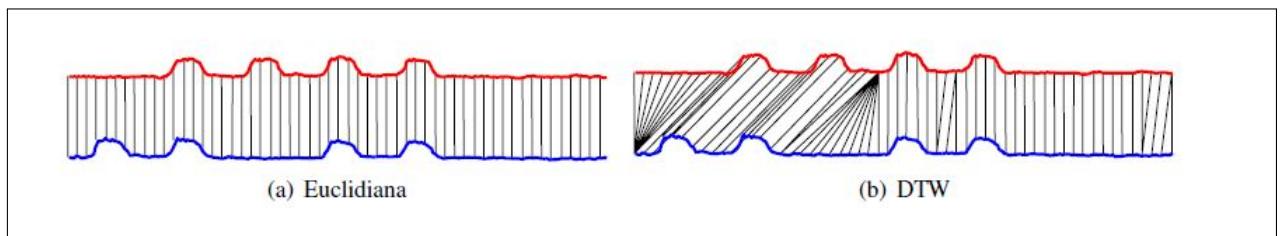


Figura 3.16: A pesar de que las dos series tienen formas similares, ellas no están aliñeadas en el eje del tiempo. La distancia Euclídea genera una medida de disimilitud pesimista, ahora y el DTW produce una medida de disimilitud más intuitiva debido a los alineamientos no-lineales [Berndt and Clifford, 1994]

Consulta por Similitud en Series Temporales

Diversos trabajos se han realizado para realizar consultas sobre series temporales. La mayoría de esos trabajos propone algoritmos eficientes que representan los datos (series

temporales) como vectores de longitud fija. Sin embargo, diversas áreas de aplicación generan series temporales sin restricción de longitud y, por lo tanto, consultas eficientes sobre esas series constituyen una importante tarea para recuperación de información y también para minería de series.

Las tareas de minería de series temporales más comunes son: identificación de agrupamientos, clasificación, identificación de intrusos, descubrimiento de motifs, descubrimiento de reglas de asociación y previsión. Una breve descripción de cada tarea es presentada a continuación [Berndt and Clifford, 1994, Mueen et al., 2009].

- **Detección de Agrupamiento:** Busca grupos de series temporales en un banco de datos de modo que series temporales de un mismo grupo son semejantes unas a las otras, mientras series temporales de grupos distintos son diferentes entre sí.
- **Clasificación:** Atribuir una serie temporal a una clase pre-definida de modo que la serie sea más parecida con las series de esa clase que con las series temporales de otras clases.
- **Detección de Intrusos:** Encuentra todas las series temporales que contienen un comportamiento diferente a lo esperado con respecto a un modelo base.
- **Detección de patrones frecuentes(motifs):** Encuentra patrones repetidos en series temporales que no sean previamente conocidas en el banco de datos
- **Detección de reglas de asociación:** Deducir reglas de una o más series temporales describiendo el comportamiento más probable que pueden presentar en un punto específico de tiempo (o rango).
- **Pronóstico:** Pronosticar eventos futuros con base en eventos pasados conocidos. En la mayoría de los casos, la predicción se basa en un modelo y los resultados de otras tareas de minería.

3.4.2. Patrones frecuentes(motifs) y Datos Irregulares(outliers)

La búsqueda de motifs es una tarea bien conocida en el área de bioinformática. Ese problema también despertó el interés de comunidades de minería de datos [Lin et al., 2007]. Los motifs son los patrones previamente desconocidos que ocurren con mayor frecuencia en los datos. Esos patrones pueden ser de particular importancia para otras tareas de minería de series temporales, tales como, identificación de agrupamientos, descubrimiento de reglas de asociación, identificación de anomalías y análisis del comportamiento. Un algoritmo eficiente para el descubrimiento de motifs también puede ser útil como una herramienta para sumarización y visualización de grandes volúmenes de datos.

Las definiciones siguientes presentan un resumen de la terminología, definida inicialmente en [Lin et al., 2007] , usada para la definición del problema.

- **Banco de series temporales:** Una base de datos de series temporales S es un conjunto no ordenado de N series temporales posiblemente de diferentes longitudes.
- **R-Motif:** Para definir el $R - motif$ de un banco de datos de series temporales S un parámetro de tolerancia R es usado para decidir si dos series temporales son suficientemente similares para ser consideradas parte del *motif*.
- **k-Motif :** Para definir el $k - motif$ de un banco de datos de series temporales S son usados los k elementos más próximos entre sí.
- **Pair-Motif :** Esta es una versión del problema, en el que se considera solo las dos sub-secuencias que están más próximas uno del otro, o sea, los que presentan la menor distancia entre sí. Así, el $Pair - Motif$ de un banco de datos de series temporales S es el par no ordenado $L_1;L_2$ en S que son lo más similares entre todos los pares posibles.
- **Top – K^{th} Motif :** El $Top - K^{th} Motif$ de un banco de datos de series temporales S

es el cluster clasificado en la K^{th} posición. Así se definen los $TopK - Motifs$ como los K patrones más frecuentes de la base, o sea los primeros $Kmotifs$.

A pesar de que muchos trabajos consideran el problema del descubrimiento de *motifs* como la búsqueda de los *motifs* en sub-secuencias de una serie temporal, esta definición tiene algunas cuestiones a resolver. Por ejemplo, se necesita definir si dos sub-secuencias seguidas son lo suficientemente diferentes como para ser consideradas dos sub-secuencias independientes, tal como puede ser visto en [Lin et al., 2007]. Así por cuestiones de simplicidad se considera trabajar solo sobre bancos de datos de series temporales.

Intuitivamente, la noción de $R - Motif$, puede ser vista como el cluster más denso en una proyección 2D de las series. En la siguiente Figura representa esta idea. La Figura (a) ilustra las series temporales a ser analizadas, la Figura (b) representa el *motif* encontrado y la Figura (c) representa la proyección de las series. Tenga en cuenta que si el *motif* es definido usando un parámetro de tolerancia R puede ser difícil saber cuál es el radio que mejor define el cluster más denso. Debido a que este parámetro es global y depende del dominio de los datos.

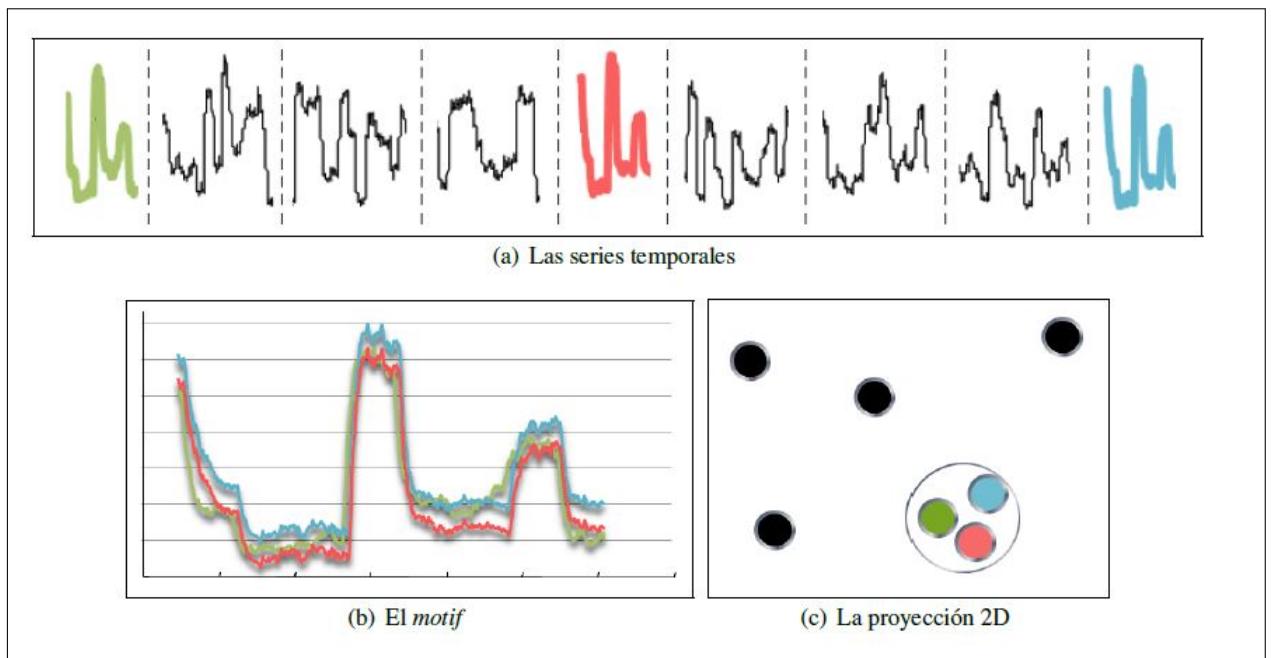


Figura 3.17: Representación del Motif. [Ocsa Mamani, 2015]

Por otro lado, si usamos la definición del $k - Motif$, en que el número de elementos en el *motif* es definido por el parámetro k , este es menos dependiente y más simple de ajustar. Así, el *motif* es definido por el radio del cluster más denso que contiene los k elementos más próximos entre sí.

Basado en el análisis previo de la definición del $R - Motif$ y el $k - Motif$, de aquí en adelante se usara la definición del $k - Motif$ para definir los $TopK - Motifs$, o sea, se buscara los patrones mas frecuentes de la base conformados por los k primeros clústers, en el que cada clúster tiene k elementos.

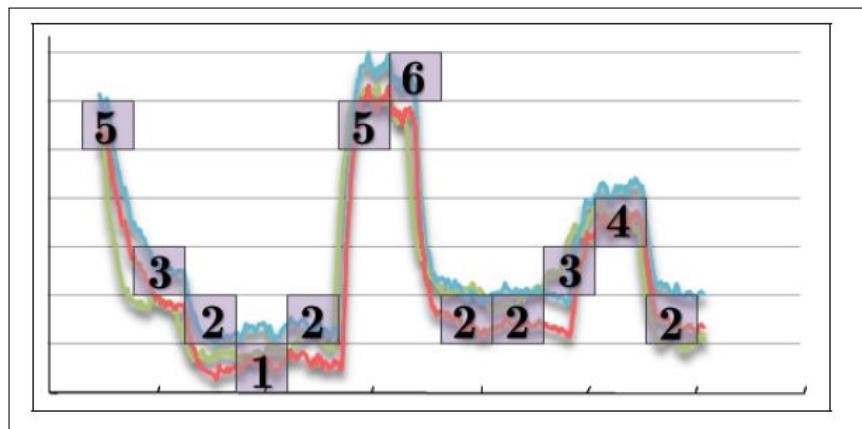


Figura 3.18: Las series temporales que definen un *motif* [Ocsa Mamani, 2015]

Algoritmo de Fuerza Bruta

Para facilitar una descripción del algoritmo general para encontrar motifs, primero considere el algoritmo de fuerza bruta para la identificación de motifs. El algoritmo de fuerza bruta, tiene un costo que depende del costo del algoritmo de búsqueda kNN. Así, si el algoritmo de búsqueda es secuencial la complejidad del algoritmo es cuadrática y si el algoritmo de búsqueda es apoyado por métodos de indexación esta complejidad puede disminuir sustancialmente.

El algoritmo mantiene el menor candidato del *motifs* así como su radio en cada iteración. Esas variables son actualizadas conforme el algoritmo encuentra clústers con radios

menores. Por lo tanto, el motif es el clúster con el menor radio. Tenga en cuenta que el algoritmo para la búsqueda del k-vecino más cercano es usado para generar clústers candidatos

Descubrimiento de *motifs* usando *Random Projection*

Debido a la complejidad cuadrática de los algoritmos exactos, los investigadores se han enfocado en soluciones aproximadas. Estas presentan, en general, la solución $O(n)$ o $O(n \log n)$ con elevados factores constantes [Mueen et al., 2009, Ocsa Mamani, 2015]. Los primeros trabajos que consideran ese enfoque [Chiu et al., 2003b, Buhler and Tompa, 2001], se basan en la investigación para el descubrimiento de patrones en secuencias de ADN de la comunidad bio-informática.

Los autores desarrollaron un algoritmo para encontrar *motifs* usando proyección aleatoria. El algoritmo de proyección aleatoria (*Random Projection*) básicamente usa vectores aleatorios provenientes de una distribución estable (*landmarks*). Después de un paso de discretización, usando la representación *SAX* [Lin et al., 2007, Ocsa Mamani, 2015], las secuencias son proyectadas en el espacio generado por esos vectores aleatorios.

Local Outlier Factor(LOF)

Para identificar valores atípicos basados en consultas KNN aproximadas, utilizamos Factor de valores atípicos locales, que es un algoritmo propuesto para encontrar puntos de datos atípicos midiendo la desviación local de un punto de datos dado con respecto a sus vecinos [Breunig et al., 2000]. Es más significativo asignar a cada objeto un grado de ser un valor atípico. Esa calificación se denomina factor atípico local de un objeto.

La diferencia clave entre LOF y las nociones existentes de Outliers es que estar aislado

no es una propiedad binaria. En cambio, le asignamos a cada objeto un factor atípico, que es el grado en que el objeto está aislado. LOF comparte algunos conceptos con bases de datos como los conceptos de "distancia de alcance", que se usa para estimar la densidad local.

3.5. GPU en CUDA

Las GPUs (Graphical Processing Units o Unidades de Procesamiento Gráfico) se están convirtiendo en omnipresentes en la computación de uso general, en especial para aplicaciones de alto desempeño. Una de las razones que fundamenta esta evolución es que como las GPUs fueron optimizadas para la renderización de gráficos en tiempo real, estas también son idóneas para operaciones de computación intensiva y altamente paralela. Ese campo es conocido como GPGPU (General-Purpose computations on the GPU) [Owens et al., 2007].

Debido al acelerado crecimiento del poder de procesamiento de las GPUs en comparación con las CPUs, se evidencia, de esta forma, que el poder de computación de la primera está creciendo a una tasa mayor que en las CPUs. Esta evolución en computación de alto desempeño usando las GPUs generó el desarrollo de aplicaciones no solo orientadas para procesamiento gráfico, sino también en áreas que presentan grandes volúmenes de datos para ser procesados, principalmente aquellos centrados en la investigación científica, tales como computación científica, bioinformática, base de datos, minería de datos, computación distribuida, entre otros [Owens et al., 2007].

En el esquema de GPGPU, la GPU es un co-procesador para la CPU. Como tal, la CPU es usada para la gestión del disco y la memoria, ósea, es responsable por lidiar con la lectura/escritura en disco, así como por las transferencias de datos entre la memoria de

la GPU y de la CPU. El procesamiento de las GPUs está basado en la arquitectura SIMD (Single Instruction, Multiple-Data), esto es, una arquitectura de procesamiento paralelo en el que varios conjuntos de datos pueden ser procesados simultáneamente por el mismo conjunto de instrucciones, de forma que estos conjuntos de datos son ejecutados en unidades de procesamiento distintas [Owens et al., 2007].

Por otra parte, las CPUs tradicionalmente poseen arquitectura SISD (Single Instruction SingleData), siendo, pues basadas en una arquitectura de procesamiento secuencial. Sin embargo, al mismo tiempo en que las GPUs mostraban tener un gran poder computacional, una arquitectura de múltiples núcleos impone restricciones en el diseño de los algoritmos, así como desafíos para alcanzar los picos de desempeño más altos.

La plataforma CUDA proporciona la oportunidad de reducir significativamente el tiempo de ejecución de tareas de recuperación y minería de datos, pues generalmente presenta su hardware disponible y accesible en términos de valores, sin necesitar de soluciones de hardware más complejos, como los servidores de aplicaciones de alto desempeño con configuraciones de múltiples CPUs [Ocsa Mamani, 2015].

Existen varias plataformas de programación GPGPU, se pueden destacar las siguientes: CUDA [Nickolls et al., 2008] y OpenCL [Stone et al., 2010]. La principal diferencia entre las plataformas es que los programas CUDA son ejecutados solamente en placas de video NVIDIA, en los programas OpenCL la ejecución es hecha en plataformas heterogéneas. La distribución de *kits* de desarrollo de software, como la NVIDIA-CUDA SDK, ha incentivado su uso como una forma de liberar la CPU de dominios de aplicación, que precisan de picos de desempeño más altos. La arquitectura NVIDIA-CUDA fue escogida para este trabajo por ser, al momento de escribir el texto, la plataforma dominante para computación de alto desempeño.

3.5.1. Arquitectura CUDA

La arquitectura de una GPU NVIDIA típica, con soporte de tecnología CUDA, está organizada en una matriz de Streaming Processors (SPs) altamente segmentados, y que forman, así, el conjunto de *StreamingMultiprocessors* (SMs). Todos los SM tienen un número de *StreamingProcessors* (SPs) que comparten el control lógico y la cache de las instrucciones. Las GPUs actualmente vienen con hasta 6 gigabytes de (GDDR) DRAM, conocida como memoria global. Por ejemplo, la *GeForceGTX470* (G470) tienen 1.280 MB de memoria global y tienen 448 SPs (14 SMs, cada uno con 32 SPs). El chip G470 soporta hasta 1.024 threads por SM, se resume en cerca de 14.000 *threads* para este chip. Note que las CPUs Intel tradicionales soportan 2, 4 o hasta 8 threads, dependiendo del modelo de la máquina.

La arquitectura CUDA de una NVIDIA GeForce GTX 8800 es ilustrada en la siguiente figura, consiste de 16 Streaming Multiprocessors (SMs), que contienen 8 núcleos o Streaming Processors (SPs). Los 8 núcleos ejecutan las instrucciones de las threads SIMD, ejecutando un warp (un grupo de 32 threads, la unidad mínima de escalonamiento) a cada 4 ciclos. Eso permite que cada multiprocesador posea apenas una unidad de instrucciones, que envía la instrucción corriente para todos los núcleos.

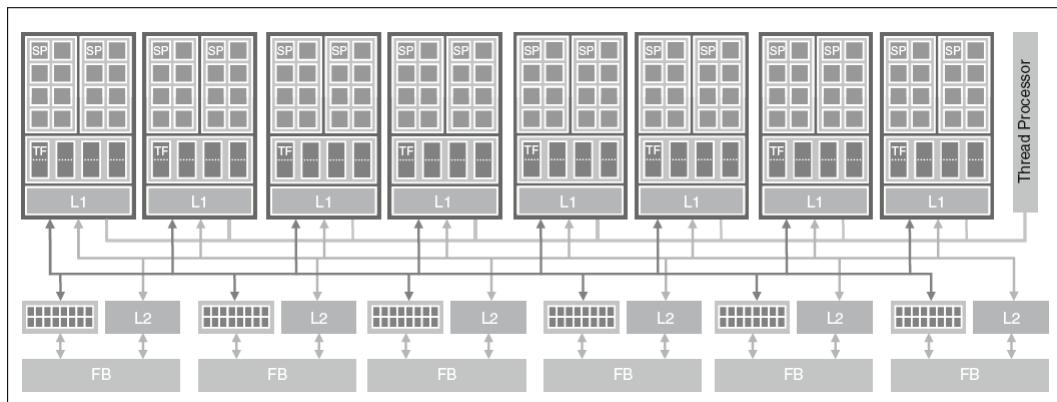


Figura 3.19: La arquitectura CUDA muestra un conjunto de procesadores de unidades programables de una NVIDIA GeForce GTX 8800. La arquitectura CUDA muestra un conjunto de procesadores de unidades programables de una NVIDIA GeForce GTX 8800. [Ocsa Mamani, 2015]

3.5.2. Aplicaciones

En esta sección, son discutidas las investigaciones relacionadas a las aplicaciones de propósito general usando GPUs, con foco especial en las operaciones de procesamiento de consultas en bases de datos y también en las tareas de minería de datos.

Operaciones de procesamiento de consultas en Base de Datos utilizando GPUs

La capacidad computacional de las GPUs ha sido explorada en el procesamiento de consultas y operaciones relacionadas. Por ejemplo, en [Zhao et al., 2014] demostraron que bloques de construcción importantes para el procesamiento de consultas en base de datos, como la ordenación, la selección conjuntiva, la agregación y las consultas sub-lineales pueden ser aceleradas significativamente con el uso de GPUs.

Además de eso, dos trabajos recientes [Lieberman and Miller, 2016] abordan el tema de unión por similitud en espacio n-dimensional para pares de objetos a partir de dos diferentes conjuntos R y S , cumpliendo de esa forma cierto predicado de unión. El predicado de unión más común es el $\in -join$ que determina todos los pares de objetos en una distancia menor que un $threshold \in$ predefinido.

La capacidad computacional de las GPUs también ha sido explorada para mejorar consultas kNN. Para exemplificar, en [Garcia et al., 2016] se describe una solución de consultas kNN usando el enfoque de fuerza bruta, además se utiliza una implementación altamente paralela en CUDA. En otro enfoque similar, [He et al., 2018] presenta el método CUKNN. El CUKNN resulta en un aumento de velocidad media de 46 veces, comparándose con el método secuencial de búsqueda kNN, basado en *quick sort*.

En otro trabajo, consultas kNN y AllkNN basadas en $kd - trees$ fueron propuestas en [Yu et al., 2016]. Debido a las restricciones del modelo de programación de CUDA, se

utilizaran arrays con el mismo enfoque del heapsort para representar los arboles binarios. Esa restricción resulta en la construcción de árboles en CPU para que luego se representen en un array en GPU, lo que limita el desempeño, pues para poder minimizar el costo de transferencia de datos, se prefiere implementaciones en CUDA con la mayoría de las etapas de pipeline del programa en GPU. Otra restricción de esa implementación es que solo trabaja para datos en dos o tres dimensiones.

Estructuras de datos espaciales y métricas pueden ser utilizadas para acelerar las consultas kNN, sin embargo la arquitectura CUDA no fue proyectada para procesar estructuras de datos complejas. Es por esa razón que hay poca motivación para que se desarrolle estructuras de datos para búsquedas kNN en GPU. [Ocsa Mamani, 2015]

Consultas kNN aproximadas en CUDA

Los algoritmos para la identificación de patrones frecuentes y datos irregulares especialmente los K patrones más frecuentes del conjunto de datos, especialmente los K patrones más frecuentes del conjunto de datos, pueden ser procesados eficientemente, inclusive para grandes conjuntos de datos. Antes de eso, a partir del esquema de indexación CUDA-LSH [Ocsa Mamani, 2015], el algoritmo de fuerza bruta para descubrimiento de motifs y outliers fue mejorado usando implementaciones eficientes y paralelas de búsquedas AllkNN. La propuesta para identificación de motifs y outliers está basada en el algoritmo de fuerza bruta para la identificación de los *TopKMotifs*, *Random projection* y *Local Outlier factor*.

La relevancia del *motif* y *outliers* es definida por la densidad de los *clusters*, ósea, por el radio del *cluster* que contiene los k elementos más próximos entre sí.

3.5.3. LSH

El método *Locality Sensitive Hashing* (LSH) fue ideado para resolver eficientemente consultas por rango aproximada ($(1 + \epsilon)R_q(q, r)$) este tipo de consulta busca recuperar los objetos que se encuentran dentro de un radio de consulta $(1 + \epsilon) \times r$. La idea principal es que, si dos objetos son próximos en el espacio original, esos dos objetos tienden a permanecer próximos después de una operación de proyección escalar randomica. Así, dada una función hash $h(x)$ que mapea un objeto (x) de dimensión n a un valor unidimensional, la función es sensible a la localidad si la posibilidad de mapeamiento de dos objetos x_1, x_2 al mismo valor crece a la medida que la distancia $d(x_1; x_2)$ disminuye.

La técnica de indexación CUDA-LSH, desarrollada con la intención de proporcionar búsquedas de alto desempeño para consultas por similitud aproximada en conjuntos de datos multidimensionales. La estrategia a seguir es de crear una implementación masivamente paralela del esquema de indexación LSH considerando una estructura de índice multinivel. El enfoque, como el esquema LSH, está basado en la idea de proyección de objetos multidimensionales en espacios de búsqueda más simples, en las cuales se conserva la intra-similitud entre objetos. Sin embargo, al contrario del LSH secuencial, el CUDA-LSH es capaz de realizar las operaciones de construcción y la consulta en paralelo.

El LSH original se compone de L tablas *hash*, en la que, utilizando conjuntos independientes de las funciones de *hash*, cada tabla *hash* indexa el conjunto de datos entero, tal que objetos semejantes son almacenados en el mismo *bucket*. Por otro lado, el *HashFile* utiliza el enfoque de proyección aleatoria, a fin de mapear datos multidimensionales en valores reales, de modo que los datos son almacenados secuencialmente en orden creciente a su valor *hash*. De ese modo, los esquemas de indexación LSH pueden ser adaptados para trabajar con estructuras de datos simples en la GPU.

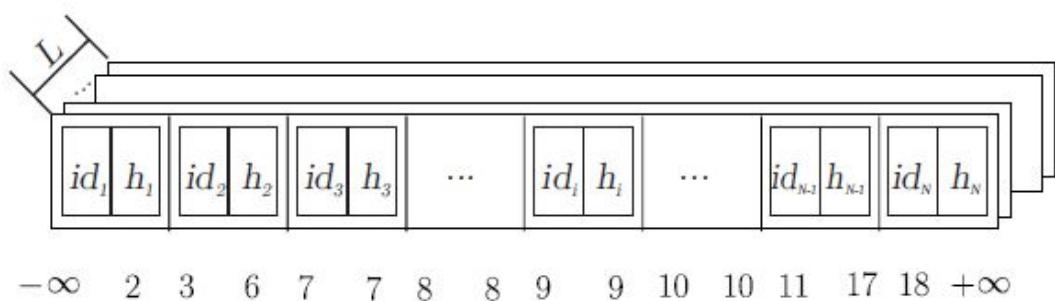


Figura 3.20: Estructura lógica del índice CUDA-LSH. [Ocsa Mamani, 2015]

Capítulo 4

Desarrollo

4.1. Caso de estudio

En esta sección describiremos como se realizó la interpolación de la información para la base de datos de Ucayali que fue usada como caso de estudio en este trabajo.

4.1.1. Base de Datos de Ucayali

En esta Sección hablaremos sobre la base de datos climatológica del departamento de Ucayali, Perú, ya que éste es el repositorio que utilizamos para nuestros experimentos.

Ucayali es una región de la selva del Perú , tiene un clima de bosque húmedo tropical y de lluvias abundantes

Es bueno resaltar que no logramos encontrar un repositorio libre en internet con información de los datos meteorológicos de esta región, sin embargo, logramos entrar en contacto con las personas que la administran y que muy amablemente nos brindaron la base de datos para poder realizar nuestros estudios.

En las siguientes Sub-secciones hablaremos sobre dos aspectos importantes de la base de datos. Primeramente en Subsubsección 4.1.1, describiremos las características generales



Figura 4.1: Departamento de Ucayali, ubicado en la parte de la selva peruana. Imagen extraída de [Ucayali,].

y algunas métricas de los registros de esta base de datos. Luego, en Subsubsección 4.1.1, tocaremos uno de los problemas más importantes que detectamos, que es las mediciones faltantes o registros incompletos. Para esto presentaremos el conjunto de pasos que seguimos para poder resolver este problema.

Especificaciones

La base de datos climatológica de Ucayali, como su nombre lo dice contiene información de las mediciones de condiciones climatológicas que se hicieron mediante el uso de distintos aparatos de medición.

Los datos que se capturaron son variados, y dentro de ellos podemos mencionar algunos como datos sobre temperatura, niveles de humedad y de calor, cantidad de rocío, datos sobre la velocidad y dirección del viento, frecuencia de lluvia, radiación solar, entre otros.

En Tabla 4.1, podemos observar el conjunto de todos los nombres de las mediciones en la base de datos.

Date	Time	TempOut	HiTemp	LowTemp	OutHum	DewPt	WindSpeed	WinRun
HiSpeed	WindChill	HeatIndex	THW Index	THSWIndex	Bar	Rain	RainRate	SolarRad
SolarEnergy	HiSolarRad	Index	UV Dose	UV	HiDD	HeatDD	CoolTemp	InHum
InDew	InHeat	InEMC	InDensity	InAirET	Samp	WindTX	WindRecept	ISSInt

Tabla 4.1: Nombres de las columnas de mediciones de la base de datos.

Las mediciones fueron programadas para capturar datos cada media hora, dentro de un periodo de 4 años, desde el 2012 al 2016. La cantidad total de registros que se lograron recolectar durante ese tiempo es de 61417 registros. Además el tipo de datos de todas las mediciones es numérico, a excepción de dos columnas que tienen información con las coordenadas del viento.

Ya que esta es una base de datos que contiene datos guardados a través del tiempo, decimos que toma el nombre de una base de datos de series temporales.

A manera de ejemplificación, en Figura 4.2, podemos ver un extracto de las filas 2534 – 2538 que contienen los datos climatológicos de la base de datos. En la figura no se muestran todos los campos de la base de datos, y no se encuentran graficados los registros entre dirección del viento(HiDir) y Humedad(InHum), ya que nuestro objetivo solo es dar una visión general del contenido de ésta base de datos climatológica.

		TempOut	HiTemp	LowTemp	OutHum	DewPt	WindSpeed	WindDir	\	
2012-02-22	19:30:00	25.2	25.9	25.2	89	23.2	0.0	---		
2012-02-22	20:00:00	24.7	25.2	24.7	92	23.3	0.0	---		
2012-02-22	20:30:00	24.4	24.7	24.4	93	23.2	0.0	---		
2012-02-22	21:00:00	24.3	24.4	24.3	94	23.2	0.0	---		
		WinRun	HiSpeed	HiDir	...	InHum	InDew	InHeat	InEMC	\
2012-02-22	19:30:00	0.0	0.0	---	...	71	23.0	32.8	12.89	
2012-02-22	20:00:00	0.0	0.0	---	...	72	23.0	32.1	13.25	
2012-02-22	20:30:00	0.0	0.0	---	...	73	23.0	31.8	13.59	
2012-02-22	21:00:00	0.0	0.0	---	...	74	23.1	31.7	13.90	
		InDensity	InAirET	Samp	WindTX	WindRecept	ISSInt			
2012-02-22	19:30:00	1.1302	0.0	701	1	100.0	30			
2012-02-22	20:00:00	1.1322	0.0	697	1	100.0	30			
2012-02-22	20:30:00	1.1341	0.0	684	1	100.0	30			
2012-02-22	21:00:00	1.1351	0.0	695	1	100.0	30			

[4 rows x 36 columns]

Figura 4.2: Extracto de los registros de medición de la base de datos de Ucayali.(elaboración propia)

Es importante notar que las columnas de Figura 4.2 tienen muchos registros incompletos. Por ejemplo en las columnas velocidad del viento(WindSpeed), velocidad mas mas de viento (HiSpeed) y InAirEt encontramos muchos registros con ceros. De igual manera en las columnas WindDir y HiDir, contienen registros que están completados con los caracteres '---'. Como mencionamos anteriormente, todos los registros que contienen '0' o '—' son considerados como datos faltantes y nuestro objetivo es buscar maneras para completarlos y así proceder con el análisis de patrones frecuentes y de datos irregulares posteriormente en esta tesis.

A continuación en Figura 4.3, se muestra la cantidad de registros faltantes en cada una de las columnas de la base de datos de Ucayali. Como vemos, estas cantidades varían desde algunos pocos a varios miles de registros faltantes.

Columna	# de datos faltantes
TempOut	3012
HiTemp	3012
LowTemp	3012
OutHum	8541
DewPt	8552
WindSpeed	46303
WinRun	46303
HiSpeed	32450
WindChill	3012
HeatIndex	3081
THW Index	3081
THSWIndex	11358
Bar	9
Rain	58624
RainRate	59529
SolarRad	30963
SolarEnergy	30963
HiSolarRad	30855
Index	37186
UV Dose	37186
UV	37002
HiDD	61146
HeatDD	3293
CoolTemp	9
InHum	9
InDew	9
InHeat	9
InEMC	9
InDensity	9
InAirET	40105
Samp	3013
WindTX	0
WindRecept	3013
ISSInt	0

Figura 4.3: Tabla con la cantidad de registros faltantes en cada columna de la base de datos de Ucayali

Otro de los datos importantes sobre la base de datos es algunas métricas para el análisis de la distribución de la data, como es el caso de la media y de la desviación estándar. Como sabemos la media representa el valor esperado y a su vez la tendencia central de la distribución de probabilidades de nuestros datos.

La desviación estándar es una medida que se usa para cuantificar la cantidad de variación o dispersión de un conjunto de datos. Una desviación estándar indica que los datos se encuentran cercanos a la media, la cual también es llamada como el valor esperado del conjunto de datos. Por otro lado una desviación estándar alta, indica que los valores se encuentran dispersos dentro de un rango amplio de valores.

En Figura 4.4, mostramos estas métricas de cada una de las columnas de la base de datos. Podemos observar que la mayoría de columnas tienen desviaciones estándar que indican que existe poca variación entre los datos de cada columna. En algunas otras como es el caso de SolarRad y HiSolarRad, estos valores son altos e indican que tienen un amplio rango en sus valores.

Columna	Media	Desviación Estándar
TempOut	26.033297	3.313239
HiTemp	26.255841	3.396611
LowTemp	25.814146	3.230024
OutHum	75.869809	19.379859
DewPt	20.792301	6.910655
WindSpeed	2.777829	1.840975
WinRun	1.393284	0.923797
HiSpeed	7.906483	4.87382
WindChill	26.032804	3.313909
HeatIndex	28.287437	5.650665
THW Index	28.286939	5.651183
THSWIndex	30.662932	8.006424
Bar	1007.812209	6.132134
Rain	1.841819	3.968764
RainRate	28.191102	104.568
SolarRad	370.415282	294.906925
SolarEnergy	15.930002	12.682594
HiSolarRad	500.370885	379.989168
Index	5.124423	3.511985
UV Dose	0.823321	0.564388
UV	6.265316	4.066309
HiDD	0.027465	0.020843
HeatDD	0.161316	0.067937
CoolTemp	28.283289	2.415037
InHum	72.113682	7.831786
InDew	22.660901	1.956081
InHeat	31.848642	4.305812
InEMC	13.635227	2.042652
InDensity	1.133498	0.015134
InAirET	0.200326	0.185275
Samp	694.540391	25.865936
WindTX	1	0
WindRecept	99.779481	3.561783
ISSInt	30	0

Figura 4.4: Tabla con la media y desviación estándar de las columnas de la base de datos climática de Ucayali.

Completación de la Base de Datos: Pandas

Como vimos en la Sub-sección anterior, nuestra base de datos contiene 61417 mediciones, sin embargo, este número corresponde a un número menor de registros para una frecuencia de una medición cada media hora. Esto puede deberse a errores físicos de los aparatos encargados de hacer estas mediciones, generando así espacios de registros faltantes. Esto se acrecienta cuando el periodo de tiempo de las mediciones es mayor.

En el caso de la base de datos de Ucayali, encontramos el mismo problema en un total de XXX registros faltantes. Es por eso que decidimos utilizar técnicas de completación de base de datos para reemplazar estos espacios incompletos.

Para el proceso de manejo de la información y de las distintas técnicas a aplicar sobre la data, decidimos utilizar una conocida librería llamada “Pandas”. Esta es una librería muy útil para facilitar el análisis de datos mediante el lenguaje de programación Python. Entre otras cosas, esta librería nos provee distintas herramientas para explorar y hacer operaciones sobre datos de series temporales en específico, como es el caso de la base de datos de Ucayali.

En Pandas, las series temporales son usualmente manejadas dentro de una estructura de datos llamada *DataFrame*. Esta estructura contiene columnas, las cuales pueden contener datos de diferentes tipos y además permite la opción de indexar toda esta data.

En la búsqueda de completar los datos faltantes de nuestra base de datos y tomando ventaja del *DataFrame*, en las siguientes Sub-secciones hablaremos sobre la serie de pasos que seguimos para llegar a este objetivo.

Primeramente hablaremos sobre la creación de índices del *DataFrame*, luego de la manera de representar nuestros datos faltantes, también hablaremos sobre el re-muestreo que se hizo a la base de datos y finalmente sobre las técnicas de interpolación para reemplazar

datos faltantes.

Todas estas técnicas serán mejor descritas en las siguientes Sub-secciones.

Creación de índices

Ya que el *DataFrame* de Pandas es una estructura de datos, eso significa que podemos crear uno o más índices especiales que nos sirvan para poder accesar las mediciones. El índice tiene que ser un valor único que identifique a un registro en específico, es por eso que decidimos utilizar la fecha y hora de la medición como el índice de cada registro de la base de datos.

Para poder generar un índice, tuvimos que juntar la información de las dos primeras columnas, llamadas respectivamente *Date* y *Time*, para así juntarlos dentro de un tipo de dato especial que permite manejar registros de fecha y hora en Panda llamado *datetime*.

Una vez ya generado el índice, podemos acceder fácilmente a cualquier registro de una fecha y hora específicas, además de poder hacer gráficos indexados por fecha.

Reemplazando valores faltantes

Una vez creado el índice, empezamos a enfocarnos en los datos faltantes. Ya que un dato faltante puede aparecer de varias maneras dentro de la base de datos, necesitamos volverlas a un formato estándar, de tal manera que un *DataFrame* pueda reconocerlas fácilmente.

En el caso de la base de datos de Ucayali, identificamos dos tipos de datos faltantes. Los primeros representados mediante (---) y el resto mediante *ceros*.

La manera más simple de indicarle al *DataFrame* que un registro es faltante, es reem-

plazándolo por un tipo de datos NaN. Una vez convertidos a NaN, podemos proceder a aplicar las distintas técnicas de completación de datos.

Haciendo re-muestreo de la data

En el proceso de análisis de la base de datos de Ucayali, nos dimos cuenta que la mayoría de mediciones fueron registradas con una frecuencia de medición de un registro cada media hora. Sin embargo, existían algunos intervalos mayores a media hora sin mediciones, es por eso de que se hizo un re-muestreo de la data para que tenga todos los registros con la frecuencia de media hora. Si es que deseamos crear más registros de los pre-existentes, lo llamamos *up-sample*, si lo que deseamos es reducir el número de registros a una frecuencia menor, entonces decimos que se trata de un *down-sample*.

Si es que hacemos *down-sample*, los datos nuevos se llenan automáticamente con valores NaN. Caso contrario, si es que se pretende reducir el número de registros, se puede llevar a cabo utilizando la media aritmética para hacer operaciones entre filas.

En éste caso, se lleva a cabo el *down-sample* y la data resultante pasará por un proceso de interpolación que se explicará en la siguiente sub-sección.

Interpolación de datos

La interpolación es área dentro de la matemática y la estadística que alberga a un conjunto de métodos que buscan construir nuevos valores o valores desconocidos a partir de un conjunto de datos conocidos que se encuentran dentro de un rango de datos discreto. La interpolación puede ser usada para predecir valores desconocidos para datos climáticos como la temperatura, radiación, velocidad del viento, etc. o cualquier otro tipo de datos que tenga una representación numérica.

Ya que la librería Pandas trabaja con series temporales de datos, ésta nos brinda herramientas para poder realizar distintos tipos de interpolación. En el caso de series temporales, la interpolación es una técnica útil ya que las bases de datos en general muchas veces contienen registros faltantes, entonces es una buena técnica de pre-procesamiento el completar estos datos, tomando como base todos los datos anteriores, y una vez completado este proceso, proceder a aplicar otras técnicas de minería de datos como búsqueda de patrones o datos irregulares, pero sobre una base de datos ya completa.

Los objetos de tipo *DataFrame* tienen como método de interpolación por defecto a la interpolación lineal. Sin embargo, existen otros métodos de interpolación más elaborados que se encuentran disponible en la librería Pandas. Algunos de estos métodos son la interpolación cuadrática y polinomial y la interpolación *spline*.

La interpolación lineal, consiste en generar una serie de líneas o funciones lineales entre cada uno de los elementos de un conjunto de datos, de tal manera que si tenemos un valor x desconocido, basta con ubicar los dos puntos más cercanos en nuestro conjunto de datos que se encuentren en el límite izquierdo y derecho de x , y calcular y basado en la función lineal que los une.

La interpolación cuadrada y la interpolación polinomial son una generalización de la interpolación lineal. En vez de calcular una función lineal que une a los puntos, ahora tendremos una función polinomial de un grado mayor. Cuando la función es de orden dos lo llamamos interpolación cuadrada, cuando es 3 se le denomina interpolación cúbica y así sucesivamente. Una de las desventajas de la interpolación polinomial es que conforme el grado de la interpolación aumenta, también aumenta el costo computacional para calcularla.

Finalmente, la interpolación *spline* usa polinomios de bajo grado en cada uno de los intervalos que generan nuestros puntos, eligiendo funciones que se ajusten suavemente entre ellas.

Interpolación lineal	<code>df.interpolate(method='lineal')</code>
Interpolación polinomial	<code>df.interpolate(method='polynomial', order=2)</code>
Interpolación temporal	<code>df.interpolate(method='time')</code>

Figura 4.5: Algunos métodos de interpolación en la estructura *DataFrame* de Pandas

En el caso de la interpolación de series temporales, se vuelve muy costoso realizar un tipo de interpolación de grados mayores a 3 ya que la cantidad de datos es muy grande y el tiempo de procesamiento se vuelve inviable, al menos si es que se planea hacer la interpolación de manera secuencial.

Dado un *DataFrame* denominado *df*, los métodos para calcular algunos de estos tipos de interpolación se pueden ejecutar con los comandos que se muestran en la figura 4.5.

El método *time*, es específicamente para datos diarios o de mayor resolución e interpolan según la longitud del intervalo.

Al aplicar la función de interpolación, todos los valores en el *DataFrame* identificados con NaN se tomaron automáticamente como valores faltantes y se interpolaron.

Para la interpolación final, hicimos tres pruebas de tipos de interpolación, las cuales son interpolación lineal, polinomial y temporal. Para probar los resultados de la interpolación y ver cuál es la que mejor se acomoda a nuestros datos, tomamos una de las columnas de la base de datos con una alta cantidad de datos faltantes, en este caso la columna 'DewPt' que como vimos en figuras anteriores contiene 8552 registros vacíos. En la figura 4.6 podemos notar visualmente que la mayoría de los datos faltantes se encuentran entre los años 2016 y 2016.

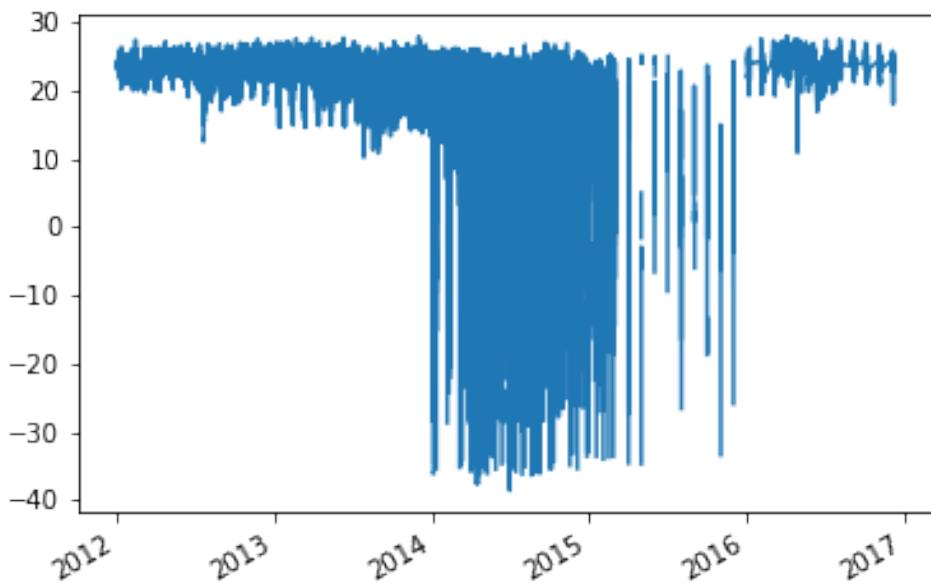


Figura 4.6: Gráfico de los valores originales de la columna 'DewPt' de la base de datos de Ucayali.

Los resultados de la interpolación de la columna 'DewPt' pueden verse en la figura 4.7.

En el caso de la interpolación lineal, podemos observar que produce un resultado suave que tiene equilibrio con la forma y la distribución general de los datos de los demás años.

Cuando vemos los resultados de la interpolación polinomial, que en este caso fue de grado 2, vemos que los resultados interpolados que se crean, llegan a tener valores extremos, tanto mínimos como máximos, siendo estos valores mínimos y máximos nunca antes vistos en los valores anteriores a la interpolación, generando valores únicos que se encuentran muy lejos de la distribución inicial de la data. Otro de los problemas de éste tipo de interpolación es que el tiempo de ejecución es mucho mayor conforme utilizamos técnicas de interpolación de grados más altos. Este fue el problema que tuvimos cuando intentamos correr la interpolación de grado dos sobre todo el *DataFrame*, lo cuál demoró demasiado en comparación a los otros métodos y que lo hace inviable para utilizarlo incluso en un conjunto de datos medianamente grande.

Finalmente, la interpolación temporal genera una interpolación que comparada con los resultados de la interpolación lineal, no resultan en una interpolación suave, sino más bien en valores con picos altos y altos bruscos.

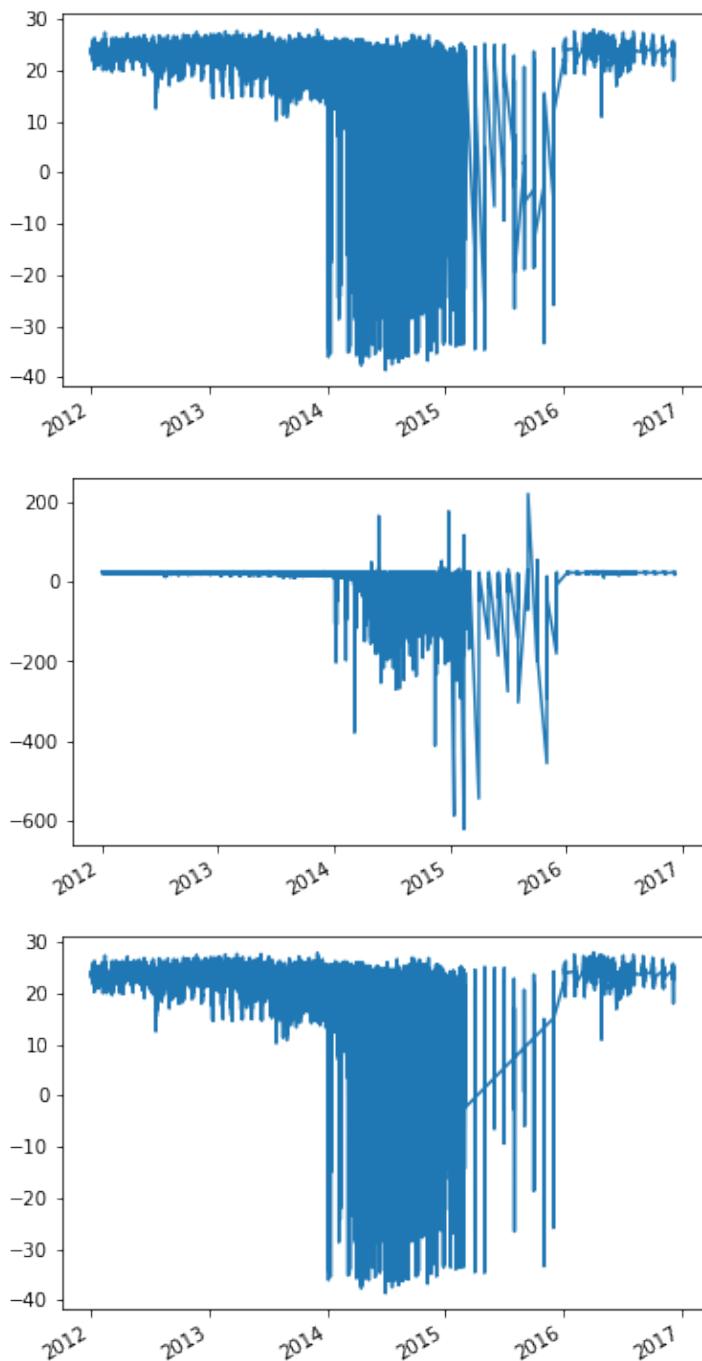


Figura 4.7: Distintos tipos de interpolación sobre la columna 'DewPt': Interpolación lineal, polinomial y temporal (De arriba hacia abajo).

En la figura 4.8 podemos observar un acercamiento de las interpolaciones obtenidas con los tres métodos de interpolación mencionadas anteriormente, pero más específicamente en el rango del 2015-2016. Elegimos hacer un acercamiento en este rango, ya que como mencionamos anteriormente, éste es el rango que contiene la mayoría de datos faltantes comparada al resto.

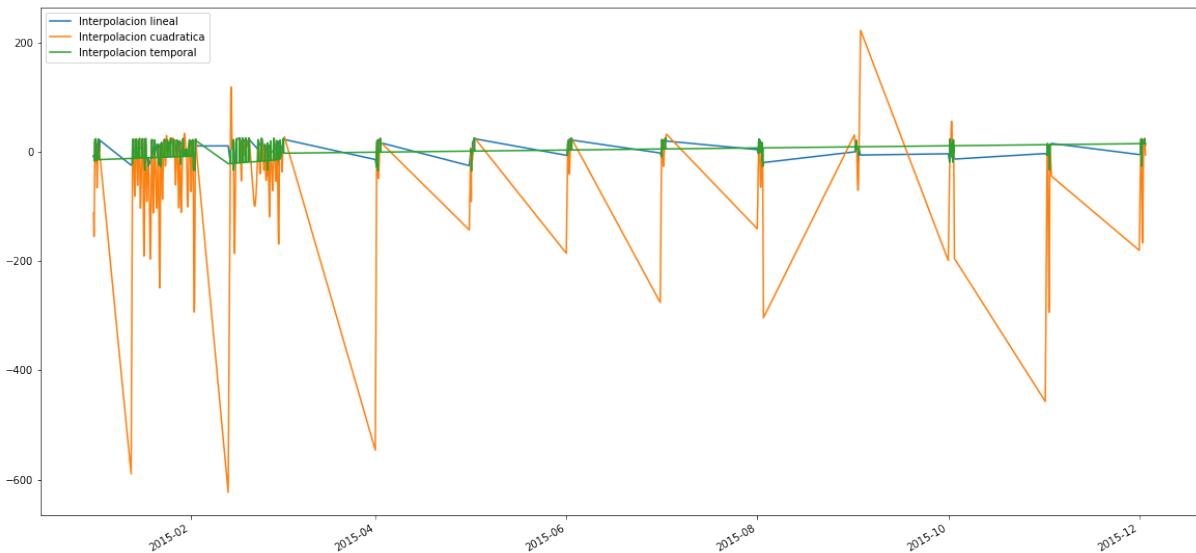


Figura 4.8: Acercamiento de la interpolación obtenida de la columna *DewPt* usando los 3 métodos, en el rango del 2015-2016. (Mejor visto en formato electrónico.)

Tomando en cuenta los experimentos con los 3 tipos de interpolación decidimos usar la interpolación lineal que es la más simple y produce resultados más naturales además de tener un tiempo de ejecución rápido.

Base de datos resultante

Ya que los datos originales de la base de datos se encuentran en gran parte con una frecuencia de un registro cada media hora, hicimos un proceso de re-muestreo de la data para una mejor visualización de los resultados. En las figuras 4.9 y 4.10 hemos resaltado los resultados de la interpolación, en escala de meses y de años respectivamente. En cada una de las imágenes se muestra graficada la línea de valores de cada una de las columnas

de la base de datos y diferenciadas una de la otra mediante los diferentes colores.

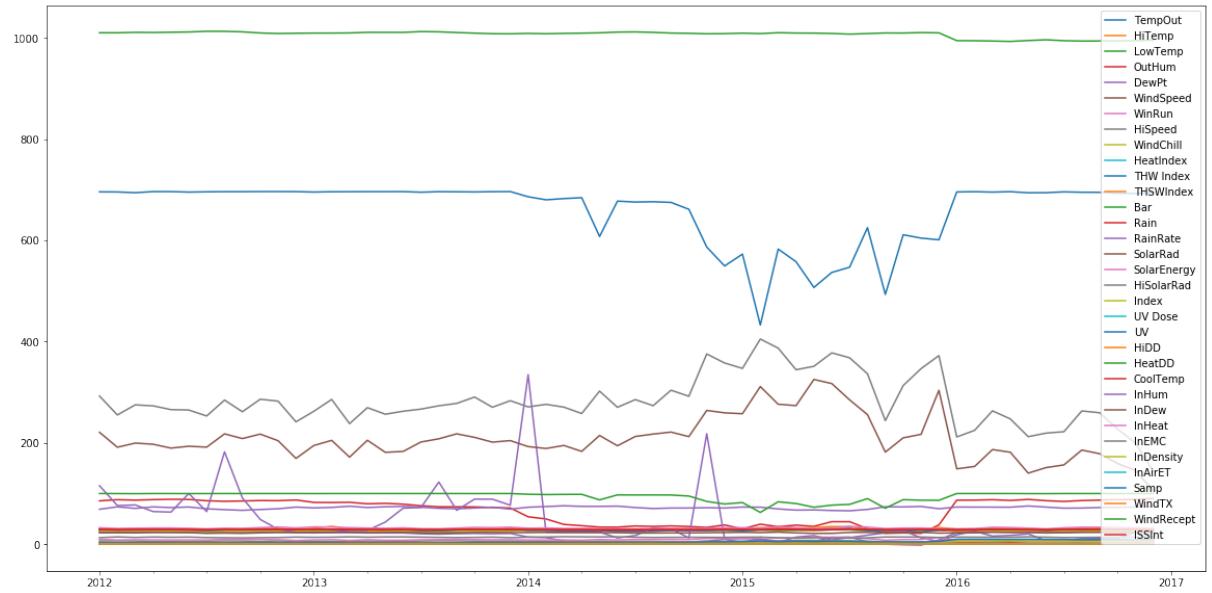


Figura 4.9: Muestreo mensual de la data (Mejor visto en formato electrónico).

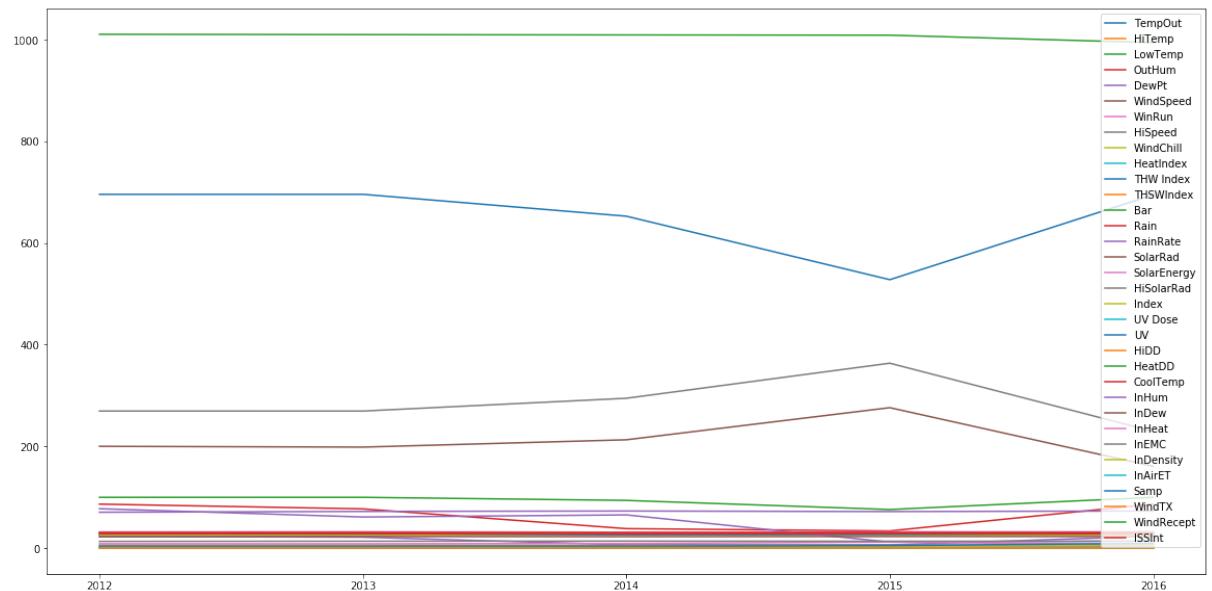


Figura 4.10: Muestreo anual de la data. (Mejor visto en formato electrónico).

4.1.2. Repositorio de la Amazonía Peruana

Para comenzar brindaremos una introducción de los repositorios que actualmente brindan información climatológica peruana (Subsubsección 4.1.2). Dado que existen pocos repositorios que a su vez ofrecen acceso a bases de datos con mediciones de data climatológica peruana.

Seguidamente en Subsubsección 4.1.2 presentamos el repositorio virtual que creamos sobre los datos climatológicos de la provincia de Ucayali, el cuál forma parte de uno de los aportes de este trabajo de tesis. Nuestro objetivo es brindar a otros investigadores esta base de datos, la cuál ya fue completada con los métodos que se mencionaron anteriormente, y que puedan utilizarla para su análisis en trabajos futuros.

Repositorios Climatológicos Peruanos

Actualmente en Perú existen algunos repositorios en donde podemos encontrar información climática y estudios relacionados al medio ambiente. Por ejemplo, la página web del IIAP (Instituto de Investigaciones de la Amazonía Peruana) [RII,], ofrece un repositorio virtual de información sobre la amazonía peruana. Aquí podemos encontrar información sobre la investigación científica y tecnológica sobre recursos naturales en la amazonía peruana.

Por otro lado, el Repositorio Digital del Instituto Geofísico del Perú [RDI,], es un repositorio virtual, el cuál es un espacio del Instituto de Geofísica para poder compartir información de la investigación científica que se realiza en el campo de la geofísica y de ramas relacionadas aplicada al Perú. En este repositorio también encontramos trabajos

sobre sismos, clima, condiciones atmosféricas, desastres, etc.

Hablando específicamente de data climática, el Instituto Geofísico del Perú tiene una sección dedicada al clima dentro de su repositorio virtual, en donde podemos encontrar información climatológica de todos los departamentos del Perú. Sin embargo, la información que brinda de cada departamento es limitada, y a la vez sólo se encuentra registrada la información de ciertas provincias. Los datos climatológicos a los que tenemos acceso constan de una breve reseña de la región, además de promedios multianuales de temperaturas máximas y mínimas y finalmente métricas sobre la precipitación acumulada mensual.



Figura 4.11: Información climatológica de los departamentos del Perú. (Extraído de [RDI,].)

Como vemos en Figura 4.11, cada punto dentro de un departamento, representa las mediciones de una de provincia específica, y dado el número de puntos nos damos cuenta que no se tiene información climatológica de todas las provincias. A pesar de que ésta información de clima es muy útil para tener noción de las características climáticas de cada departamento, nos dimos cuenta que la información de cada departamento correspondiente a la temperatura y precipitación son bastante antiguos.

Para mencionar algunos ejemplos, tenemos registros de temperaturas y precipitaciones que datan de 1964 a 1980 para Abancay, de 1950 a 1991 para Arequipa, de 1953 a 1991 para Cuzco, etc.

Repository de Datos Propuesto

Como mencionamos anteriormente, como parte de los objetivos de la tesis proponemos un repositorio que contiene la información de la base de datos climatológica de Ucayali. Esta base de datos originalmente contenía un conjunto de datos faltantes el cuál era considerablemente grande. A manera de poder difundir el estudio de los datos climatológicos de Ucayali, en el repositorio propuesto, ponemos a disposición la base de datos ya completada utilizando los métodos de interpolación que mencionamos anteriormente.

A continuación mostramos algunas visualizaciones del tipo de información que hicimos pública en nuestro repositorio, el cuál puede ser accesado desde el siguiente link:¹.

En Figura 4.12, podemos apreciar la portada de la página web que creamos para nuestro repositorio. Como podemos ver, la estructura general de nuestra página web se conforma de: Bases de datos, gráficos, algoritmos, descarga, publicaciones y contacto.

¹<http://repositorio.netlify.com>

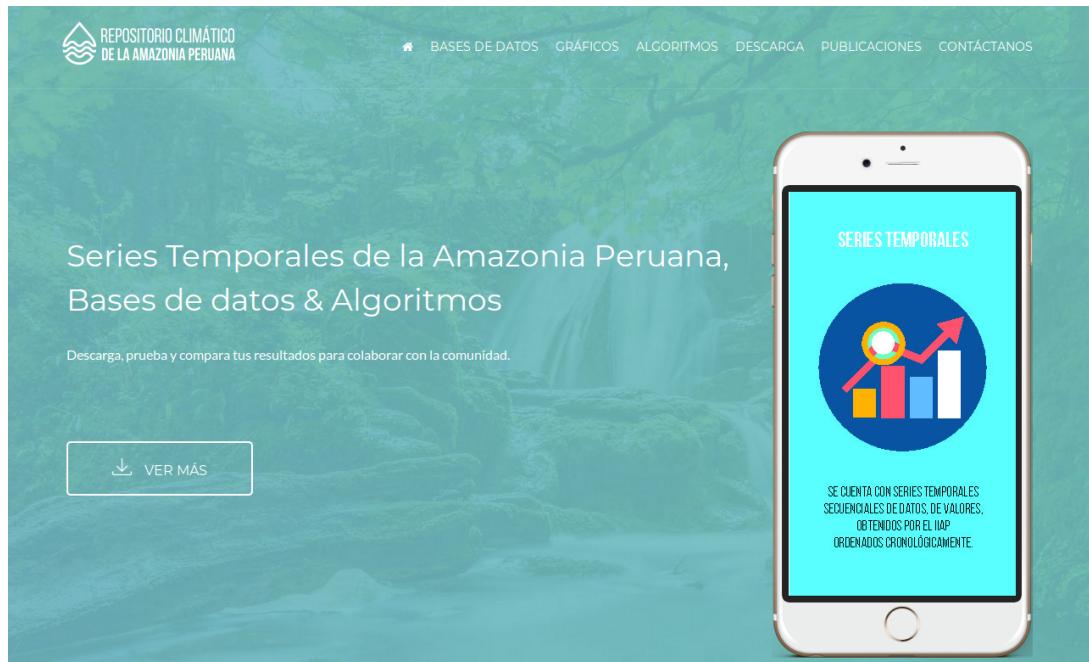


Figura 4.12: Página principal de nuestro repositorio. [RCA,]

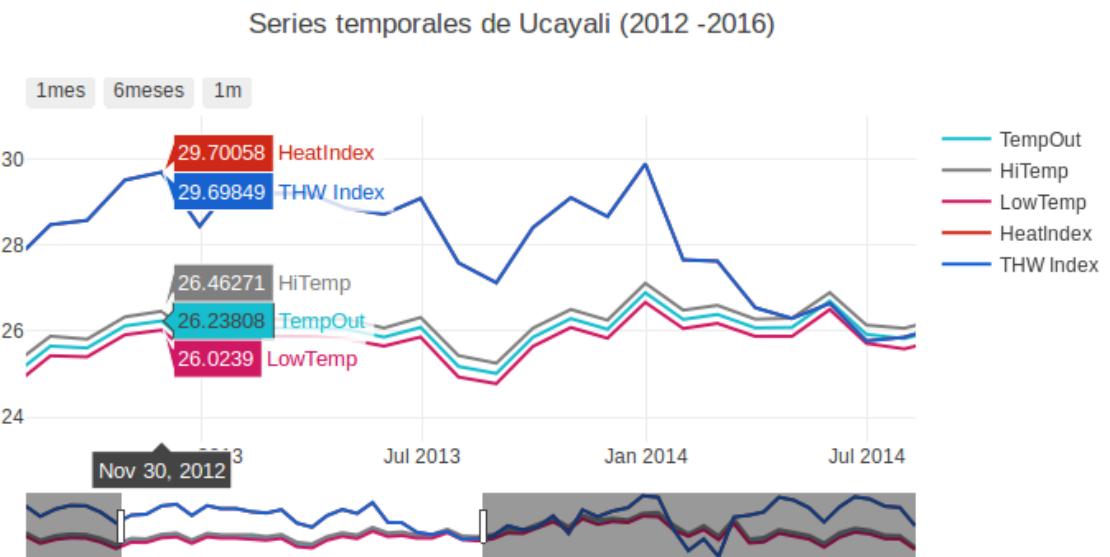
Para poder hacer fácil la visualización de los registros de la base de datos de Ucayali, adecuamos una tabla que los muestra de 10 en 10. Esto con el objetivo de facilitar una visualización rápida y general de los datos. En Figura 4.13, se muestra la base de datos tal y cuál como en nuestro repositorio web.

UCAYALI (2012 - 2016)

Show <input type="button" value="10"/> entries <input style="width: 150px; margin-left: 10px;" type="text"/> Search: <input style="width: 150px; margin-left: 10px;" type="text"/>						
	TempOut	HiTemp	LowTemp	OutHum	DewPt	WindSp
2012-01-31	26.292816091954034	26.50531609195402	26.082040229885056	85.5	23.4625718390805	2.24813
2012-02-29	25.482819548872207	25.678082706766975	25.291090225563885	87.80676691729323	23.140187969924796	2.27641
2012-03-31	25.956131650135273	26.186248872858464	25.73990081154196	86.9107303877367	23.414968440035995	2.14831
2012-04-30	26.14422032583399	26.3647013188518	25.920713731574896	87.93095422808379	23.807757951900697	1.92885
2012-05-31	26.093888888888912	26.312708333333372	25.876249999999995	88.61875	23.902430555555608	1.80758
2012-06-30	25.673353293413196	25.890818363273482	25.44231536926149	88.2624750499002	23.406886227544884	1.83423
2012-07-31	24.963382250174696	25.2116701607267	24.720684835779153	85.7267645003494	22.13885394828792	2.10117
2012-08-31	25.652638888888887	25.881805555555548	25.42729166666667	84.61875	22.648055555555565	2.34498
2012-09-30	25.608333333333363	25.813074712643644	25.401580459770116	85.23994252873563	22.72579022988507	2.22983

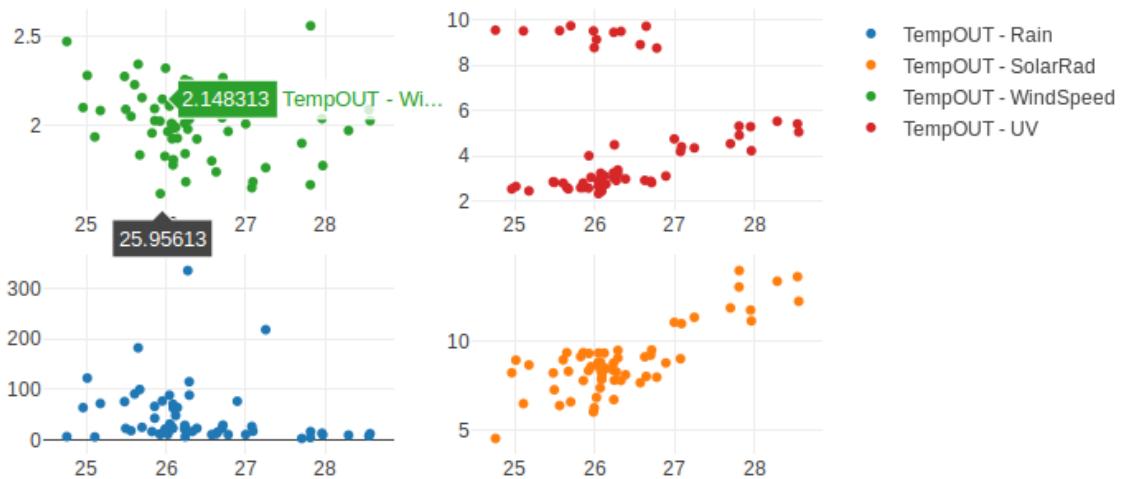
Figura 4.13: Extracto de la base de datos de Ucayali en nuestro repositorio [RCA,].

Como mencionamos, nuestra página también brinda mapas interactivos, los cuales permiten explorar los datos con facilidad. Por ejemplo, es posible elegir un periodo de tiempo específico en una línea de tiempo que se encuentra en cada gráfico y también podemos acceder a su valor exacto, sólo posicionando el puntero del ratón encima de su posición en el eje de coordenadas del mapa. Algunos ejemplos de estos mapas interactivos se muestran en Figura 4.14.



(a) Mapa de mediciones de cada columna.

Relaciones TempOut con Dimensiones(2012 -2016)



(b) Mapa de relaciones de la columna 'TempOut' con Dimensiones.

Figura 4.14: Mapas interactivos disponibles en nuestro repositorio. [RCA,].

Finalmente, brindamos acceso a la base de datos completada y a algunos de los algoritmos que fueron utilizados en esta tesis y que se brindan libremente para que otros investigadores lo puedan utilizar. La base de datos completada se puede encontrar en el

siguiente link:².

4.2. Experimentos

En la literatura de las áreas de bases de datos y la minería de datos, el método más común encontrado para validar nuevas técnicas esta basados en estudios experimentales. Como este trabajo tiene como objetivo investigar y desarrollar soluciones de alto desempeño de búsqueda por similitud aproximada y computación GPU para reducir el coste computacional de tareas de minería de datos que utilizan el kNN como procedimiento más importante, en general los experimentos implican:

- Análisis de los resultados obtenidos con conjuntos de datos reales y sintéticos para mostrar la ventaja de las técnicas para la búsqueda de similitud aproximada, con el objetivo de demostrar la eficacia y el rendimiento de las técnicas.

Es importante mencionar que las soluciones propuestas en este trabajo fueron diseñados para manejar conjuntos de series temporales que puedan ser definidos también como datos multidimensionales compuestos por atributos numéricos

Los experimentos presentados son:

- Experimentos con datos reales y sintéticos, discutiendo el rendimiento de las consultas y escalabilidad de los métodos exactos y aproximados para buscar por similitud aproximada en consultas por rango.

²<https://github.com/carloseab16/Bd-Climaticas>

4.2.1. Materiales y Métodos

Para detallar el proceso de análisis tanto de los vecinos más cercanos como de los *motifs* y *outliers* encontrados utilizando las técnicas propuestas, son evaluados los resultados de los estudios experimentales realizados con conjuntos de datos sintéticos y reales autorizados en la siguiente lista describe los conjuntos de datos utilizados

- **AUDIO:** Este conjunto de datos real contiene 54.387 vectores de dimensión 192. El conjunto de datos de audio proviene de la colección LDC SWITCHBOARD-13. Esta colección de series temporales tiene cerca de 2400 conversaciones telefónicas entre dos lados de 543 oradores de todas las áreas de los Estados Unidos: ³
- **RWALK:**Este conjunto de datos se compone de series aleatorias de 100.000 elementos de longitud 32. Las series de temporales fueron reproducidas siguiendo las instrucciones en el sitio MK-motif 5, utilizando la misma inicialización aleatoria: ⁴
- **AGRODATA:** Este conjunto de datos se proporciona mediante la colaboración de investigadores del *Centro de Pesquisas Meteorológicas e Climáticas Aplicadas à Agricultura*(Cepagri -Unicamp), y la Embrapa Informática Agropecuária de Campinas, en el contexto del proyecto AgroDataMine6. Estas series temporales climáticas fueron proporcionados originalmente por Agritempo7. Este conjunto de datos contiene 100,000 vectores de dimensión 24 correspondientes a las mediciones diarias de temperatura media realizadas cada día recogidas por 24 estaciones meteorológicas localizadas en el estado de Sao Paulo, Brasil, en período 1961-1990: ⁵
- **UCAYALI:** Este conjunto de datos se proporciona mediante la colaboración del IIAP y comprende datos climatológicos de 36 dimensiones reales del año 2012 al 2016 con una frecuencia de mediciones cada media hora.

³<http://www.ldc.upenn.edu/Catalog/docs/switchboard/>

⁴<http://www.cs.ucr.edu/~mueen/MK/>

⁵<http://gbdi.icmc.usp.br/agrodatamine/>

Los principales aspectos relacionados a los conjuntos de datos fueron:

- Los conjuntos de datos fueron seleccionados ya que se utilizan en la literatura para probar métodos de indexación y el descubrimiento de *motifs*.
- Dado que gran parte de los métodos para la identificación de los *motifs*, concretamente las basadas en representaciones SAX, exploran la naturaleza temporal de las series temporales apenas los conjuntos de datos AUDIO, Ucayali, RWALK y AGRODATA fueron considerados para evaluar el rendimiento de los algoritmos para la búsqueda de *motifs*;

Para la configuración del equipo se utilizó un servidor de aplicaciones de alta demanda (workstation) adquirido por el círculo de investigación de cambio climático, con las siguientes características un procesador INTEL CORE i7-6800K, tarjetas de vídeo NVI-DIA TITAN X, CUDA CORES GPU TX1080, 16 GB de memoria RAM, y una motherboard ASUS RAMPAGE V EXTREME. Este equipo especializado, nos asegura realizar eficientemente nuestros experimentos que consistieron en utilizar técnicas de patrones frecuentes y datos irregulares en conjuntos de datos de tiempo con el objetivo de demostrar la eficiencia de estas técnicas con diferentes arquitecturas para el procesamiento de datos en este caso, se realizaron pruebas en procesamiento secuencial CPU y en procesamiento paralelo GPU.

4.2.2. Métricas para la evaluación del desempeño

Los métodos de búsqueda de similitud y el descubrimiento de motifs y outliers implementados, fueron evaluados utilizando los siguientes parámetros para la comparación.

- El rendimiento de consultas: en este experimento se evaluó el rendimiento del enfoque propuesto en relación con las implementaciones paralelas de consultas AllkNN.

El objetivo de este experimento es medir el tiempo total necesario para recuperar los objetos más próximos a los objetos de consulta de una serie de pruebas utilizando consultas kNN.

El número de vecinos k a encontrarse en consultas kNN fueron elegidos de acuerdo a los valores comunes que se utilizan en situaciones reales. Debido a limitaciones de espacio en la GPU, en el caso de las consultas AllkNN el número de vecinos a ser encontrados fueron limitados a 25.

- Escalabilidad: Para evaluar la escalabilidad del método en consultas AllkNN, se implementó en dos versiones: GPU y CPU. En el primer experimento de escalabilidad se mantiene el tamaño del conjunto de datos mientras que el nivel de paralelismo aumenta. En el segundo experimento, se comparan los tiempos de implementación en CPU y GPU de las consultas AllkNN cuando el tamaño del conjunto de datos crece. El desempeño es evaluado mediante el speedup entre el tiempo de ejecución utilizando CPU y GPU.

$$speedup = \frac{Tiempo_CPU}{Tiempo_GPU} \quad (4.1)$$

4.2.3. Experimento 1: Descubrimiento de Patrones Frecuentes

En esta sección de los experimentos, mostraremos los tiempos de ejecución de dos de los algoritmos que elegimos para el descubrimiento de motifs llamados: Random Projection y Top-K motifs. Ambas técnicas de descubrimiento de motifs son muy utilizadas, sin embargo existe una diferencia sustancial en el tiempo de procesamiento, dependiendo de las implementaciones. En este caso hicimos pruebas entre dos implementaciones, tanto CPU como GPU y mostramos las diferencias.

Para correr nuestros experimentos utilizamos las bases de datos que describimos anteriormente: RWALK, AGRODATA, AUDIO y la base de datos climatológica de UCAYALI. Para ambos casos, como son la técnica de Random Projection y la de Top-K Motifs, brindamos tablas comparativas mostrando los tiempos de procesamiento en segundos. En Tabla 4.2 están los resultados para el algoritmo random projection y en Tabla 4.3 podemos ver la tabla comparativa de los tiempos obtenidos por el algoritmo TopK - Motifs. Los tiempos descritos en la tabla están en escala de milisegundos.

METHOD (Random Projection)	RWALK	AGRODATA	UCAYALI	AUDIO
CPU	16,755	11,571	13,374.5	2,440
GPU	2,234	1,446	1,486	278

Tabla 4.2: Tiempos de ejecución de algoritmo “Random Projection” en CPU y GPU, para la detección de motifs en los distintos datasets.

METHOD (TopKMotifs)	RWALK	AGRODATA	UCAYALI	AUDIO
CPU	118.014	63.81	133.759944	224.562
GPU	13.26	7.09	13.4028	28.79

Tabla 4.3: Tiempos de ejecución de algoritmo “TopKMotifs” en CPU y GPU, para la detección de motifs en los distintos datasets.

Para poder tener una mejor visualización de los métodos anteriormente descritos y hacer una comparación visual entre los tiempos obtenidos en cada una de las bases de datos. La visualización de los tiempos obtenidos por Random Projection se encuentra en

la Figura 4.15. A su vez, la visualización de los tiempos de descubrimiento de motifs se encuentra en Figura 4.16.

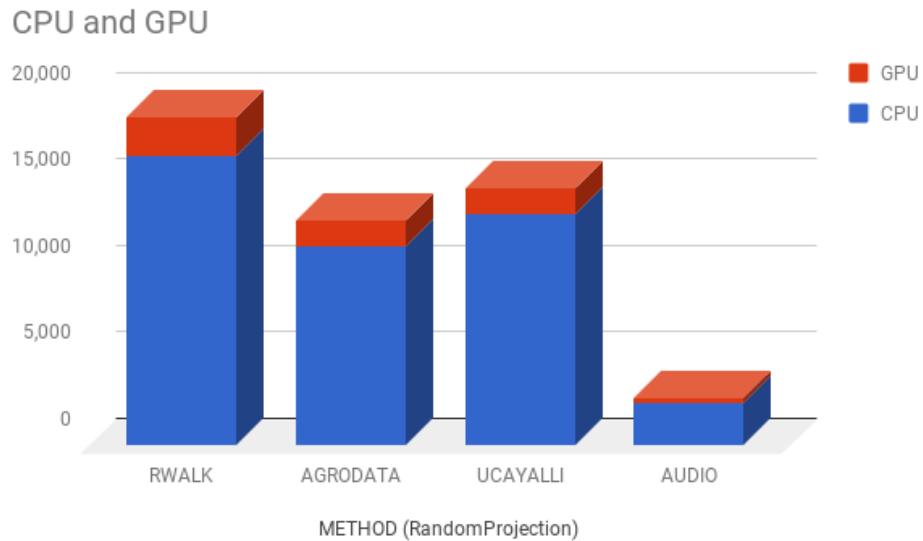


Figura 4.15: Tiempos de descubrimiento de motifs usando Random Projection en los distintos datasets.

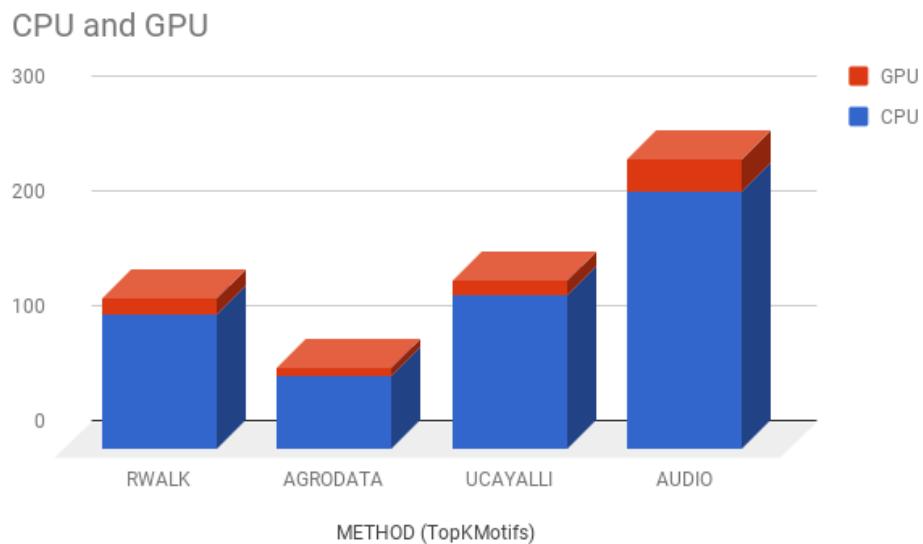


Figura 4.16: Tiempos de descubrimiento de motifs usando Top-K Motifs en los distintos datasets.

Como observamos en las figuras 4.15 y 4.16, los tiempos de procesamiento CPU para todas las bases de datos es sustancialmente mayor.

Para mostrar la mejora de tiempo de procesamiento de las implementaciones CPU y GPU entre los algoritmos Random Projection y Top-K Motifs, en las siguientes tablas mostraremos las métricas del speedup obtenido. Como explicamos en la sección anterior, el speedup es una métrica para cuantificar el nivel de escalabilidad entre nuestros algoritmos y se obtiene mediante la división entre el tiempo de procesamiento CPU y GPU. De esta manera podemos obtener el grado de mejora en tiempo de procesamiento entre estos dos enfoques. En la Tabla 4.4, se encuentran las métricas del speedup para el algoritmo Random Projection, y en la Tabla 4.5, observamos los tiempos de speedup para cada una de las bases de datos procesadas con el algoritmo Top-K Motifs.

Speedup	RWALK	AGRODATA	UCAYALI	AUDIO
METHOD (Random Projection)	7.50	8.00	9.00	8.79

Tabla 4.4: Speedup del algoritmo “Random Projection” comparando sus tiempos CPU y GPU.

Speedup	RWALK	AGRODATA	UCAYALI	AUDIO
METHOD (TopKMotifs)	8.9	9	9.98	7.8

Tabla 4.5: Speedup del algoritmo “Top-K Motifs” comparando sus tiempos CPU y GPU.

4.2.4. Experimento 2: Descubrimiento de Outliers

En esta parte de los experimentos hacemos una comparación de los tiempos de procesamiento obtenidos para el descubrimiento de outliers mediante el algoritmo Local Outlier Factor. En este caso mostramos la comparación de 3 tiempos de procesamiento. Primera-mente mostramos los tiempos obtenidos mediante la implementación secuencial CPU del

algoritmo LOF. Seguidamente, mostramos dos implementaciones, tanto CPU como GPU del mismo algoritmo usando la técnica de indexación mediante LSH. En estos experimentos, utilizamos las mismas bases de datos de la sección anterior en donde describimos los experimentos para el descubrimiento de motifs. La tabla con los tiempos de procesamiento para el LOF, se encuentran en la tabla Tabla 4.6.

	LOF (Local Outlier Factor)	RWALK	AGRODATA	UCAYALI	AUDIO
CPU	Fuerza bruta	180.5	153.8	4,301,495.3	242.8
	LSH-CPU	152.5	100.9	456,513.2	210.3
GPU	LSH-GPU	18.4	14.4	13,374.5	30.9

Tabla 4.6: Tiempos de ejecución de algoritmo “Local Outlier Factor” usando fuerza bruta y por indexación LSH en CPU y GPU, para la detección de outliers en los distintos datasets.

De igual manera, en la Figura 4.17 tenemos un gráfico en donde mostramos de manera visual las diferencias de tiempo de procesamiento usando los 3 enfoques: usando la fuerza bruta, el algoritmo LSH implementado en CPU y el algoritmo LSH implementado en GPU.

Brute-Force, LSH-CPU and LSH-GPU

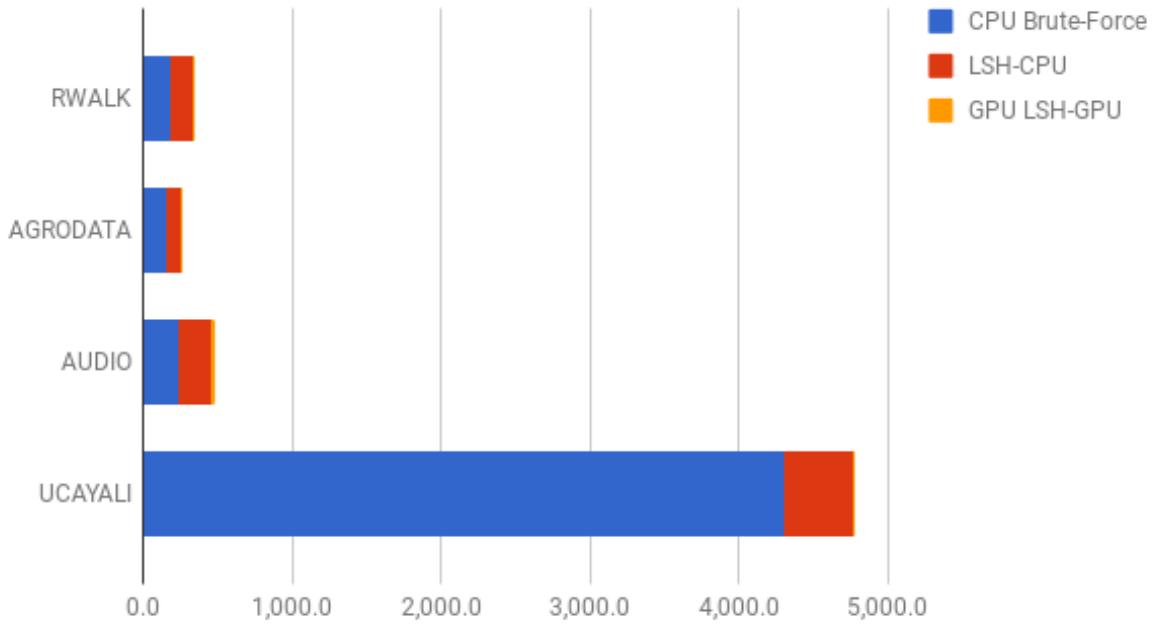


Figura 4.17: Tiempos de descubrimiento de outliers usando la técnica LOF en los distintos datasets.

Los resultados observados en Figura 4.17 son bastante contundentes. Como era de esperarse, el tiempo de procesamiento secuencial es el mayor, comparado con los tiempos obtenidos mediante LSH-CPU y LSH-GPU. Esta diferencia se destaca mucho más en el dataset de Ucayali, ya que el desempeño del algoritmo LSH-GPU es muchísimo más rápido que el algoritmo secuencial. Finalmente, podemos observar que en todos los datasets, el tiempo de procesamiento obtenido por el algoritmo LSH-GPU es notablemente inferior.

A continuación, en la Tabla 4.7, al igual que en los experimentos mostrados anteriormente para la detección de motifs, mostramos las métricas del speedup, para comparar cuanta fue la mejora en tiempo de procesamiento del algoritmo LSH en sus implementaciones en CPU y GPU.

Speedup	RWALK	AGRODATA	UCAYALI	AUDIO
METHOD (LSH)	8.3	7	6.8	34.1330572

Tabla 4.7: *Speedup* del algoritmo “LSH” para detección de *outliers* comparando sus tiempos CPU y GPU.

Como podemos observar, las mediciones de *speedup* para los *datasets* RWALK, AGRODATA y AUDIO, nos muestran la mejora de tiempos entre el algoritmo LSH en GPU comparado con el LSH en CPU. Además, para el caso del *speedup* obtenido para la base de datos de Ucayali es muchísimo mayor que el caso de los otros *datasets*, siendo de 34.13 milisegundos.

Capítulo 5

Conclusiones

En el área de recuperación de información, la búsqueda por similitud en datos complejos es un campo de investigación importante, ya que muchas tareas de minería de datos, como la clasificación, detección de agrupamientos, *Motif* y *outliers*, dependen de algoritmos de búsqueda a los vecinos más próximos.

No obstante, considerando las soluciones de búsqueda de patrones frecuentes como un proceso adecuado para el soporte en investigaciones para el cambio climático, muchas soluciones están siendo consideradas impracticables en contextos que el alto costo computacional es una limitante.

Para mejorar los tiempos de ejecución en operaciones de búsqueda y de tareas de minería de datos, se mostró que desarrollando soluciones de alto desempeño utilizando el paradigma de programación GPU se consigue mejorar el desempeño de estos en varias ordenes de magnitud.

En este trabajo fueron propuesto algoritmos de alto desempeño de algoritmos para identificar patrones frecuentes(*motifs*) y datos irregulares(*outliers*) los cuales usan métodos de búsqueda aproximada y los recursos del paradigma de programación GPGPU. computacional es una limitante.

Para ofrecer búsquedas por similitud de alto desempeño fue desarrollado el método de indexación de datos multidimensionales CUDA-LSH. De ese modo, utilizando implementaciones eficientes y paralelas de búsqueda AllkNN, así como la comparación entre distintas implementaciones en CPU y GPU.

En resumen, tres aspectos de este trabajo deben ser destacados. Lo primero fue una reconstrucción de una base de datos climática, segundo fue un estudio de los algoritmos de búsqueda por similitud, enfocado en los algoritmos aproximados. y el tercero fue la adaptación de algoritmos para arquitecturas paralelas en CPU y GPU y su comparación.

los métodos LSH, mostraron un mejor desempeño en relación a otros algoritmos exactos. No obstante, debido a las restricciones de estas implementaciones en la plataforma de programación CUDA, fue necesaria la adaptación y re-definición de las operaciones de construcción y consulta, así como la simplificación de las estructuras de datos a ser diseñadas en GPU.

De este modo, el método CUDA-LSH fue utilizado para explotar las técnicas aproximadas y los recursos de computación de alto desempeño proporcionados por la GPU. Se mostró que para poder alcanzar altos niveles de desempeño, el esquema de indexación debe ser diseñado de tal forma que la mayoría de etapas sean ejecutadas en la GPU, para poder minimizar el flujo de datos entre la CPU y la GPU.

5.1. Principales Contribuciones

Las principales contribuciones de este trabajo están relacionadas con :

1. La Re-construcción de una base de datos de series temporales climatológica del de-

partamento de Ucayali del 2012 al 2016, que fue completada mediante interpolación

2. Demostramos que los tiempos de ejecución en tareas de minería de datos en series temporales, integrando soluciones de alto desempeño utilizando GPU aceleran el tiempo computacional grandemente a comparación de otras soluciones con enfoque secuencial.
3. Colaborar con el estudio e investigación del cambio climático, mediante la publicación de nuestro repositorio de la base de datos climatológica de Ucayali .
4. Una publicación que se realizó durante la investigación.

5.2. Discusión

1. Reconocemos que son mas efectivas los métodos implementados en GPU, pero también reconocemos que el costo energético para su utilización es bajo, a comparación a varios CPUs que equivaldrían su trabajo.
2. Damos énfasis en que para utilizar técnicas en GPU, deben cumplir en ser simples y a su vez efectivas porque depende de la simpleza de la implementación la reconstrucción en GPU.
3. Recalcamos la necesidad de mayor investigación entorno a el cambio climático en el Perú. A la vez recalcar son muy pocas las bases de datos climatológicas peruanas en repositorios compartidos para investigación.

5.3. Trabajos Futuros

1. Debemos reconocer que este trabajo aporta en la elección de técnicas efectivas para el procesamiento de grandes volúmenes de datos climáticos, complejos y multidimensionales, por lo que en este trabajo no se logro utilizarlas para integrarlas con redes neuronales, para lograr aprendizaje de maquina y predicción, por lo que vemos que seria muy interesante realizar una investigación conjunta para lograr optimizar el tiempo de un proceso de predicción.
2. Reconocemos que la predicción no es el único fin para el uso de grandes volúmenes de datos, por lo que también identificamos como un trabajo futuro, el uso de otros fines para utilizar la base de datos y las técnicas, por ejemplo entender sucesos pasados y sus causas.

Bibliografía

[RCA,] Repositorio climático de la amazonía peruana. <http://repositorio.netlify.com>.

[RDI,] Repositorio digital del instituto geofísico del perú. <http://repositorio.igp.gob.pe/>.

[RII,] Repositorio institucional del instituto de investigaciones de la amazonía peruana. <http://repositorio.iiap.org.pe/>.

[Aftab et al.,] Aftab, S., Ahmad, M., Hameed, N., Bashir, M. S., Ali, I., and Nawaz, Z. Rainfall prediction in lahore city using data mining techniques.

[Agrawal et al., 1993] Agrawal, R., Imieliński, T., and Swami, A. (1993). Mining association rules between sets of items in large databases. In *Acm sigmod record*, volume 22, pages 207–216. ACM.

[Agrawal and Srikant, 1995] Agrawal, R. and Srikant, R. (1995). Mining sequential patterns. In *Data Engineering, 1995. Proceedings of the Eleventh International Conference on*, pages 3–14. IEEE.

[Asif et al., 2018] Asif, N., Malik, M., and Chaudhry, F. (2018). A review of on environmental pollution bioindicators. *Pollution*, 4(1):111–118.

[Astete et al., 2009] Astete, J., Cáceres, W., Gastañaga, M. d. C., Lucero, M., Sabastizagal, I., Oblitas, T., Pari, J., and Rodríguez, F. (2009). Intoxicación por plomo y otros problemas de salud en niños de poblaciones aledañas a relaves mineros. *Revista Peruana de Medicina Experimental y Salud Publica*, 26(1):15–19.

- [Balzter et al., 2015] Balzter, H., Tate, N. J., Kaduk, J., Harper, D., Page, S., Morrison, R., Muskulus, M., and Jones, P. (2015). Multi-scale entropy analysis as a method for time-series analysis of climate data. *Climate*, 3(1):227–240.
- [Bannayan and Hoogenboom, 2008] Bannayan, M. and Hoogenboom, G. (2008). Weather analogue: a tool for real-time prediction of daily weather data realizations based on a modified k-nearest neighbor approach. *Environmental Modelling & Software*, 23(6):703–713.
- [Bellman, 1962] Bellman, R. (1962). Dynamic programming treatment of the travelling salesman problem. *Journal of the ACM (JACM)*, 9(1):61–63.
- [Ben-Gal I., 2005] Ben-Gal I., Maimon O., R. L. e. (2005). Data mining and knowledge discovery handbook: Outlier detection.
- [Bentley and Ottmann, 1979] Bentley, J. L. and Ottmann, T. A. (1979). Algorithms for reporting and counting geometric intersections. *IEEE Transactions on computers*, 100(9):643–647.
- [Berndt and Clifford, 1994] Berndt, D. J. and Clifford, J. (1994). Using dynamic time warping to find patterns in time series. In *KDD workshop*, volume 10, pages 359–370. Seattle, WA.
- [Bezdek et al., 1984] Bezdek, J. C., Ehrlich, R., and Full, W. (1984). Fcm: The fuzzy c-means clustering algorithm. *Computers & Geosciences*, 10(2-3):191–203.
- [Blott and Weber, 2008] Blott, S. and Weber, R. (2008). What’s wrong with high-dimensional similarity search? *PVLDB*, 1(1):3.
- [Böhm et al., 2001] Böhm, C., Berchtold, S., and Keim, D. A. (2001). Searching in high-dimensional spaces: Index structures for improving the performance of multimedia databases. *ACM Computing Surveys (CSUR)*, 33(3):322–373.

- [Borah and Nath, 2018] Borah, A. and Nath, B. (2018). Fp-tree and its variants: Towards solving the pattern mining challenges. In *Proceedings of First International Conference on Smart System, Innovations and Computing*, pages 535–543. Springer.
- [Breunig et al., 2000] Breunig, M. M., Kriegel, H.-P., Ng, R. T., and Sander, J. (2000). Lof: identifying density-based local outliers. In *ACM sigmod record*, volume 29, pages 93–104. ACM.
- [Brigham and Brigham, 1988] Brigham, E. O. and Brigham, E. O. (1988). *The fast Fourier transform and its applications*, volume 1. prentice Hall Englewood Cliffs, NJ.
- [Buhler and Tompa, 2001] Buhler, J. and Tompa, M. (2001). Finding motifs using random projections. In *Proceedings of the annual international conference on Computational biology*, pages 69–76, New York, NY, USA.
- [Casas Luna and Mejía Marcacuzco, 2017] Casas Luna, S. A. and Mejía Marcacuzco, J. A. (2017). Modelo autorregresivo de primer orden aplicado a la predicción anual de caudales en la amazonia peruana: cuenca del río mayo. In *I International Congress on Water and Sustainability*. UPC. Departament d’Enginyeria de Projectes i de la Construcció.
- [Chan and Fu, 1999] Chan, K.-P. and Fu, A. W.-C. (1999). Efficient time series matching by wavelets. In *Data Engineering, 1999. Proceedings., 15th International Conference on*, pages 126–133. IEEE.
- [Chen et al., 2010] Chen, Y., Miao, D., and Zhang, H. (2010). Neighborhood outlier detection. *Expert Systems with Applications*, 37(12):8745–8749.
- [Chi et al., 2003a] Chi, B., Keogh, E., and Lonardi, S. (2003a). Probabilistic discovery of time series motifs. In *Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 493–498. ACM.

- [Chiu et al., 2003b] Chiu, B., Keogh, E., and Lonardi, S. (2003b). Probabilistic discovery of time series motifs. pages 493–498, New York, NY, USA.
- [Domingues et al., 2018] Domingues, R., Filippone, M., Michiardi, P., and Zouaoui, J. (2018). A comparative evaluation of outlier detection algorithms: Experiments and analyses. *Pattern Recognition*, 74:406–421.
- [Eddy, 1996] Eddy, S. R. (1996). Hidden markov models. *Current opinion in structural biology*, 6(3):361–365.
- [Estévez et al., 2018] Estévez, J., Gavilán, P., and García-Marín, A. (2018). Spatial regression test for ensuring temperature data quality in southern spain. *Theoretical and Applied Climatology*, 131(1-2):309–318.
- [Faghmous and Kumar, 2014] Faghmous, J. H. and Kumar, V. (2014). Spatio-temporal data mining for climate data: Advances, challenges, and opportunities. In *Data mining and knowledge discovery for big data*, pages 83–116. Springer.
- [Fayyd et al., 1996] Fayyd, U. M., Shapiro, G. P., and Smyth, P. (1996). From data mining to knowledge discovery: An overview.
- [Fell et al., 1993] Fell, J., Röschke, J., and Beckmann, P. (1993). Deterministic chaos and the first positive lyapunov exponent: a nonlinear analysis of the human electroencephalogram during sleep. *Biological cybernetics*, 69(2):139–146.
- [Fu, 2011] Fu, T.-c. (2011). A review on time series data mining. *Engineering Applications of Artificial Intelligence*, 24(1):164–181.
- [Gad et al., 2017] Gad, I., Manjunatha, B., et al. (2017). Performance evaluation of predictive models for missing data imputation in weather data. In *Advances in Computing, Communications and Informatics (ICACCI), 2017 International Conference on*, pages 1327–1334. IEEE.

- [Ganguly and Steinhaeuser, 2008] Ganguly, A. R. and Steinhaeuser, K. (2008). Data mining for climate change and impacts. In *Data Mining Workshops, 2008. ICDMW'08. IEEE International Conference on*, pages 385–394. IEEE.
- [Garcia et al., 2016] Garcia, M.-P., Bert, J., Benoit, D., Bardies, M., and Visvikis, D. (2016). Accelerated gpu based spect monte carlo simulations. *Physics in Medicine & Biology*, 61(11):4001.
- [Garcia-Molina, 2008] Garcia-Molina, H. (2008). *Database systems: the complete book*. Pearson Education India.
- [Guttman, 1984] Guttman, A. (1984). *R-trees: a dynamic index structure for spatial searching*, volume 14. ACM.
- [Han et al., 2011] Han, J., Pei, J., and Kamber, M. (2011). *Data mining: concepts and techniques*. Elsevier.
- [Han et al., 2000] Han, J., Pei, J., Mortazavi-Asl, B., Chen, Q., Dayal, U., and Hsu, M.-C. (2000). Freespan: frequent pattern-projected sequential pattern mining. In *Proceedings of the sixth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 355–359. ACM.
- [He et al., 2018] He, Z., Lei, L., Welp, L. R., Zeng, Z.-C., Bie, N., Yang, S., and Liu, L. (2018). Detection of spatiotemporal extreme changes in atmospheric co₂ concentration based on satellite observations. *Remote Sensing*, 10(6):839.
- [Hennemuth et al., 2015] Hennemuth, B., Bender, S., Bülow, K., Dreier, N., Hoffmann, P., Keup-Thiel, E., and Mudersbach, C. (2015). Collecting statistical methods for the analysis of climate data as service for adaptation projects. *American Journal of Climate Change*, 4(01):9.
- [Hudson and Quade, 2013] Hudson, A. M. and Quade, J. (2013). Long-term east-west asymmetry in monsoon rainfall on the tibetan plateau. *Geology*, 41(3):351–354.

[IIAP, 2018] IIAP (2018). *Instituto de investigacion de la Amazonia Peruana*.

[Jakhar et al., 2018] Jakhar, Y. K., Mishra, N., and Poonia, R. (2018). Predication accuracy analysis of data mining algorithms on meteorological data using r programming.

[Kantardzic, 2011] Kantardzic, M. (2011). *Data mining: concepts, models, methods, and algorithms*. John Wiley & Sons.

[Keogh et al., 2001] Keogh, E., Chakrabarti, K., Pazzani, M., and Mehrotra, S. (2001). Dimensionality reduction for fast similarity search in large time series databases. *Knowledge and information Systems*, 3(3):263–286.

[Kohonen, 2008] Kohonen, O. (2008). Fp-tree. <http://www.cis.hut.fi/Opinnot/T-61.6020/2008/fptree.pdf>.

[Korn et al., 1997] Korn, F., Jagadish, H. V., and Faloutsos, C. (1997). Efficiently supporting ad hoc queries in large datasets of time sequences. In *Acm Sigmod Record*, volume 26, pages 289–300. ACM.

[Koshti and Patel, 2018] Koshti, S. and Patel, A. (2018). Frequent pattern mining using genetic algorithm in data mining.

[Kriegel et al., 2010] Kriegel, H.-P., Kröger, P., and Zimek, A. (2010). Outlier detection techniques. *Tutorial at KDD*, 10.

[Kumar et al., 2001] Kumar, V., Steinbach, M., Tan, P.-N., Klooster, S., Potter, C., and Torregrosa, A. (2001). Mining scientific data: Discovery of patterns in the global climate system. In *Joint Statistical Meeting*.

[Lieberman and Miller, 2016] Lieberman, A. and Miller, L. (2016). Learning communities: Harness the energy of collaboration. *The Learning Professional*, 37(1):14.

[Lin et al., 2007] Lin, J., Keogh, E., Wei, L., and Lonardi, S. (2007). Experiencing SAX: a novel symbolic representation of time series. *Data Mining and Knowledge Discovery*, 15(2):107–144.

[Mannila et al., 1997] Mannila, H., Toivonen, H., and Verkamo, A. I. (1997). Discovery of frequent episodes in event sequences. *Data mining and knowledge discovery*, 1(3):259–289.

[Mark Hall, 2009] Mark Hall, Eibe Frank, G. H. B. P. (2009). The weka data mining software: An update. 11.

[Marple and Marple, 1987] Marple, S. L. and Marple, S. L. (1987). *Digital spectral analysis: with applications*, volume 5. Prentice-Hall Englewood Cliffs, NJ.

[Marsland et al., 2002] Marsland, S., Shapiro, J., and Nehmzow, U. (2002). A self-organising network that grows when required. *Neural networks*, 15(8-9):1041–1058.

[McGuffie and Henderson-Sellers, 2005] McGuffie, K. and Henderson-Sellers, A. (2005). *A climate modelling primer*. John Wiley & Sons.

[Mudelsee et al., 2014] Mudelsee, M., Bickert, T., Lear, C. H., and Lohmann, G. (2014). Cenozoic climate changes: A review based on time series analysis of marine benthic $\delta^{18}\text{O}$ records. *Reviews of Geophysics*, 52(3):333–374.

[Mueen et al., 2009] Mueen, A., Keogh, E. J., Zhu, Q., Cash, S., and Westover, B. (2009). Exact Discovery of Time Series Motifs. pages 473–484, Sparks, NV, USA.

[Nickolls et al., 2008] Nickolls, J., Buck, I., Garland, M., and Skadron, K. (2008). Scalable Parallel Programming with CUDA. *Queue*, 6:40–53.

[NOAA, 2018] NOAA (2018). *Datos Históricos de NOAA*.

[Ocsa Mamani, 2015] Ocsa Mamani, A. V. (2015). *Soluciones Aproximadas Para Algoritmos Escalables De Mineracion De Datos En Dominios Complejos*. Universidad Nacional de San Agustín.

[Olaiya and Adeyemo, 2012] Olaiya, F. and Adeyemo, A. B. (2012). Application of data mining techniques in weather prediction and climate change studies. *International Journal of Information Engineering and Electronic Business*, 4(1):51.

- [Owens et al., 2007] Owens, J. D., Luebke, D., Govindaraju, N., Harris, M., Krüger, J., Lefohn, A. E., and Purcell, T. J. (2007). A survey of general-purpose computation on graphics hardware. In *Computer graphics forum*, volume 26, pages 80–113. Wiley Online Library.
- [Paredes et al., 2015] Paredes, F., Barbosa, A., Guevara, E., et al. (2015). Análisis espacial y temporal de las sequías en el noreste de brasil. *Agriscientia*, 32(1):1–14.
- [Perego et al., 2001] Perego, R., Orlando, S., and Palmerini, P. (2001). Enhancing the apriori algorithm for frequent set counting. In *International Conference on Data Warehousing and Knowledge Discovery*, pages 71–82. Springer.
- [Quinn and Sugiyama, 2014] Quinn, J. A. and Sugiyama, M. (2014). A least-squares approach to anomaly detection in static and sequential data. *Pattern Recognition Letters*, 40:36–40.
- [Ramaswamy et al., 2000] Ramaswamy, S., Rastogi, R., and Shim, K. (2000). Efficient algorithms for mining outliers from large data sets. In *ACM Sigmod Record*, volume 29, pages 427–438. ACM.
- [Sellis et al., 1987] Sellis, T., Roussopoulos, N., and Faloutsos, C. (1987). The r+-tree: A dynamic index for multi-dimensional objects.
- [Singh and Stuart, 1998] Singh, S. and Stuart, E. (1998). A pattern matching tool for time-series forecasting. In *Pattern Recognition, 1998. Proceedings. Fourteenth International Conference on*, volume 1, pages 103–105. IEEE.
- [sociodemográfico del Perú, 2008] sociodemográfico del Perú, P. (2008). Censos nacionales 2007: Xi de población y vi de vivienda.
- [Sridevi et al., 2018] Sridevi, S., Parthasarathy, S., and Rajaram, S. (2018). An effective prediction system for time series data using pattern matching algorithms. *International Journal of Industrial Engineering*, 25(2).

- [Steinbach et al., 2003] Steinbach, M., Tan, P.-N., Kumar, V., Klooster, S., and Potter, C. (2003). Discovery of climate indices using clustering. In *Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 446–455. ACM.
- [Stone et al., 2010] Stone, J., Gohara, D., and Shi, G. (2010). OpenCL: A Parallel Programming Standard for Heterogeneous Computing Systems. *Computing in Science Engineering*, 12:66 –73.
- [Suárez and Pérez, 2017] Suárez, C. B. Q. and Pérez, C. M. A. L. (2017). Las ciencias naturales en el cuidado y conservación del medio ambiente. *Revista Mapa*, 2(1).
- [Sun and Trevor, 2017] Sun, W. and Trevor, B. (2017). Combining k-nearest-neighbor models for annual peak breakup flow forecasting. *Cold Regions Science and Technology*, 143:59–69.
- [Tipping and Bishop, 1999] Tipping, M. E. and Bishop, C. M. (1999). Probabilistic principal component analysis. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 61(3):611–622.
- [Ucayali,] Ucayali. Departamento de ucayali.
- [Van Hulle, 2012] Van Hulle, M. M. (2012). Self-organizing maps. In *Handbook of Natural Computing*, pages 585–622. Springer.
- [Vergara et al., 2016] Vergara, S., Espinoza, J. C., Medina, B., et al. (2016). Eventos hidrológicos extremos en la amazonía peruana: Sistema de alerta cualitativo para la previsión. informe mensual: Abril de 2016.
- [VILLAR et al., 2010] VILLAR, J. C. E., Ronchail, J., LAVADO, W., CARRANZA, J., Cochonneau, G., DE OLIVEIRA, E., and POMBOSA, R. (2010). Variabilidad espaciotemporal de las lluvias en la cuenca amazónica y su relación con la variabilidad hidrológica regional. un enfoque particular sobre la región andina spatio-temporal rainfall

variability in the amazon basin and its relationship to regional hydrological variability. a particular focus in the andean region. *Revista Peruana Geo-atmosférica*, 2:99–130.

[Weiss and Indurkhya, 1998] Weiss, S. M. and Indurkhya, N. (1998). *Predictive data mining: a practical guide*. Morgan Kaufmann.

[Williams et al., 2002] Williams, G., Baxter, R., He, H., Hawkins, S., and Gu, L. (2002). A comparative study of rnn for outlier detection in data mining. In *Data Mining, 2002. ICDM 2003. Proceedings. 2002 IEEE International Conference on*, pages 709–712. IEEE.

[Yu et al., 2016] Yu, W., Zhou, W., Qian, Y., and Yan, J. (2016). A new approach for land cover classification and change analysis: Integrating backdating and an object-based method. *Remote Sensing of Environment*, 177:37–47.

[Zhao et al., 2014] Zhao, B., Zhong, J., He, B., Luo, Q., Fang, W., and Govindaraju, N. K. (2014). Gpu-accelerated cloud computing for data-intensive applications. In *Cloud Computing for Data-Intensive Applications*, pages 105–129. Springer.

Anexos

Articulo

Time Series Mining using Approximate Nearest Neighbor Search on GPUs

Carlos Arbieto¹, Jose Huillca¹, Alexander Ocsa¹, Ricardo Coronado¹, Cristian Lopez²

¹Universidad Nacional de San Agustín de Arequipa - Perú

email: carloseab16@gmail.com, jaselhuillca@gmail.com, aocsa.cs@gmail.com,
rikardo.corp@gmail.com,

²Universidad La Salle de Arequipa - Perú

email: criloal23@gmail.com

1. ABSTRACT

Motif discovery and outlier detection in time series databases is of prime importance and a major challenge in many areas, so exists a necessity of real-time algorithms. For this challenges, single CPU architectures impose limits on performance to deliver this kinds of solutions. Resorting to GPU programming is one approach to overcome these performance limitations. In this work, we discuss various time series motif discovery and outlier detection algorithms in order to select only those with highly parallelizable algorithm structure. More important, we present a comparative study among parallelizable exact and approximate algorithm to find motifs and outliers in very large time series databases. We use a NVIDIA CUDA framework to create a massively parallel implementation of the analyzed techniques. Experimental studies allow us to run experiments on large real and synthetic datasets thanks to the highly CUDA parallel implementation. Our performance evaluation on GeForce GTX 1080 GPU, result in average runtime speedups of 10 (15) using the MK (Random Projection) and Local Outlier Factor algorithms. The runtime speedups of CUDA is better than to those of Parallel MK and Parallel Random Projection running on 8 CPU cores of high-performance workstation.

Keywords: Locality Sensitive Hashing, High Performance Computing, GPU, Outlier detection, Motif discovery.

2. INTRODUCTION

Data Mining is a research area included in the broad context of Knowledge Discovery in Databases (KDD), whose main objective is to extract knowledge to be used in decision making from a dataset. Mining time series is a relatively new field, which includes data mining techniques adapted to take into consideration the nature of time series. In particular, the problem of discovering previously unknown frequent patterns in time series, also called motifs, is an challenging problem. Basically, because still there is a huge gap between the scalability of the methods in current literature and the needs of professionals in various fields. In real contexts, data can vary from a few megabytes to terabytes, so, scalable motif discovery algorithms are needed to manipulate and analyze data efficiently and effectively.

In this paper, we discuss various time series motif discovery and outlier detection algorithms in order to select only those with highly parallelizable algorithm structure. More important, we present a comparative study among parallelizable exact and approximate algorithm to find motifs in very large time series databases. We use a NVIDIA

CUDA framework to create a massively parallel implementation of the analyzed techniques. Experimental studies allows us to run experiments on large real and synthetic datasets thanks to the highly CUDA parallel implementation. Our performance evaluation on GeForce GTX 1080 GPU, result in average runtime speedups of 10 (15) using the MK (Random Projection) and Local Outlier Factor algorithms. The runtime speedups of CUDA is better than to those of Parallel MK and Parallel Random Projection running on 8 CPU cores of high-performance workstation.

3. RELATED WORK

3.1. *Motifs* discovery and Oulier detection

The *motifs* are the previously unknown patterns that occur most frequently in the data. These patterns can be of particular importance for other mining tasks time series, such as identification of groupings, discovery of association rules, anomalies identification and behavioral analysis. An efficient algorithm for finding *motifs* can also be useful as a tool for summarizing and displaying large amounts of data[5], and exceptions or outliers can be defined as “An observation which deviates so much from the other observations as to arouse suspicions that it was generated by a different mechanism”[4]. In this paper, we are going to use a method for discover outliers in a multidimensional dataset called “local outlier Factor(LOF)”, which quantifies how outlying an object is.

3.2. CUDA model programming

A CUDA program source code is a code covering both the *host* and the *device*. The implementation of a CUDA program execution begins with the *host*. When a kernel function *kernel* is released, the mode changes execution *device* device and generates a large number of *threads* to leverage the parallel architecture.

4. Aproximate Nearest Neighbor Search on GPU

In this section is described the indexing technique CUDA-LSH, developed to provide high performance queries for similarity searches of multidimensional data sets described. The next strategy is to create a parallel application of LSH indexing system considering an index structure of several levels. The approach is based LSH scheme on the idea projection multidimensional objects into simpler space search in which the intra-similarity between the objects is preserved. However, unlike the sequential LSH, CUDA-LSH is able to perform the operations and construction of the parallel query[6].

4.1. Approximate AllkNN queries

Solve the search problem AllkNN is not trivial, especially when working with large data sets of large dimension. They must also take into account the problems related to the implementation of efficient KNN [7] consultations, particularly the problem of the “curse of the high dimensionality”. Thus, based on an approximate approach, CUDA-LSH method is developed to provide efficient kNN searches CUDA has extended for AllkNN consultations. Due to the highly parallel nature of AllkNN consultations, it is possible to overcome some of the problems associated with the use of large data sets. Thus, supported by the indexing method CUDA-LSH a parallel solution for AllkNN search algorithm was developed.

4.2. Motif Discovery based on Approximate AllkNN queries

Once developed high-performance solution for approximate similarity searches, algorithms for identifying *motifs*, specifically the K more frequent patterns in the data set can be processed efficiently, even for large sets of data. Thus, from the indexing scheme CUDA-LSH, the brute force algorithm for discovery of *motifs* it has been enhanced through efficient and parallel implementations to search AllkNN. The algorithm 1 describes the pseudocode algorithm *CUDA-TopKMotifs* for identifying the *TopK-Motifs* using the algorithm AllkNN as the main procedure. Basically, the search algorithm kNN is replaced by the parallel search algorithm provided by the method of *CUDA-LSH*. We note that the algorithm for the identification of *motifs*, at first, prepare the indexing (Line 6-7), then, the AllkNN (line 8) query is executed, after that sorts the candidates based on the density of the *cluster* and finally prints the first *TopK motifs* (line 10-13).

Listing 1. Algorithm CUDA TopK-Motifs

```

1  algorithm CUDA-TopKMotifs ( $S, k, topK$ )
2    deviceMem LSH := initHashFunctions()
3    deviceMem  $S'[] := S[]$ 
4    threads := nThreads
5    blocks :=  $N/nThreads$ 
6    LSH-Kernel<<<blocks, threads>>> ( $S', LSH$ )
7    cuda_sort(LSH.keys, LSH.ids)
8    [ $ids, dists$ ] := AllkNN-Kernel<<<blocks, threads>>> ( $S', LSH, k$ )
9    cuda_sort(dists, ids)
10   for  $i := 1$  to  $topK$  do
11     result := kNN( $ids[i], k$ )
12     print_motif(result)
13   end for
14 end.

```

5. Outlier detection based on approximate kNN queries

The key difference between *Local Outlier Factor(LOF)* and the existing notions of Outliers, is that being isolated is not a binary property. Instead, we assign each object an outlier factor, which is the degree where the object is isolated. LOF shares some concepts with DB [2] such as the concepts of "reachability distance", which is used to estimate local density [1]. LOF (local outlier factor) of an o object is the average of the local accessibility ratio of o and the nearest k of o [1].

6. Evaluation

In this section, we are interested in answering the following question: (a) How accurate is our model in estimating the LSH parameters, when the LSH parameters are obtained using the fractal dimension of the datasets; (b) How the proposed LSH method behave in terms of *indexing and querying performance and accuracy*. Thus, the estimated LSH parameters are compared with the E2LSH algorithm proposed by Andoni [8]. And the performance evaluation by the proposed method. Moreover, for all the LSH-based methods we compared their query results with linear-scan search to measure the search accuracy. All the LSH indexing methods were implemented using the same algorithms for a fair comparison.

6.1. Results

An assessment of both quantitative and qualitative was conducted on the behavior of search algorithms for similarity and the discovery algorithms *motifs*, considering both

exact solutions as approximate. To answer queries by approximate similarity, exact and approximate methods were compared in two types of queries: range query and query the k-nearest neighbors. To identify *motifs*, they compared the methods focused on the discovery of *TopK-Motifs*. The performance evaluation were made in different configurations, involving real data sets, where the major difference was the size of objects and the set of data, the function of distance used, and data distribution. Furthermore, it is considered two ways of implementation: CPU and GPU.

Experiment 1: Performance of methods on kNN queries

The following are the results of performance, measured in terms of total time, precision, error rates (*error ratio*) [3] and use space to approximate kNN queries on sets of COLOR data and MNIST . In order to test the effectiveness of the methods in different search spaces were used functions away L_1 and L_2 for COLOR and MNIST sets respectively.

Table 1. Query time comparison, accuracy, *error ratio* and use of space index to approximate 100-NN queries. It presents the results for the sets of data COLOR (L_1) and MNIST (L_2).

COLOR (L_1)	time and accuracy	error ratio	mem.	
CUDA-LSH	0,005	0,92	0,98	25
LSH Multi-probe	0,004	0,60	0,79	351
LSH Multi-level	0,004	0,90	0,97	352
MNIST (L_2)	time and accuracy	error ratio	mem.	
CUDA-LSH	0,010	0,75	0,97	30
LSH Multi-probe	0,012	0,70	0,95	185
LSH Multi-level	0,014	0,72	0,96	235

The results show that the performance of CUDA-LSH is comparable to LSH methods but requires less memory space and least domain parameter setting, since in most cases only need to adjust the polling interval λ to improve the quality of results and Table 2 shows a comparison between two techniques used to get the nearest neighbor used in k-nn, brute force and LSH. We measure the response times of these two algorithms applied in knn to obtain the Local Outlier Factor.

Table 2. Time comparison in seconds to get Local Outlier Factor in synthetic datasets.

	RWALK	AGRODATA	AUDIO
Brute Force	180,471	153,758	242,793
LSH	152,511	100,917	210,274

Experiment 2: CUDA-LSH Queries Performance AllkNN

To evaluate the performance of the method AllkNN queries CUDA-LSH was implemented in different configurations: CPU (1, 2, 4 e 8 *threads*) and the GPU (*CUDA-AllkNN*). Since the focus of this work is the development of high-performance solutions for discovery algorithms *motifs* based on approximate solutions of similarity search, the scalability of the approximate technique is an important factor. Therefore, only scalability experiments are shown in this section. Thus, for the first scale experiment keeps the set data size, while the level of parallelism increases. Figure 1 shows the time required to index and query answer AllkNN in its various configurations. The entry consists of

SYNTH32 dataset consisting of 100 thousand elements, and the algorithm is configured to recover 25 nearest neighbors.

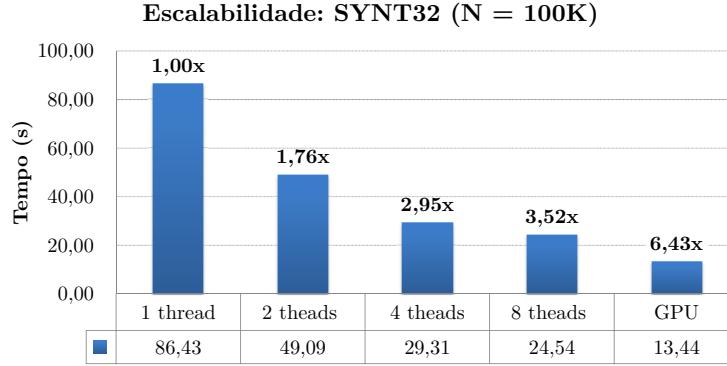


Figure 1. Comparison of total indexing time and AllkNN query to the CUDA-LSH method in its various configurations.

For a second experiment scalability was tested the implementation of AllkNN algorithm CPU with 8 *threads* (Par-AllkNN) and the implementation of AllkNN algorithm GPU (CUDA-AllkNN). The comparison of the results in terms of query times as the size of the dataset grows for CUDA-AllkNN algorithms, Par-AllkNN and AllkNN are summarized in Figure 2. For this experiment, it divides the data set into SYNT32 10,000 subsets elements. After entering each subset, performs A AllkNN query. To speed up the individual kNN queries AllkNN algorithms, both implemented in CPU and GPU are supported by CUDA-LSH method. These results demonstrate the potential of CUDA programming platform to accelerate similarity queries supported by algorithms and simple data structures to be designed in GPU. The results also show that although the total running time implementation CUDA increases with the data set size, the increase in time is much more linear than the algorithms implemented sequentially and in parallel in the CPU, which have large jumps between times of execution.

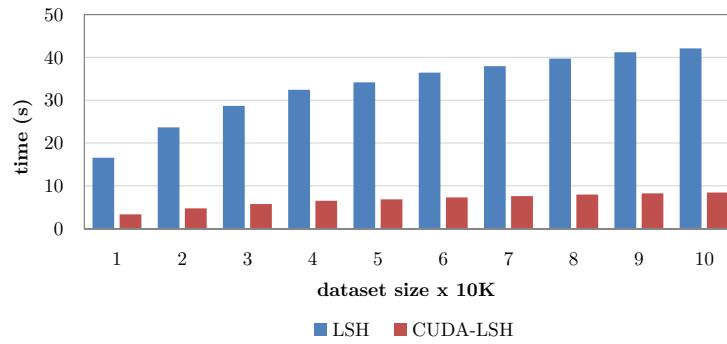


Figure 2. Comparison AllkNN queries as the data set size increases. The total time of indexing and querying for the set of SYNT32 data.

7. Conclusions

In this work we presented high performance algorithms for finding *motifs* and *outliers* using the search approach for approximate similarity and features of GPGPU programming paradigm. Thus, using efficient and parallel implementations CUDA-LSH

method was developed to provide high performance solutions for search and identification. To evaluate the results, several experiments were performed with real and synthetic data, and comparison between different implementations CPU and GPU. The experiments showed that the use of GPGPU computing ensures greater performance gain for large data sets. In this comparison, algorithms CUDA demonstrated the potential CUDA programming platform, with an acceleration of up to 7 times when comparing to the sequential version search and up to 15 times, when compared with the method of identifying motifs with *Random Projection* and Outliers with *Local Outlier Factor*. Overall, the results of this study showed that the appropriate operating approximate and programming *multi-thread* in complex issues of recovery and data mining techniques ensures significant performance gain, especially for algorithms that require large computing resources .

Acknowledgment

The authors thank CONCYTEC for the support and the financed Research, as well as the National University of San Agustín, La Salle University and CIENCIACTIVA for the assistance provided.

References

- [1] Markus Breunig, Hans-Peter Kriegel, Raymond T. Ng, and Jörg Sander. Lof: Identifying density-based local outliers. In *PROCEEDINGS OF THE 2000 ACM SIGMOD INTERNATIONAL CONFERENCE ON MANAGEMENT OF DATA*, pages 93–104. ACM, 2000.
- [2] Martin Ester, Hans-Peter Kriegel, Jörg Sander, and Xiaowei Xu. A density-based algorithm for discovering clusters a density-based algorithm for discovering clusters in large spatial databases with noise. In *Proceedings of the Second International Conference on Knowledge Discovery and Data Mining*, KDD’96, pages 226–231. AAAI Press, 1996.
- [3] Aristides Gionis, Piotr Indyk, and Rajeev Motwani. Similarity Search in High Dimensions via Hashing. In *Proceedings of the International Conference on Very Large Data Bases*, pages 518–529, San Francisco, CA, USA, 1999.
- [4] Douglas M Hawkins. *Identification of outliers*, volume 11. Springer, 1980.
- [5] Jessica Lin, Eamonn J. Keogh, Stefano Lonardi, and Pranav Patel. Finding Motifs in Time Series. In *Proceedings of the International Conference on Knowledge Discovery and Data Mining*, pages 53–68, Edmonton, Alberta, Canada, 2002.
- [6] Alexander Ocsa Mamani. Soluções aproximadas para algoritmos escaláveis de mineração de dados em domínios de dados complexos usando gpgpu, 2011.
- [7] Alexander Ocsa, Carlos Bedregal, and Ernesto Cuadros-Vargas. A new approach for similarity queries using neighborhood graphs. In *Proceedings of the Brazilian Symposium on Databases*, pages 131–142, João Pessoa, Paraíba, Brasil, 2007.
- [8] G Shakhnarovich, T Darrell, and P Indyk. Locality-sensitive hashing using stable distributions. 2004.

Póster Paper

Time Series Mining using Approximate Nearest Neighbor Search on GPUs

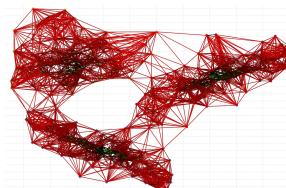
Abstract:

Motif discovery and outlier detection in time series databases is of prime importance and a major challenge in many areas, so exists a necessity of real-time algorithms. For this challenges, single CPU architectures impose limits on performance to deliver this kinds of solutions. Resorting to GPU programming is one approach to overcome these performance limitations. In this work, we discuss various time series motif discovery and outlier detection algorithms in order to select only those with highly parallelizable algorithm structure. More important, we present a comparative study among parallelizable exact and approximate algorithm to find motifs and outliers in very large time series databases. We use a NVIDIA CUDA framework to create a massively parallel implementation of the analyzed techniques. Experimental studies allow us to run experiments on large real and synthetic datasets thanks to the highly CUDA parallel implementation. Our performance evaluation on GeForce GTX 1080 GPU, result in average runtime speedups of 10 (15) using the MK (**Random Projection**) and **Local Outlier Factor** algorithms. The runtime speedups of CUDA is better than to those of Parallel MK and Parallel Random Projection running on 8 CPU cores of high-performance workstation.

Knn & ALL-Knn

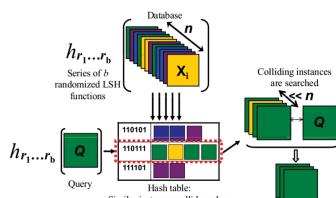


Figure 1. The clusters candidates to motif using the nearest neighbors



All-Knn Representation

Locality Sensitive Hashing



Proposal

the estimated LSH-CUDA parameters are compared with the E2LSH algorithm. And the performance evaluation by the proposed method. to find motifs in time series, and we presented a comparison between LSH and brute force to discover outliers.

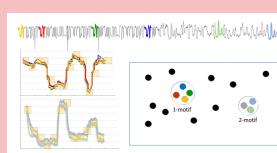


Image. Motifs Representation

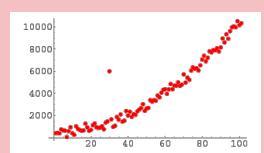


Image. Outliers Representation

Results

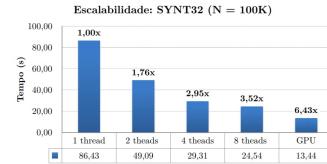


Figure 2. Comparison of total indexing time and AllkNN query to the CUDA-LSH method in its various configurations.

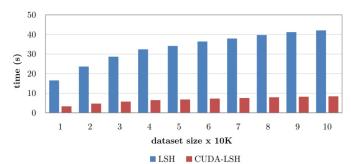


Figure 3. Comparison AllkNN queries as the data set size increases. The total time of indexing and querying for the set of SYNT32 data.

Conclusions

In this work we presented high performance algorithms for finding motifs and outliers using the search approach for approximate similarity and features of GPGPU programming paradigm. Thus, using efficient and parallel implementations CUDA-LSH method was developed to provide high performance solutions for search and identification. To evaluate the results, several experiments were performed with real and synthetic data, and comparison between different implementations CPU and GPU. The experiments showed that the use of GPGPU computing ensures greater performance gain for large data sets. In this comparison, algorithms CUDA demonstrated the potential CUDA programming platform, with an acceleration of up to 7 times when comparing to the sequential version search and up to 15 times, when compared with the method of identifying motifs with Random Projection and Outliers with Local Outlier Factor. Overall, the results of this study showed that the appropriate operating approximate and programming multi-thread in complex issues of recovery and data mining techniques ensures significant performance gain, especially for algorithms that require large computing resources.

Repositorio de la Amazonia Peruana

Series Temporales de la Amazonía Peruana, Bases de datos & Algoritmos

Descarga, prueba y compara tus resultados para colaborar con la comunidad.

VER MÁS



UCAYALI (2012 - 2016)

Show 10 entries Search:

	TempOut	HiTemp	LowTemp	OutHum	DewPt	WindSpeed
	TempOut	HiTemp	LowTemp	OutHum	DewPt	WindSpeed
2012-01-31	26.292816091954034	26.50531609195402	26.082040229885056	85.5	23.4625718390805	2.24813457
2012-02-29	25.482819548872207	25.678082706766975	25.291090225563885	87.80676691729323	23.140187969924796	2.27641301
2012-03-31	25.956131650135273	26.186248872858464	25.73990081154196	86.9107303877367	23.414968440035995	2.14831317
2012-04-30	26.14422032583399	26.3647013188518	25.920713731574896	87.93095422808379	23.807757951900697	1.92885289
2012-05-31	26.09388888888912	26.312708333333372	25.876249999999995	88.61875	23.902430555555608	1.80758679
2012-06-30	25.673353293413196	25.890818363273482	25.44231536926149	88.2624750499002	23.406886227544884	1.83423741
2012-07-31	24.963382250174696	25.2116701607267	24.720684835779153	85.7267645003494	22.13885394828792	2.10117873
2012-08-31	25.652638888888887	25.881805555555548	25.427291666666667	84.61875	22.648055555555565	2.34498774
2012-09-30	25.608333333333363	25.813074712643644	25.401580459770116	85.23994252873563	22.72579022988507	2.22983572

Showing 1 to 10 of 61 entries Previous 1 2 3 4 5 6 7 Next

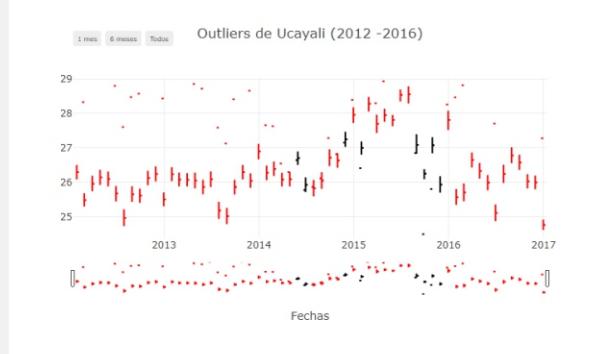
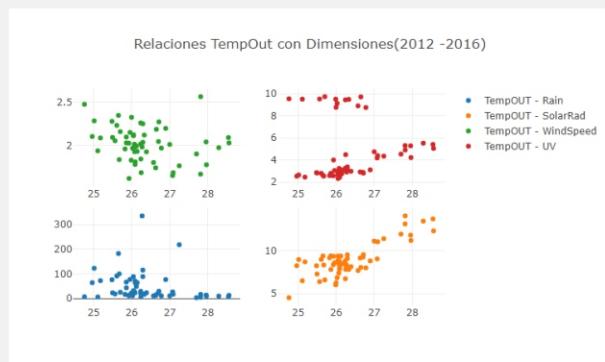
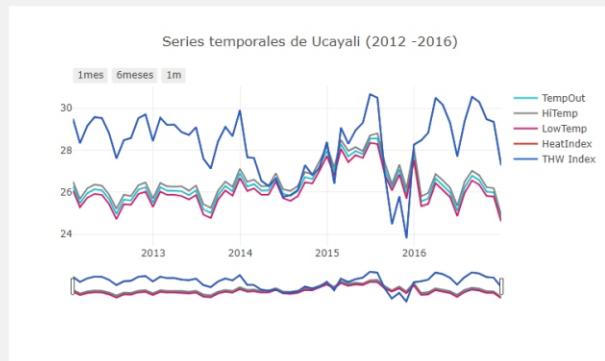
Base de datos climatológica de la provincia del Ucayali proporcionada por el [Instituto de Investigaciones de la Amazonía Peruana](#), que comprende mediciones con frecuencia de datos de cada media hora, desde el año 2012 al 2016, y cuenta con datos números.

Dado a la complejidad de recolección de información a través de sensores, esta base de datos fue interpolada para completar las mediciones faltantes, para esto se utilizaron diversas herramientas que fueron: Pandas una librería de Python, los datos faltantes se reemplazó por NaN, y para completarlos se utilizó una interpolación de orden lineal, y no de orden 2 o 3 porque deformaban la data, las columnas de 'WindDir' y 'HiDir' que son strings y no se podían interpolar ya que son puntos cardinales

En esta tabla solo se muestra una muestra mensual, dado al tamaño de la tabla, cada dimensión comprende 60,000 datos y se cuenta con 36 dimensiones puede descargarla desde nuestro repositorio.

DESCARGAR

REPRESENTACIONES GRAFICAS



ALGORITMOS

Los algoritmos que se utilizaron para analizar la información comprendió, en dos tipos, uno para analizar los patrones frecuentes que incidían en la información más conocidos como Motifs, estos nos muestran incidencias repetitivas en ratios controlados como semanas, meses o años.

Para la detección de anomalías o ruido se empleó local outlier factor un algoritmo muy preciso que se encarga en encontrar los outliers más distantes de las series de tiempo.



DESCARGAR

ALGORITMOS

Los algoritmos que se utilizaron para analizar la información comprenden, en dos tipos, uno para analizar los patrones frecuentes que inciden en la información más conocidos como Motifs, estos nos muestran incidencias repetitivas en ratios controlados como semanas, meses o años.

Para la detección de anomalías o ruido se empleó el factor outlier local, un algoritmo muy preciso que se encarga de encontrar los outliers más distantes de las series de tiempo.



DESCARGAR

Descarga las bases de datos y los algoritmos

Descarga Ya

Dale click y descarga las herramientas usadas que se encuentran almacenadas en Github

BD

ALGORITMOS

Publicaciones y colaboradores



Administración

Islas Pitcairn

Google

Contáctanos

Nombre

Email

Título

Mensaje

ENVIAR

