**ESCUELA TÉCNICA SUPERIOR DE INGENIEROS INDUSTRIALES Y DE TELECOMUNICACIÓN**

**UNIVERSIDAD DE CANTABRIA**

*Trabajo Fin de Máster*

# Uso de Machine-Learning en el control de congestión sobre redes 5G

# On the use of Machine-Learning for congestion control over 5G networks

**Para acceder al Titulo de**

***Máster Universitario en Ingeniería de Telecomunicación***

**Autor: Alfonso Fernández Gutiérrez**

**Julio - 2020**

E.T.S. DE INGENIEROS INDUSTRIALES Y DE TELECOMUNICACION

# MASTER UNIVERSITARIO EN INGENIERÍA DE TELECOMUNICACIÓN

CALIFICACIÓN DEL TRABAJO FIN DE MASTER

**Realizado por: Alfonso Fernández Gutiérrez**
**Director del TFM: Luis Francisco Diez Fernández**
**Ramón Agüero Calvo**
**Título:** " Uso de Machine-Learning en el control de congestión sobre redes 5G"
**Title: " On the use of Machine-Learning for congestion control over 5G networks "**

**Presentado a examen el día: 21/07/2020**

para acceder al Título de

# MASTER UNIVERSITARIO EN INGENIERÍA DE TELECOMUNICACIÓN

Composición del Tribunal:
Presidente (Apellidos, Nombre): Granda Miguel, María Mercedes
Secretario (Apellidos, Nombre): García Arranz, Marta
Vocal (Apellidos, Nombre): Irastorza Teja, José Ángel

Este Tribunal ha resuelto otorgar la calificación de: .....

Fdo.: El Presidente

Fdo.: El Secretario

Fdo.: El Vocal

Fdo.: El Director del TFM
(sólo si es distinto del Secretario)

Vº Bº del Subdirector

Trabajo Fin de Máster Nº
(a asignar por Secretaría)

# Índice

# Índice de figuras

# Resumen ejecutivo

El número de usuarios conectados a las redes móviles está aumentando rápidamente, a la vez que los requisitos de los usuarios. Las redes 5G surgen como una solución a este problema de demanda de tráfico creciente.

La próxima generación de comunicaciones móviles incluye el uso de redes de alta frecuencia, por encima de 24 GHz. El canal físico en estas longitudes de onda es altamente variante, además de existir elevadas pérdidas de transmisión.

Los mecanismos tradicionales de control de congestión no funcionan de manera adecuada en este entorno, por lo que el uso de técnicas más novedosas en este campo como las pertenecientes al campo del *Machine Learning* pueden cobrar mayor relevancia.

En este documento se realiza un estudio de los datos disponibles por parte de los nodos finales de la red, así como su utilidad en la predicción de la congestión, con el objetivo final de poder realizar un control de la congestión más eficiente en estos entornos.

Para realizar esta investigación se han llevado a cabo diferentes simulaciones sobre entornos de ondas milimétricas, utilizando el software de simulación ns-3. Los datos obtenidos en estas simulaciones han servido para elaborar diferentes soluciones que predicen el nivel de congestión.

# Abstract

The number of devices connected to mobile networks is heavily increasing day by day, as well as user's requirements. 5G networks emerge as a feasible solution to increasing traffic demand situation.

Next generation networks include the use of high-frequency technologies, above 24 GHz. Physical propagation at these wavelengths is highly variable and presents high transmission losses.

Traditional congestion control mechanisms do not work properly on these environments. The use of novel techniques in this field, such as *IA* solutions, seems to be an approach still undiscovered.

This report studies the available performance indicators at the network end nodes, as well as their benefits in predicting congestion. The goal is being able to perform more efficient congestion control mechanism on these environments.

In order to carry out this research, different simulations on mmW environments have been developed using the well-known ns-3 simulation platform. Collected data has been afterwards used to develop different solutions that are able to predict the level of congestion.

# Agradecimientos

En primer lugar, quiero agradecer a Luis Francisco y a Ramón la ayuda que me han prestado ya que sin ellos no hubiera sido capaz de sacar adelante este trabajo, siempre que he necesitado consejo han estado dispuestos a ayudarme.

También debo agradecer a mi familia especialmente a mis padres todo el apoyo que me han mostrado, no solo para la elaboración de este trabajo sino a lo largo de todo el Máster.

Por último, dar gracias a mis amigos y compañeros de clase con los cuales he podido alejarme de las preocupaciones de los estudios siempre que lo he necesitado.

# 1 Introducción

En la primera parte de este documento se plantea el problema que motiva la realización de este trabajo, así como los resultados que se pretenden obtener. Por último, se describe la estructura del documento y de las diferentes partes que lo componen.

## 1.1. Planteamiento del problema

Desde hace unos años venimos experimentando una gran revolución en lo que se refiere al ámbito de las telecomunicaciones, especialmente en el entorno inalámbrico. La nueva generación de redes móviles (5G) promete unas prestaciones muy superiores a las de los sistemas actuales.

Para tratar de lograr estos objetivos en las redes de 5ª generación se van a incluir nuevas bandas de frecuencia con objeto de conseguir incrementar el ancho de banda. Concretamente se van a usar bandas de ondas milimétricas (a partir de ahora mmW), por encima de 24 GHz [3].

El uso de mmW presenta una serie de problemas a nivel físico [4], entre los que se pueden destacar algunos como: alto nivel de absorción por parte de las moléculas de oxígeno, altas pérdidas por penetración, difracción... Todo esto provoca que el estado del canal sea altamente cambiante, lo que también afecta a su capacidad.

El protocolo tradicional más extendido para asegurar la recepción de la información extremo a extremo es Transport Control Protocol (TCP), que fue diseñado para entornos cableados. El uso de TCP sobre redes inalámbricas provoca una gran caída en términos de eficiencia, debido al control de la congestión, ya que no es capaz de distinguir entre pérdidas debido a la congestión o provocadas por el estado físico del canal. Como resultado, el rendimiento de TCP sobre aquellas redes inalámbricas se ve fuertemente perjudicado.

En este trabajo se propone utilizar técnicas de *machine learning* para tratar de predecir el nivel de congestión de la red a través de las diferentes métricas disponibles a nivel de transporte.

## 1.2. Objetivos

Los principales objetivos que se quieren alcanzar con este trabajo son:

- Estudiar diferentes técnicas de aprendizaje automático, para predecir la congestión.
- Servir como punta de lanza para demostrar el potencial de estas técnicas en la predicción de la congestión.
- Obtener resultados de aprendizaje no supervisado, ideal para implementar en dispositivos finales, y compararlos con los resultados obtenidos a través de otras técnicas más avanzadas de aprendizaje supervisado.

El uso de técnicas de aprendizaje no supervisado es ideal para el uso *on the fly*, ya que, en primer lugar, no suelen requerir tanta memoria ni tiempo de procesado como pueden necesitar los algoritmos de aprendizaje supervisado. Además, no necesitan tener datos etiquetados para su funcionamiento, siendo esta característica muy interesante de cara a una solución pensada para funcionar de manera autónoma.

Para la obtención de estos datos se ha utilizado el simulador ns-3, del cual se hablará más adelante. Por otra parte, el análisis de los datos, así como el propio uso de los algoritmos se ha desarrollado a través de Python, principalmente, y MATLAB.

## 1.3. Estructura de la memoria

En esta sección se presenta la estructura del documento, explicando brevemente los contenidos de cada uno de los capítulos que lo conforman.

- Capítulo 2: Se describen los desafíos que presentan las redes de nueva generación. Se explica de forma detallada los conceptos necesarios para su desarrollo. Se estudian las diferentes variantes del protocolo TCP utilizadas hoy en día para la transmisión sobre redes inalámbricas. Además, se menciona algunas de las soluciones que tratan de cubrir las carencias que presenta este protocolo, las cuales no ofrecen un rendimiento adecuado en este tipo de canales mmW.

- Capítulo 3: Se explica el escenario que se ha utilizado para la obtención de las métricas. Se introducen además algunos aspectos a tener en cuenta respecto la tecnología 5G.

- Capítulo 4: En este apartado se realiza un análisis de los datos extraídos del simulador, con el objetivo de discernir aquellos más valiosos en lo que a la predicción de la congestión se refiere.

  También se detalla brevemente los diferentes algoritmos utilizados, describiendo sus principios y sus casos de uso habituales.

- Capítulo 5: Se presentan los resultados obtenidos de las diferentes pruebas realizadas, y se realiza una comparativa entre las mismas.

- Capítulo 6: Finalmente se concluye el trabajo, resumiendo los resultados obtenidos y extrayendo las principales conclusiones obtenidas. Además, se plantean futuras líneas de uso/investigación, basadas en estos resultados.

# 2

# Estado del Arte

Este capítulo recoge una introducción a cada uno de los elementos en los que se sustenta el trabajo, así como una explicación de la problemática actual. De esta forma el lector puede adquirir unos conocimientos básicos que le permitan comprender en mayor medida los diferentes conceptos tratados a lo largo del documento.

## 2.1. Comunicaciones en transmisiones celulares (5G)

Las redes 5G o redes New Generation (NG) suponen un cambio drástico en lo que a prestaciones se refiere, respecto a tecnologías anteriores. Prometen mejor rendimiento en ancho de banda, latencia y fiabilidad. Además, a esto hay que sumarle que el número de dispositivos conectados que se espera durante los próximos años es muy superior al actual, debido principalmente al Internet of Things (IoT). Este hecho condiciona también el desarrollo de esta nueva tecnología, así como las soluciones utilizadas.

En la Tabla 2.1 se puede ver una comparativa de las diferentes generaciones celulares que se han utilizado hasta la fecha, así como las características de a las redes 5G.

Dentro de la especificación 5G se describen diferentes tipos de servicios, cada uno de ellos con unos objetivos muy diferenciados, lo que desembocará en diferentes tipos de conexiones dentro de la misma red.

Tabla 2.1: Comparativa de las diferentes tecnologías celulares

| | 1G | 2G | 3G | 4G | 5G |
|---|---|---|---|---|---|
| Fecha | 1970-1980 | 1990-2004 | 2004-2010 | Actualmente | 2020-2025 |
| Tecnología | Analógica | GSM | WCDMA | LTE | NR |
| Freq. | 30 KHz | 1.8 GHz | 1.6-2 GHz | 2-8 GHz | 3-300 GHz |
| Tasa binaria | 2 kbps | 64 kbps | 144 kbps - 2 Mbps | 100-1000 Mbps | $>$ 1 Gbps |
| Multiplexación | FDMA | TDMA,CDMA | CDMA | OFDMA | OFDMA |
| Intercambio de: | Circuitos | Circuitos | Paquetes | Todo paquetes | Todo paquetes |

Los tipos de conexiones que se distinguen son 3 (Figura 2.1):

- enhanced Mobile Broadband (eMMB): Este tipo de conexiones están orientadas a obtener el mayor ancho de banda posible, se contemplan servicios como Video on Demand (VoD) de alta calidad, realidad virtual,...
- Ultra Reliable Low Latency Communications (URLLC): Conexiones de latencia mínima con alta fiabilidad, pensado para control de maquinaría en remoto, conexiones entre vehículos, etc.
- massive Machine Type Communications (mMTC): Se espera la conexión masiva por parte de dispositivos de bajas prestaciones, con una duración limitada de la batería.

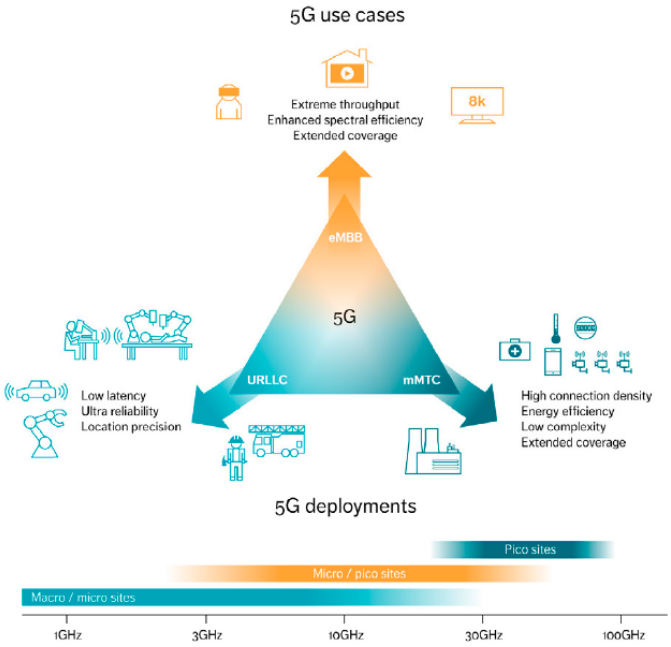


Figura 2.1: Diferentes tipos de servicios que abarca 5G [1]

Para poder lograr estos objetivos de rendimiento se pretende hacer uso de diferentes nuevas técnicas, así como aumentar el ancho de banda utilizado.

La tecnología 5G va a tener 3 bandas principales de despliegue, en España las bandas destinadas a este servicio son: 700 MHz, 3.6 GHz y 26 GHz. La primera actualmente se encuentra parcialmente ocupada por Television Digital Terrestre (TDT) mientras que la segunda y la tercera se encuentran disponibles. Este trabajo está orientado a esta tercera banda de frecuencia, por no haberse utilizado hasta ahora, además de los desafíos añadidos que implican frecuencias tan elevadas.

Se van a describir brevemente algunas características de las capas que conforman una comunicación celular en 5G, también denominado 5G New Radio (NR):

- Capa física / MAC [1, 5]:
  - Tamaño de las células: Con objeto de aumentar la capacidad de envío de información, así como con previsión a un crecimiento sin precedentes en el número de dispositivos conectados (debido principalmente al IoT) se plantea reducir el tamaño de las células para dar servicio a áreas más pequeñas. Así, las macro células operaran mayoritariamente en frecuencias más bajas, ya que son las que sufren menores atenuaciones debido al canal, mientras que las pico celdas en mmW.
  - Formas de onda: Mientras que en LTE se está usando OFDM, en 5G se va a usar Cyclic Prefix OFDM (CP-OFDM), tanto en el Up link (UL) como en el Down link (DL), para facilitar su uso, especialmente orientado a comunicaciones Device to device (D2D), en los que los dispositivos tienen prestaciones limitadas.
  - MIMO masivo multiusuario(MU-MIMO): Uso de múltiples antenas en recepción y en transmisión (arrays de 64-256 antenas). Se presentan diferentes soluciones en función de la banda de trabajo, ya que las características de las antenas son altamente dependientes de la frecuencia.
  - Uso de turbo-códigos como mecanismo de codificación de canal, para tener una comunicación robusta.
- Radio Link Control (RLC)[6]: Se encarga de numerosas tareas que garantizan la entrega de los paquetes. Presenta diferentes modos de operación: Unacknowledged mode (UM), Transparent mode (TM) y Acknowledged mode (AM). De ellos el más utilizado es AM, por garantizar la entrega de los datos de manera ordenada. Las características descritas a continuación hacen referencia a este modo:
  - Concatenación, segmentación y reensamblado de las Service Data Unit (SDU)s de la capa superior.
  - Detección de duplicados y reordenación de los paquetes.
  - Corrección de errores a través de Hibrid Automatic Repeat Request (HARQ).RLC solicita retransmisión cuando no sea capaz de corregir el paquete por sí mismo, en caso de que se exceda el número máximo de retransmisiones establecido, la conexión se cierra.

- Packet Data Convergence Protocol (PDCP) [7]: Esta capa se encarga de coordinar el plano de datos con el plano de control en la red celular, así como de la compresión de cabeceras y el cifrado de la información.

## 2.2. Comparativa TCP e implementaciones actuales

TCP es el protocolo de transporte por excelencia en el tráfico a través de Internet. Desde 1988, cuando apareció la primera especificación de este protocolo en el RFC 793, hasta hoy han aparecido numerosas versiones que tratan de solventar y mejorar algunos aspectos problemáticos del protocolo. Algunas de las más populares son: (Véase Figura 2.2)

- Tahoe: Fue una de las primeras soluciones propuestas. Presenta 2 fases diferenciadas, la primera de ellas *Slow Start*, donde la ventana de congestión comienza con tamaño igual a 1 Maximum Segment Size (MSS) y se va incrementando de forma exponencial hasta llegar a un umbral establecido. A partir de ese momento entra en la fase de *Congestion Avoidance*, donde el crecimiento de la ventana es lineal. Se detecta congestión cuando expira el tiempo máximo de espera Retransmission TimeOut (RTO) o cuando se recibe 3 veces la confirmación del mismo segmento. Tras estos casos se modifica el umbral para entrar en la 2ª fase y se reinicia la ventana en *Slow Start*.

- Reno: El problema que presenta la versión anterior es que la política de la ventana de congestión es muy "conservadora" reduciendo la ventana a tamaño 1 tras un triple ACK. Reno implementa *Fast Recovery*, que reduce la ventana a la mitad cuando se recibe un triple ACK.

- Vegas: A diferencia de las dos versiones presentadas anteriormente, Vegas basa su interpretación de la congestión en el retardo, en lugar de en la pérdida de paquetes. Trata de detectar la congestión cuando empieza a producirse y para detectarlo se basa en la variación del Round Trip Time (RTT).

- Westwood: Esta implementación está pensada para ser implementada en redes inalámbricas. Usa como métrica el Bandwidth Delay Product (BDP) para estimar el ancho de banda disponible. Tiene una mayor rapidez de recuperación del tamaño de la ventana tras momentos de congestión.

- Cubic: Esta variante se diseñó orientada a redes con una alta tasa de transmisión. En este caso la evolución de la ventana sigue la siguiente expresión $W = C(\Delta - \sqrt[3]{\beta \frac{W_{max}}{C}})^3 + W_{max}$. La forma que sigue esta función es cúbica, dándose un crecimiento elevado al principio, disminuyendo en la zona cercana a $W_{max}$ y volviendo a crecer después. Actualmente está implementación es la que se usa por defecto en el Kernel de Linux desde la versión 2.6.19 y en Windows 10 a partir de la versión 10.1709.

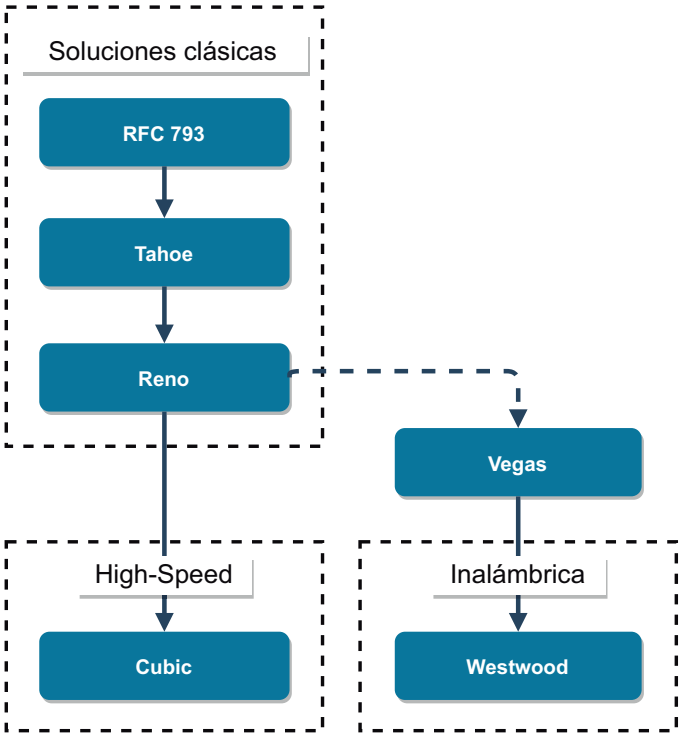


Figura 2.2: Diagrama de evolución de las versiones de TCP

## 2.3. Problemática en el entorno inalámbrico y soluciones previas

Como ya se ha comentado anteriormente, la transmisión de información a través de la interfaz radio presenta diferentes problemáticas, las cuales se acentúan cada vez más, a medida que se sube en frecuencia.

TCP ha demostrado no ser un protocolo eficiente en este tipo de entornos. En la mayoría de las implementaciones actuales se utiliza *TCP Cubic* como protocolo de capa de transporte, que trata las pérdidas de paquetes como congestión. Así, en un canal con capacidad y prestaciones cambiantes, como es el entorno inalámbrico, esta implementación desemboca en una infrautilización de los recursos.

Existen soluciones que ofrecen un rendimiento superior que Cubic, que se basan en control de la congestión en base al retardo como puede ser los algoritmos Verus [8] o Sprout [9].

En 2016 Google publicó el algoritmo BBR [10], que usa como métrica el BDP. Actualmente, TCP trata de ofrecer el mayor ancho de banda posible, lo que provoca un aumento del retardo en las transmisiones. Este hecho se acentúa a medida que se aumenta el número

de usuarios. Por este motivo, resulta más eficiente tratar de buscar un compromiso entre ambas métricas.

Todas estas soluciones mejoran el control de la congestión en redes inalámbricas tradicionales. Sin embargo, no ofrecen buenos resultados en un escenario de mmW.

## 2.4. ECN y solución propuesta

Por otra parte, desde el año 2001 se propone introducir una modificación en los protocolos TCP/IP para notificar a los diferentes nodos de la red de situaciones de congestión [11]. Esta cabecera introduce el campo Explicit Congestion Notification (ECN), que puede ser modificado por los diferentes nodos de la red. De esta manera se puede detectar una situación de congestión en la red sin tener que esperar a que se produzca una pérdida de paquetes o un *timeout*. Para transmitir esta información de extremo a extremo se utilizan 2 bits, que codifican los siguientes estados (Tabla 2.2).

Tabla 2.2: Codificación ECN

| Código | Descripción |
|---|---|
| 00 | ECN No disponible |
| 01 | ECN Disponible , ECT(0) |
| 10 | ECN Disponible , ECT(1) |
| 11 | Congestión detectada , CE |

Los códigos ("01" y "10") se marcan indistintamente cuando ambos dispositivos extremos soportan el uso de ECN. Si los nodos intermedios tienen implementada alguna solución basada en Random Early Detection (RED), los nodos empezarán a descartar paquetes con probabilidad $p$ antes de llegar al estado de congestión. Además, modificarán el valor del campo ECN de la cabecera a "11" de los paquetes que se encolen con probabilidad ($1$-$p$), véase Figura 2.3.

Esta información en la cabecera acabará llegando al nodo receptor, que podrá adaptar su tasa de transmisión para controlar la congestión.

Esta idea de notificación explícita por parte de la red ya ha sido explotada por algoritmos previos de capa de transporte, como Data Centers TCP (DCTCP) desarrollado por Microsoft [12, 13], que ofrece un rendimiento muy superior en escenarios donde existe un alto requisito de latencia, y de ancho de banda.

El problema que presenta el uso de ECN es que la mayoría de los nodos intermedios de la red no implementan ninguna solución RED [14], por lo que la implementación por parte de los dispositivos finales de un algoritmo basado en esta información no es eficaz.

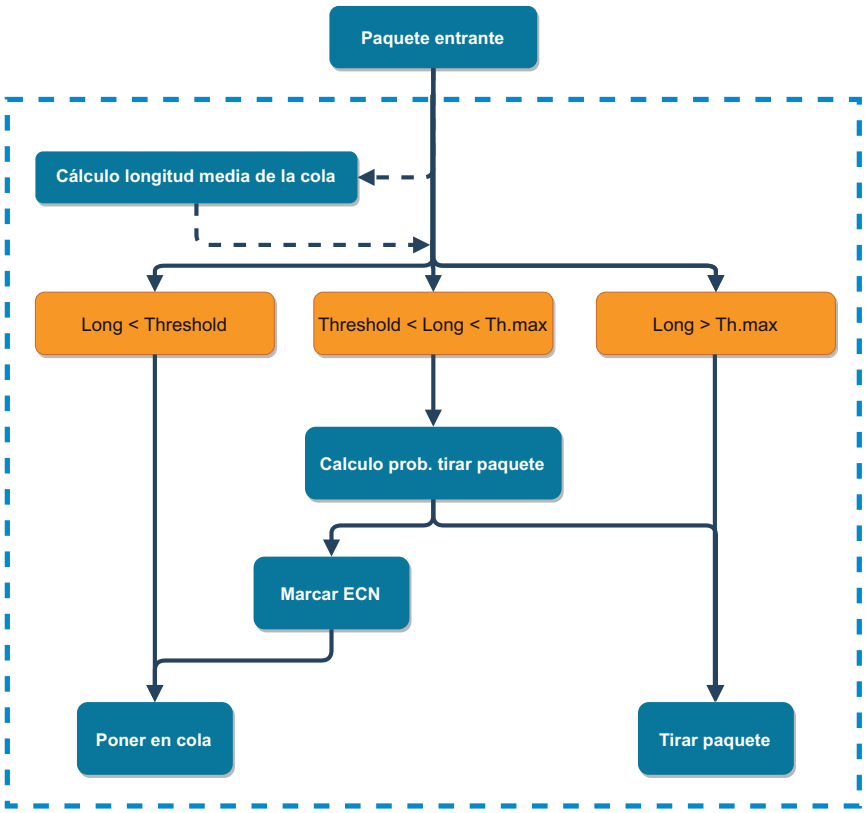


Figura 2.3: Esquema funcionamiento RED

En su lugar la propuesta presentada en este trabajo, inspirada en esta idea, trata de predecir el estado de la congestión a partir de algunas de las métricas disponibles a nivel de transporte. Estas métricas son el *delay* entre paquetes y el Inter Arrival Time (IaT).

Toda la obtención de datos se ha realizado en la plataforma de simulación ns-3, de la cual se va a hablar en el próximo capítulo.

# 3

# Obtención de trazas

Para analizar una solución que pueda ser aplicada en un entorno real es necesario tener en cuenta todos los elementos que forman parte de la comunicación. Este capítulo describe los escenarios generados, así como los modelos usados con el simulador ns-3.

## 3.1. Simulador ns-3

Ns-3 es un conocido simulador por eventos discretos que ya se encuentra en su tercera versión. Es una herramienta diseñada en C++, aunque actualmente presenta frameworks para trabajar también sobre Python.


Figura 3.1: Logo simulador ns-3

Es una herramienta de software libre, distribuida bajo la licencia GNU.

Se estructura en gran cantidad de módulos, los cuales pueden combinarse para crear la arquitectura completa de una red, con un alto nivel de detalle y control sobre cada una de sus partes.

Contiene módulos para redes IP y para redes basadas en otras tecnologías. En lo que

respecta a la capa física incorpora modelos que simulan entornos cableados, así como Wifi, LTE y mmW. También permite configurar protocolos de capa de control de la red, como pueden ser los *schedulers* encargados de distribuir los recursos en las redes.

Esta herramienta cuenta con una gran comunidad investigadora que contribuye a la elaboración y mantenimiento de nuevos módulos. Como dato para mostrar el tamaño de la comunidad, en *Google Scholar* aparecen más de 2000 artículos desde 2017 basados en el uso de esta herramienta.

## 3.2. Escenario utilizado

En otros trabajos previos del grupo se ha realizado un análisis similar al realizado en este trabajo utilizando el emulador de canales de mmW MahiMahi [15].

Sin embargo, utilizar ns-3, implementando la pila de protocolos mencionada anteriormente (Sec. 2.1), parece una solución más exacta. Permite realizar un análisis más exhaustivo de la contribución de las diferentes entidades que componen la comunicación.

Dado que este trabajo está centrado en analizar la interfaz radio en comunicaciones de 5G, la configuración en el *backbone* de la red ha sido simplificada. La gestión de la congestión y de otros parámetros en esta parte de la red requeriría otro estudio, complementario al realizado en la parte celular.

La arquitectura del escenario diseñado puede verse en la Figura 3.2, donde se asume que la capacidad de transmisión de la red de transporte (desde la red celular hasta el destino) es infinita, por lo que se modela como una conexión punto a punto con una capacidad que no llegue a saturarse, 100 *Gbps*.

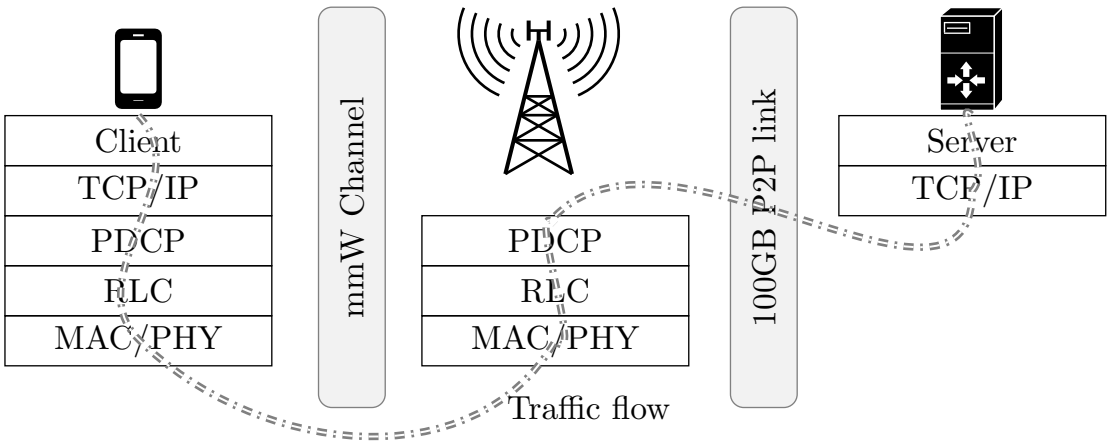


Figura 3.2: Pila de protocolos usada en la simulación

En lo que respecta al canal físico, es necesario modelar correctamente su comportamiento en el rango de frecuencias de mmW, donde se produce un gran número de desvanecimientos, absorciones elevadas... Para ello se hace uso de la especificación del 3GPP [16] referente a este tipo de canales, utilizando un módulo que sea capaz de modelar adecuadamente estos escenarios.

Estos modelos suelen estar elaborados a través de métodos empíricos y realizan una representación del canal basada en procesos estocásticos, donde se tienen en cuenta factores como la tasa de transmisión, la distancia entre emisor-receptor, el tipo de escenario...

En este caso, se aplica el modelo *urban macro channel*, pensado para situaciones en entornos urbanos donde se pueden producir multitud de desvanecimientos, trayectorias multicamino provocadas por reflexiones, etc.

Uno de los factores más determinantes a la hora de obtener el valor de la Signal Interference Noise Ratio (SINR) en un instante dado es la estimación que hace el modelo sobre si se encuentra en una situación con Line of Sight (LOS) o sin ella. Se utiliza para ello una distribución de probabilidad dependiente con la distancia.

Se considera un escenario sencillo, con un único usuario. Este permanece estático a una distancia fija de la estación base, que se va modificando en cada uno de los escenarios generados.

Tanto el receptor como el transmisor están equipados con una única antena. Además, se asume que ambas antenas están perfectamente orientadas.

Para un escenario más realista es necesario incluir en la simulación el uso de diversidad espacial en las transmisiones, a través de técnicas Multiple Input Multiple Output (MIMO).

La matriz que caracteriza el canal físico se actualiza cada $ms$, cambiando las condiciones para el resto de protocolos.

Por último, para poder caracterizar de manera correcta la congestión se hace uso de User Datagram Protocol (UDP), que no incorpora ningún tipo de control de congestión, ya que si usásemos TCP los mecanismos de control que incorpora afectarían los resultados obtenidos.

Se han generado diferentes escenarios con tasa de envío constante en cada uno de ellos.

## 3.3. Parámetros de rendimiento

Debido a la versatilidad del simulador se pueden obtener numerosos parámetros de rendimiento de las simulaciones. La información extraída del simulador se ha divido por capas.

- Capa física: La información de esta capa no puede ser utilizada a priori por los protocolos de transporte, a no ser que implementen algún tipo de solución *cross-layer*.

  Sin embargo, se recoge la cantidad de datos transmitidos en cada oportunidad de transmisión, ya que describe en parte el comportamiento del sistema en función del planificador (*scheduler*) que se utilice.

  Se han utilizado 2 schedulers diferentes en este trabajo: El **greedy** scheduler y el scheduler **max-weight**.

- Capa de enlace y RLC: Una mayor congestión en el enlace se podría apreciar como un aumento de la ocupación en este nivel, aunque existen diferentes buffers de almacenamiento, que sirven para implementar el HARQ. El principal almacenamiento se realiza en el protocolo RLC.

  La ocupación del buffer correspondiente se calcula como la diferencia entre los bytes recibidos y aquellos que han sido enviados y confirmados. De esta forma se consideran no solo los bytes pendientes de transmisión sino, también aquellos pendientes de posibles retransmisiones.

- Capa de transporte: El hecho de que se produzca una pérdida a nivel 4 es bastante improbable, debido principalmente a las técnicas utilizadas por las capas inferiores, como son la codificación de canal con el uso de turbo-códigos, así como las retransmisiones de RLC.

  La congestión se observa por tanto como retardo en la llegada de los paquetes, lo que puede provocar RTOs. Las métricas capturadas son: el retardo entre paquetes y el tiempo entre llegadas (IaT).

  Estos parámetros, a pesar de no ser visibles explícitamente por los mecanismos de la capa de transporte, pueden ser fácilmente estimados a partir del RTT y el tiempo entre ACKs consecutivos.

Como se ha comentado anteriormente, se han utilizado dos *schedulers* diferentes. Tanto el *scheduler greedy* como *max-weight* [17] son ampliamente utilizados en comunicaciones celulares. Para el reparto de los recursos radio hacen uso de métricas como el estado del buffer, la modulación seleccionada, así como el Modulation and Coding Scheme (MCS).

En las Figuras 3.3 y 3.4 se puede observar el comportamiento de ambas configuraciones en un fragmento de una transmisión con una duración de 5 segundos.

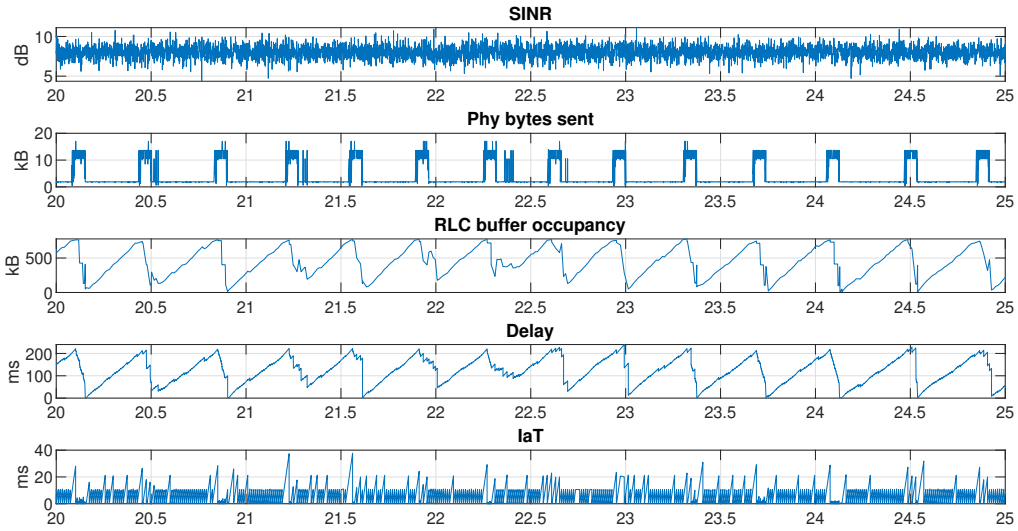


Figura 3.3: Transmisión para una distancia de 30 m y una tasa de envío de 30 Mbps (*Greedy scheduler*)

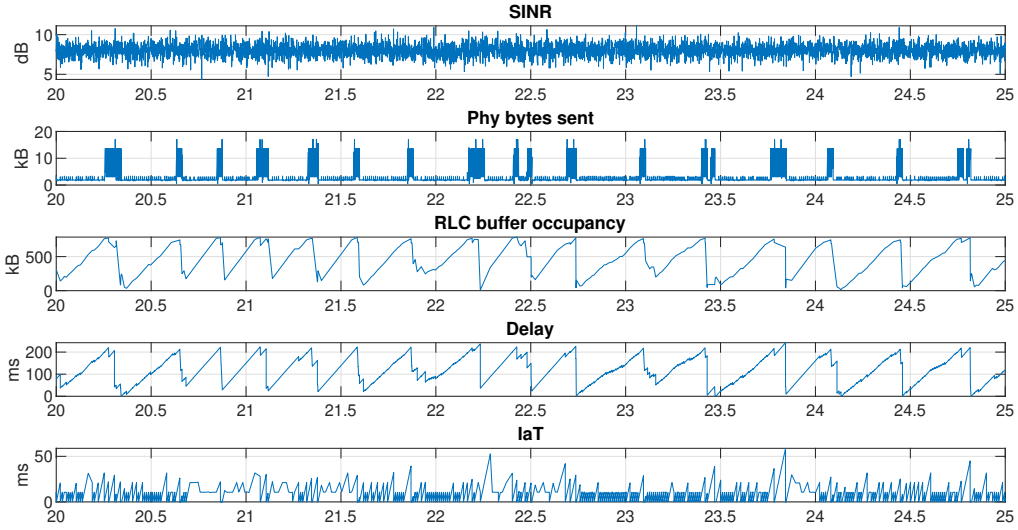


Figura 3.4: Transmisión para una distancia de 30 m y una tasa de envío de 30 Mbps (*Max-weight scheduler*)

Se observa como el tráfico enviado a través del canal se envía en forma de ráfagas. Estos periodos de transmisión deberían coincidir con periodos de buenas condiciones de canal.

Se puede apreciar como los picos en la ocupación del buffer RLC coinciden con periodos donde no se producen transmisiones en el canal físico y posteriormente se vacía parte del buffer.

Por otra parte, se puede ver como la ocupación tiene un carácter periódico, pasando de un estado de ocupación mínima a otro de ocupación máximo, que se repite a lo largo del tiempo.

En el caso del segundo *scheduler* también se observa un comportamiento periódico y a ráfagas, pero algo más errático que en el primer caso. Los picos de congestión no están igualmente equiespaciados, además de tener diferentes alturas. En la transmisión en el canal físico se observa que el envío no tiene una periodicidad tan grande como el primero, y las transmisiones tampoco tienen la misma duración.

Aun así, a simple vista se puede apreciar como existe una relación importante en ambos casos entre el nivel de ocupación del buffer RLC y el retardo de los paquetes transmitidos.

# 4

# Soluciones analizadas

En este capítulo se hace un estudio de los parámetros que se van a utilizar, así como de las diferentes soluciones propuestas para predecir el nivel de congestión.

Con el objetivo de simplificar la tarea de predicción de la congestión se ha decidido reducir los valores continuos de congestión a niveles discretos, de esta forma los algoritmos de aprendizaje no supervisado pueden dividir los datos en *clusters*, cada uno de los cuales se corresponderá con un nivel de congestión.

A pesar de tener una agrupación correcta de los valores por grupos, los algoritmos no supervisados son incapaces de conocer qué grupo corresponde a que nivel. De hecho, el aprendizaje no supervisado se caracteriza por agrupar sin la necesidad de utilizar etiquetas en los datos de entrenamiento, pero la desventaja es que la salida de estos algoritmos requiere interpretar, mediante otros mecanismos, estos grupos creados.

Debido a la gran relación que existe entre el *delay* y la congestión (Ver Sec. 4.3) se pueden asignar etiquetas a los clusters que nos devuelve el algoritmo en función del valor medio de *delay* en cada uno de los grupos.

De esta forma se puede utilizar una solución basada en aprendizaje no supervisado junto con la asignación de etiquetas basada en el valor de *delay*, como substitución de una solución basada en aprendizaje supervisado.

Esta idea ha sido el razonamiento utilizado en la mayor parte del trabajo, ya que, desde el punto de vista de la implementación en los nodos de la red, el hecho de usar una

solución basada en aprendizaje no supervisado permite el trabajo de manera autónoma del modelo.

A continuación, se describirán las herramientas analizadas, así como el procesado realizado sobre los datos previo a la aplicación de los algoritmos de predicción. Seguidamente, se realizará un análisis de los datos y se presentarán los algoritmos utilizados para predecir el nivel de congestión.

## 4.1. Herramientas utilizadas

El trabajo ha sido realizado mayoritariamente usando el lenguaje de programación Python. Este es complementado en ocasiones con MATLAB para la elaboración de algunos gráficos concretos. Las principales librerías de Python utilizadas son las siguientes:

- Matplotlib: Utilizada para graficar resultados.
- Numpy: Librería para operaciones matemáticas y la manipulación de datos numéricos.
- Scipy: Este módulo ofrece un mayor número de operaciones no contenidas en la librería anterior, así como herramientas estadísticas, entre otras utilidades.
- Pandas: Librería especializada en el manejo de datos y su organización en tablas.
- Sklearn: Contiene numerosas soluciones de *Machine Learning (ML)*, así como de preprocesamiento de los datos.
- Tensorflow: Librería especializada en el manejo de tensores (matrices multidimensionales), usada para *Deep Learning* mediante Redes Neuronales.

## 4.2. Pre-procesado de los datos

Antes de implementar los algoritmos utilizados para predecir el nivel de congestión es necesario estudiar qué parámetros son útiles para predecir dicha congestión, así como cuál es la mejor forma de tratarlos.

En primer lugar, en la mayoría las soluciones basadas en ML, ya sea de regresión lineal, clasificación, predicción, etc, es aconsejable normalizar los valores de entrada. En casos donde las variables de entrada difieren mucho en la magnitud de sus valores pueden producirse un funcionamiento inadecuado de los algoritmos. Además, normalizar los valores

suele provocar que el algoritmo converja con mayor rapidez a la solución. Por otro lado, algunos algoritmos requieren datos de entrada cuya media sea 0 para funcionar.

Por este motivo se han normalizado los valores de entrada de la forma más habitual, estandarizando, que consiste en restar a cada valor la media del conjunto y dividir por la desviación típica. De este modo cada muestra se actualiza de acuerdo con la siguiente expresión: $X_i = \frac{X_i - \mu}{\sigma}$.

Este procedimiento se puede realizar fácilmente usando el objeto *StandardEscaler* dentro de la clase *scikit.preprocessing* de la librería *Sklearn*.

Normalmente antes de realizar este proceso es recomendable eliminar los *outliers* de los datos, es decir eliminar aquellos valores que difieren mucho del resto de valores. Este proceso suele realizarse filtrando aquellos valores cuyo $Z_{score}$ [1] sobrepase un umbral establecido o eliminando valores en los extremos según un percentil (normalmente entorno a un 5 %).

En nuestro caso además de no existir apenas *outliers*, los mismos coinciden con situaciones de alta congestión o congestión muy baja, por lo que no es necesario suprimir estos valores ya que no provocan estimaciones erróneas, al contrario, pueden ser útiles para detectar situaciones en los extremos del valor de congestión.

Por otra parte, dado que se trata de series de datos temporales la riqueza de los datos es mayor que el propio valor de las métricas en un instante dado. Por este motivo, tanto para el retardo como para el IaT, se han calculado la desviación típica móvil y la media móvil de los datos de entrada, lo que nos ha permitido extender el conjunto de métricas sobre las que aplicar los algoritmos.

El tamaño de la ventana para el cálculo de estos estadísticos es aquel que proporciona el máximo valor de correlación entre el valor de congestión y los estadísticos utilizados.

Utilizando estos valores se obtienen los siguientes valores de correlación para el caso de la desviación típica del IaT, Figura 4.1.

Los valores de correlación calculados en todo el trabajo hacen referencia al coeficiente de correlación de *Pearson*, $\rho_{X,Y} = \frac{\sigma_{XY}}{\sigma_X \sigma_Y}$, el cual toma valores entre -1 y 1, siendo 1 totalmente correlados y 0 totalmente incorrelados; además el signo indica el sentido de crecimiento/decrecimiento comparando ambos valores.

Se puede ver como para el caso de distancia de 10 y 20 metros la correlación de este valor es muy baja, por lo que para estos escenarios se ha optado por no usar estos estadísticos.

[1] La métrica $Z_{score}$ indica la distancia de un valor respecto a la media medido mediante un número de veces la desviación típica

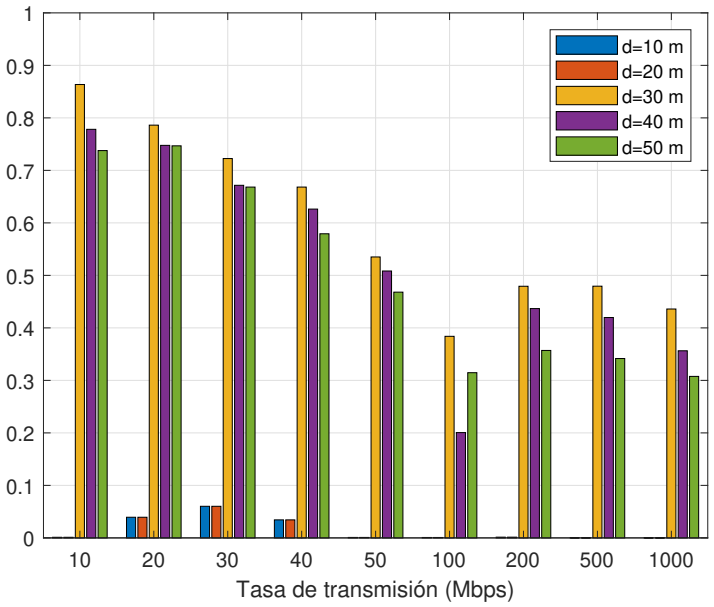


Figura 4.1: Correlación con los tamaños de ventana escogidos

El tamaño de ventana difiere para cada uno de los escenarios, estando relacionado con la tasa binaría y la distancia del usuario. Los tamaños con los cuales se obtiene la mayor correlación con el valor de la congestión se pueden ver en la Figura 4.2. Aunque en este trabajo se ha usado un único tamaño de ventana, se podrían usar varios de forma simultánea.

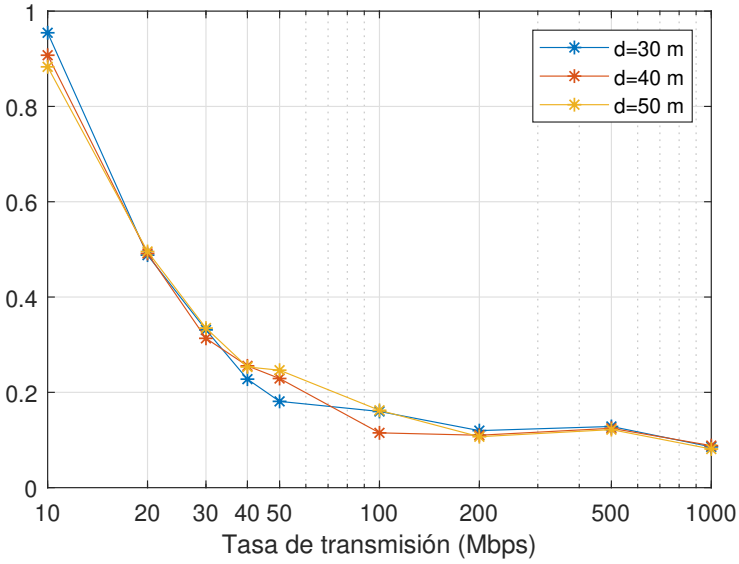


Figura 4.2: Tamaño óptimo de ventana para maximizar la correlación

## 4.3. Análisis de los datos

Una vez se han calculado los diferentes estadísticos se hace una comparativa global de los valores de correlación de cada una de las métricas. Dada la similitud de los resultados para las diferentes distancias, se ha optado por mostrar un único análisis en la Figura 4.3. En esta figura se muestran los valores de correlación tanto de los estadísticos originales (retardo e IaT), así como de los obtenidos mediante la media y desviación estándar móvil.

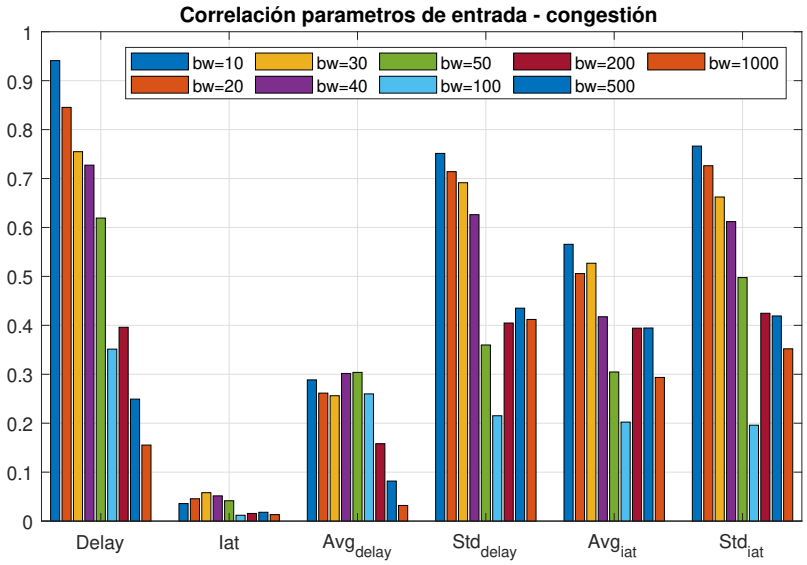


Figura 4.3: Correlación de los estadísticos obtenidos para una distancia de 40 m y diferentes tasas de envío

Se puede apreciar la diferencia en el valor de correlación entre las diferentes métricas. Como se puede ver el valor de correlación del IaT es muy bajo en todas las situaciones, al igual que la desviación típica y la media móvil del retardo.

Otro aspecto que se puede apreciar es cómo la correlación de todos los valores tiende a descender a medida que aumentamos la tasa de envío. Esta característica es más apreciable en el delay que en otros valores, los cuales también tienen una tendencia descendente, pero presentan valles en el valor de correlación a determinadas tasas de transmisión.

Tras analizar la dependencia de la correlación para una distancia fija en función de la tasa de envío, se ha realizado el mismo análisis para una tasa de envío constante variando la distancia. En la Figura 4.4 se muestran los valores obtenidos para todas las métricas usando una tasa de envío de 50 Mbps.

El aspecto más notable de esta gráfica es el valor prácticamente nulo de correlación de todos los valores para distancia iguales o por debajo de los 20 m de distancia. Esto se puede explicar con el funcionamiento del modelo que se usa para simular el canal físico.

El modelo usado para obtener las trazas se basa en la distancia para establecer la probabilidad de LOS, la cual es igual a 1 para distancias inferiores a 20 m. El hecho de tener visión directa del transmisor repercute en el valor medio de la SINR, tal como se muestra en la Tabla 4.1. En última instancia, situaciones con una SINR muy favorable (10 m, 20 m) no presentan apenas congestión, en el enlace radio, por lo que no hay correlación entre esta y las métricas utilizadas.

Tabla 4.1: Valores medios de SINR en función de la distancia

| Distancia(m) | SINR(dB) |
|---|---|
| 10 | 29.67 |
| 20 | 24.92 |
| 30 | 8.025 |
| 40 | 6.85 |
| 50 | 5.23 |

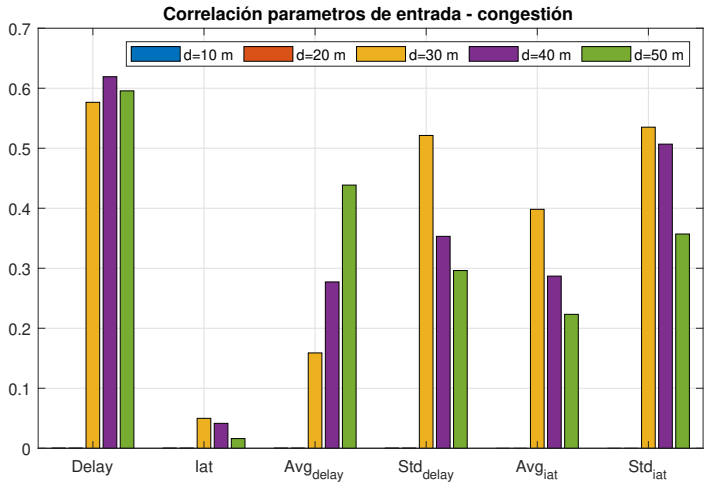


Figura 4.4: Correlación de las diferentes métricas para una tasa de envío de 50 Mbps

Hasta ahora se ha estudiado la correlación entre las diferentes métricas y la congestión. Sin embargo, otro factor a tener en cuenta cuando se hace uso de múltiples variables de entrada es la multicolinealidad, la cual indica el grado de similitud o relación entre las variables de entrada. Si 2 variables de entrada presentan un alto grado de correlación el uso de ambas no aporta valor añadido al modelo.

De hecho, en algunos modelos, como la regresión lineal, la multicolinealidad puede tener efectos adversos sobre la predicción, mientras que en el caso del *clustering* no afecta negativamente.

En la Figura 4.5 se puede ver una imagen que representa el nivel de correlación de

las diferentes posibles métricas y la congestión entre sí, para el caso de una tasa de envío de 30 Mbps a una distancia de 30 m.

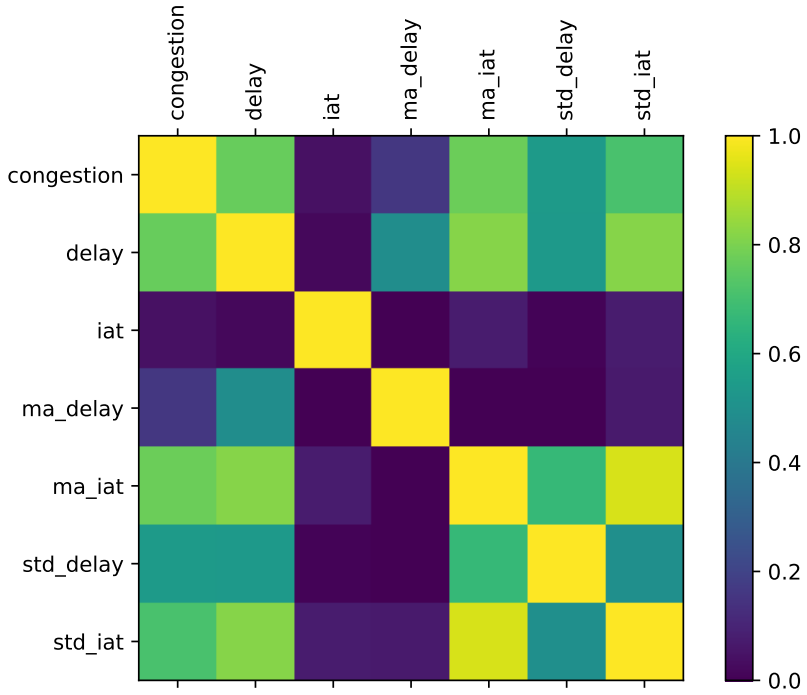


Figura 4.5: Correlación entre los diferentes valores

Aunque, al igual que en la Figura 4.3, vemos los mismos valores con un alto nivel de correlación con la congestión ($delay$,$Avg_{iat}$,$Std_{iat}$). Sin embargo, se aprecia como estos 3 están altamente correlados entre sí, por lo que el uso de unos u otros parámetros, o la combinación de estos no produciría grandes cambios en los resultados.

Se han estudiado diversas opciones haciendo diferentes combinaciones de estos valores con objeto de encontrar aquella combinación más eficiente de estos valores.

Existen mecanismos más avanzados para el tratamiento de múltiples entradas como puede ser el análisis Principal Component Analysis (PCA), el cual realiza una proyección de los datos en un subespacio de menor dimensión basado en minimizar el error de mínimos cuadrados entre los puntos de entrada y la base sobre la que se define el subespacio, de esta forma se consiguen variables ortogonales entre sí y por tanto incorreladas.

Estos métodos no han sido implementados en este trabajo, entre otros motivos, por tratar de elaborar una solución sencilla para no incrementar la carga computacional de la solución. Por otro lado, dado que parte del trabajo se ha dedicado a la generación de nuevas métricas, el análisis llevado a cabo nos permite saber qué enfoques resultan más idóneos.

## 4.4. Test estadísticos

Los test estadísticos o el contraste de hipótesis son una herramienta de la inferencia estadística utilizada para comprobar si una determinada población estadística cumple una característica específica, la cual se puede comprobar con un test estadístico.

El funcionamiento se basa en contrastar la veracidad o falsedad de una hipótesis planteada, llamada hipótesis nula ($H_0$). El funcionamiento de los test de hipótesis está basado en estimar el *p-value*, este valor representa la probabilidad de que ocurra un resultado concreto o resultados más allá de dicho valor. En este caso se calcula el *p-value* asumiendo que la hipótesis $H_0$ es cierta. Si el valor obtenido es muy bajo se puede rechazar la hipótesis nula (basado en el umbral de la región de rechazo elegida[2]); sin embargo, valores altos no garantizan que se cumpla la hipótesis $H_0$.

Hay numerosos test estadísticos utilizados para comparar poblaciones estadísticas entre sí o para comprobar si cumplen determinadas características.

En este trabajo se han comprobado dos características esenciales para poder realizar estimaciones temporales: La estacionalidad de los datos y la causalidad entre las variables de entrada y el valor a predecir.

### 4.4.1. Estacionalidad

Un proceso estacionario se define como un proceso estocástico cuya distribución de probabilidad no varía con el tiempo, parámetros como la media o la varianza permanecen estables. El ejemplo más claro de un proceso estacionario es el ruido blanco.

Para poder realizar una estimación es un requisito que los datos presenten un alto grado de estacionalidad, aunque en series temporales donde existe una tendencia en los datos (no estacionalidad) esto se puede solucionar aplicando diferentes técnicas como la derivada sobre los datos.

Para comprobar esta característica se han usado 2 test estadísticos diferentes:

- Test de Dickey-Fuller aumentada (ADF)[18]
- Test Kwiatkowski–Phillips–Schmidt–Shin (KPSS) [19]

Ambos test reflejan unos resultados aproximados de 0 y 0.03 respectivamente, con ligeras variaciones entre los diferentes escenarios. Con lo que se puede considerar que los datos forman una serie temporal estacionaria.

[2]Un umbral típico de rechazo puede ser $p = 0.05$

### 4.4.2. Causalidad

Dos variables aleatorias pueden estar correlacionadas entre sí. Sin embargo, la existencia de correlación no implica causalidad[3].

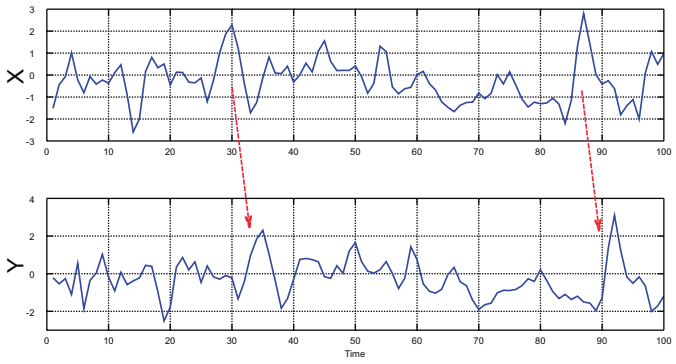


Figura 4.6: Ejemplo de causalidad entre 2 series temporales[2]

Para corroborar si existe causalidad entre los datos se ha usado la prueba de causalidad de Granger. Este test está formado por varios t-test y F-test usando versiones desplazadas en el tiempo de ambas secuencias[20].

Se ha probado este test sobre la pareja **congestión-delay** en ambos sentidos, dando como resultado un *p-value* en el primer caso de 0 y en el segundo caso de 0.04. Esto indica que el delay está producido por la congestión, hecho que parece ser razonable. Sin embargo, también parece existir cierta relación en el caso inverso, esto es lo que se denomina una causalidad bidireccional.

Los resultados obtenidos en estos test demuestran la posibilidad de predecir valores futuros de las variables haciendo uso de los valores actuales y pasados, así como la relación que existe entre las variables de entrada y la variable a predecir.

## 4.5. Algoritmos utilizados

A la hora de aplicar los diferentes algoritmos, se ha pensado una solución donde, en lugar de calcular el nivel de congestión exacto, se discretice en varios niveles. De esta forma se facilita mucho las tareas de computación y la hace más adecuada para los algoritmos utilizados.

[3]Aunque la relación opuesta si se cumple, la causalidad implica correlación

En concreto, se ha realizado tanto división binaría, que se asemejaría al funcionamiento del campo ECN, como una clasificación en 4 niveles. Los motivos para utilizar estos valores están descritos a lo largo de esta sección.

En el caso de soluciones no-supervisadas, una vez ejecutado el algoritmo correspondiente y obtenido el conjunto de clusters, se les asignan las etiquetas basándose en los valores de *delay* promedio de cada uno de ellos, correspondiéndose los valores más altos con los valores del extremo superior de los niveles de congestión.

Además de estas soluciones no supervisadas, se han estudiado otras opciones de aprendizaje supervisado con el objetivo de comparar el rendimiento de ambos enfoques.

Los diferentes algoritmos utilizados para aprendizaje no supervisado son:

- Kmeans
- Clustering jerárquico (basado en el Kmeans)
- Cadenas ocultas de Markov (HMM)

En lo que respecta al aprendizaje supervisado se han estudiado dos modelos:

- Support Vector Machine (SVM), concretamente enfocado a la clasificación Support Vector Clasiffication (SVC)
- Redes neuronales recurrentes (RNN)[4]

### 4.5.1. Kmeans

Es uno de los algoritmos para clustering más utilizados debido principalmente a su buena relación sencillez/resultados. Está pensado para ser usado con datos discretos, que no guardan autocorrelación temporal entre sí ni tienen ninguna estructura temporal. Sin embargo, en este caso a pesar de tener una serie temporal de datos, el algoritmo sigue teniendo unas prestaciones adecuadas.

Se trata de un algoritmo que separa un conjunto de datos en k grupos, parámetro especificado previamente. El criterio que utiliza este algoritmo para la agrupación es minimizar la suma de los cuadrados en cada grupo Within-cluster Sum of Squares (WCSS), este valor se denomina inercia.

[4]En este caso en lugar de predecir niveles de congestión se trata de predecir el valor exacto de la misma

Cada uno de los grupos está definido por el valor medio de sus muestras, llamado centroide. De esta forma se busca una solución iterativa donde se usa la distancia de los puntos de cada clúster al centroide como parámetro a minimizar. De esta forma se llega a una solución que minimiza la inercia.

$$argmin(S) \sum_{i=1}^{k} \sum_{x \in S_i} ||x - \mu_i||^2 \tag{4.1}$$

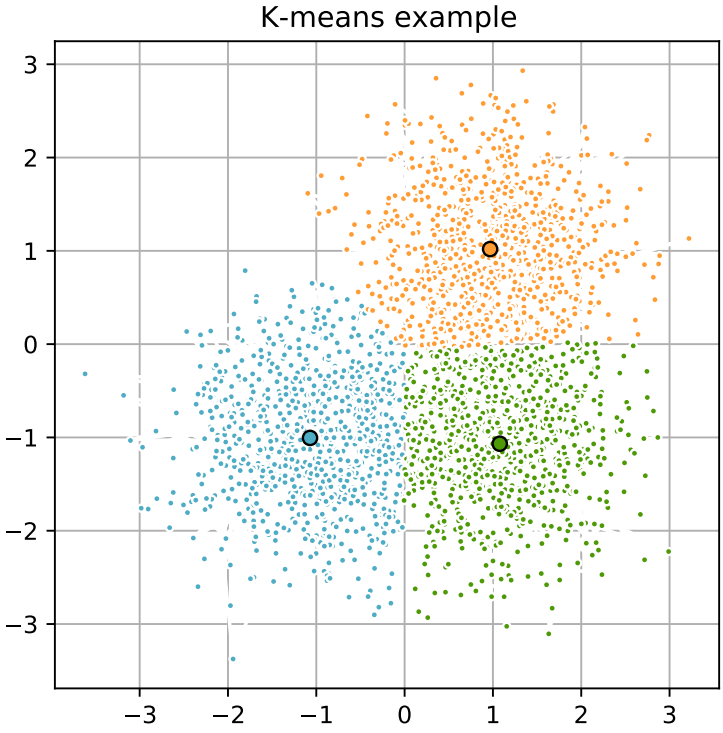


Figura 4.7: Ejemplo de algoritmo Kmeans

En la Figura 4.7 se muestra un ejemplo de solución obtenida con k-means. Como se puede observar, el algoritmo converge hacia soluciones donde la forma de los grupos tiende a ser circular y de tamaño similar. Esto es consecuencia de usar como métrica para elegir los grupos la distancia euclídea. Esto provoca que el algoritmo pueda no funcionar correctamente cuando existen formas irregulares o elongadas. En estos casos otros algoritmos como *Gaussian Mixture clustering* funcionan mejor.

El funcionamiento del algoritmo consiste en 3 pasos, el primero de ellos inicializa los

centroides de los grupos, después asigna cada muestra al grupo cuyo centroide está más cercano y, por último, computa de nuevo los centroides de los grupos. Estos 2 pasos se repiten hasta que los centroides de los grupos no superen un valor un umbral.

Uno de los inconvenientes que presenta este algoritmo es la necesidad de definir el número de clusters de antemano.

Para solucionar este problema se ha utilizado el *Bayesian Index Criterion* (BIC) y el método del codo, donde se busca un equilibrio entre la precisión del modelo y la complejidad del modelo, en este caso medida mediante el número de grupos utilizados para la clasificación.

En las Figuras 4.8 y 4.9 se puede ver este valor para un caso de tasa de envío de 50 Mbps y una distancia de 40 m. Se aprecia cómo la puntuación desciende rápidamente en el primer paso, sin embargo, reduce su pendiente a partir de este punto. Esta figura es muy dependiente del escenario utilizado. Sin embargo, el valor critico donde se produce un cambio importante en la pendiente suele oscilar entre el uso de 2-4 clusters. Este análisis nos ha servido para elegir el número de niveles de congestión que se evaluarán en el siguiente capítulo.

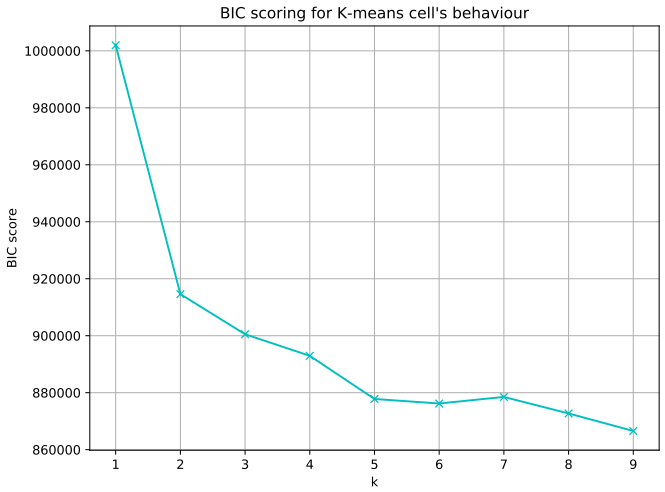


Figura 4.8: Tasa de envío 50 Mbps, dist = 40 m

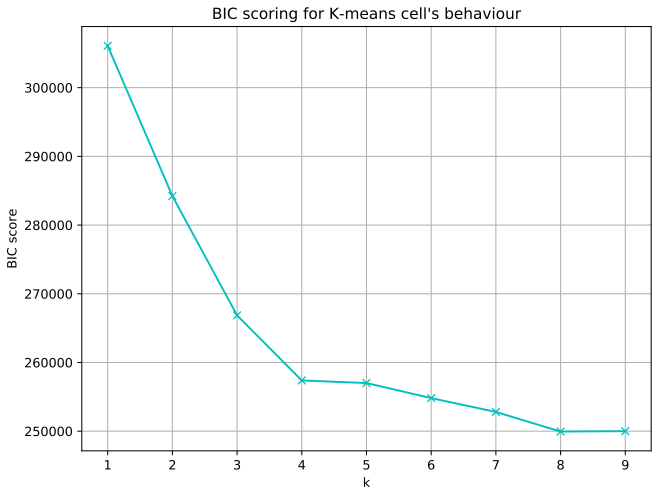


Figura 4.9: Tasa de envío 50 Mbps, dist = 50 m

Para aplicar esta solución se ha utilizado la librería *scikit* que contiene numerosos modelos ya definidos con intención de facilitar el uso de estos.

Como se verá en el próximo capitulo, en primera instancia se ha realizado un análisis basado en una clasificación binaría (Congestión, No congestión) mediante el uso de k-means. Sin embargo, con la intención de obtener una mayor precisión en las predicciones en los casos en los que se han definido varios niveles de congestión, se ha probado una solución jerárquica que se describe a continuación.

### 4.5.2. Clustering jerárquico

Hay numerosas formas de realizar un clustering jerárquico, la principal diferencia es la metodología llevada a cabo para la agrupación de los clusters.

Se puede partir de un solo clúster e ir creando subdivisiones de estos, usando una metodología *up-bottom*, también llamada divisiva. O, por el contrario, se puede empezar con un número elevado de clusters e ir agrupándolos, creando grupos mayores (*bottom-up*), metodología llamada aglomerativa.

En este trabajo parece tener más sentido una división *up-bottom*. Cabe preguntarse hasta qué punto merece la pena subdividir los grupos anteriores, es decir ¿es suficientemente grande la mejora en precisión como para aumentar el costo computacional?

Una forma gráfica de ver esta situación es a través de un dendograma. Este tipo de gráfico realiza una representación en forma de árbol de los diferentes grupos, indicando la

distancia entre los grupos mediante la distancia vertical entre los clusters. En la Figura 4.10 se muestra el dendograma obtenido para el escenario de 50 m de distancia y tasa de datos de 50 Mbps.

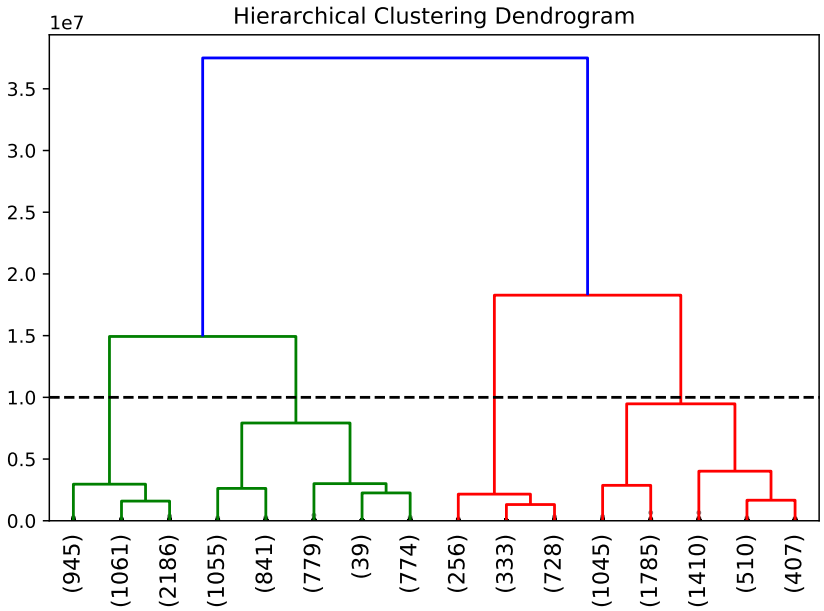


Figura 4.10: Dendograma para un escenario: BW = 50 Mbps, dist = 50 m

Se puede apreciar cómo existe mucha "distancia" entre los 2 grupos más grandes, esta distancia sigue siendo relativamente grande para el caso de 4 grupos. Sin embargo, a partir de 4 los grupos están próximos, lo que va a aumentar poco la precisión respecto a dividir en 4. Por otro lado, sí que produce un incremento sustancial del tiempo de computación, por lo que 4 niveles es el número elegido.

Para implementar esta solución se ha utilizado el algoritmo Kmeans de forma iterativa, realizando nuevas divisiones sobre los clusters ya obtenidos.

Existen otras soluciones que utilizan otra métrica diferente a la distancia euclídea, pero en el clustering jerárquico cobra todavía más importancia usar un algoritmo eficiente en términos de tiempo de computación ya que la complejidad aumenta en gran medida con el número de clusters.

### 4.5.3. Cadenas ocultas de Markov

El Hidden Markov Model (HMM) se trata de un modelo estadístico ampliamente utilizado en el campo de reconocimiento de formas temporales. Algunos de sus campos de aplicación pueden ser el reconocimiento del habla [21], reconocimiento de escritura o gestos

[22].

Se trata de un modelo estadístico en el cual existen una serie de estados internos $Z$, que no pueden ser observados directamente, los cuales generan una serie de variables observables $Y$. Se asume que los estados internos tienen la forma de una cadena de Markov de primer orden, esto significa que el estado del modelo en el instante $(t)$ solo depende del estado en el instante $(t-1)$.

El modelo está representado por la siguiente tupla $(Q, \pi, A, \theta)$, donde:

- $Q$ representa el conjunto de estados que conforman la cadena $Q = \{1, 2, ..., N\}$. En este caso el número de estados coincide con el número de clusters que se desean.
- $\pi$: Vector que representa las probabilidades iniciales del sistema de encontrarse en cada uno de los estados.
- $A$: Matriz de probabilidades de transición entre los diferentes estados.
- $\theta$: Es una matriz con los parámetros de la distribución de probabilidad de los observables, estos valores están condicionados al estado de la cadena oculta en el que se encuentran. $\theta(Q) = P(Y = y|Q = Q_i)$

Este modelo tiene 2 usos fundamentales:

- Estando definidos los diferentes parámetros del modelo, estimar la secuencia de estados más probable que puede haber generado una determinada secuencia de observables. Esta tarea suele realizarse con el algoritmo de Viterbi, aunque también puede usarse como criterio la estimación máxima a posteriori (MAP).
- Dado una secuencia de observables estimar las probabilidades de transición de los estados, así como los parámetros que determinan las probabilidades de emisión. Este problema se suele abordar usando el algoritmo iterativo de *Expectation-Maximization* (EM), también conocido como algoritmo de Baum-Welch[5].

Normalmente la función de distribución de emisión de los estados suele ser una función discreta o Gaussiana, otra alternativa utilizada es la mezcla de varias gaussianas.

Se puede ver una representación gráfica de la estructura de HMM en la Figura 4.11

[5]Se trata de un algoritmo iterativo el cual trata de obtener el modelo que mejor represente los datos de entrada, al tratarse de un algoritmo iterativo puede converger hacia un mínimo local por lo que es aconsejable utilizarlo varias veces con diferentes inicializaciones.

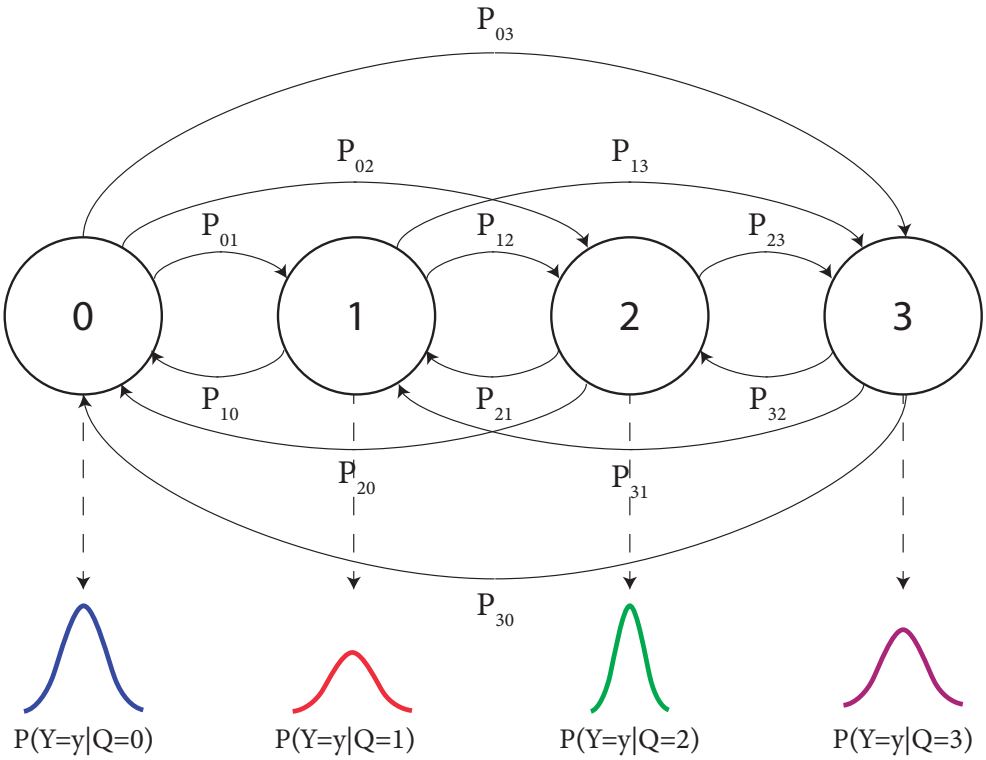


Figura 4.11: Representación gráfica HMM

### 4.5.4. Support Vector Machine (SVM)

Se trata de un algoritmo de aprendizaje supervisado ampliamente utilizado para tareas como: regresión, clasificación y detección de valores atípicos. Aunque principalmente es utilizado para clasificación en base a etiquetas.

Suponiendo una clasificación binaria en 2 grupos, el algoritmo trata de definir un hiperplano que separe los dos conjuntos de la manera más precisa posible. Para esta tarea trata de maximizar el margen, que se define como la distancia entre la frontera de decisión y el punto más cercano a la misma del grupo. Los puntos de los grupos más cercanos al hiperplano componen los vectores de soporte.

Este funcionamiento se puede extender a clasificación en varios grupos, más de 2, usando una estrategia denominada "*one-against-one*" donde se entrenan varios clasificadores binarios entre los pares de grupos. Para un caso de $n$ grupos diferentes son necesarios $n\frac{n-1}{2}$ clasificadores, cada uno entrenado de manera binaria.

Es posible extender este algoritmo para agrupar datos cuando la frontera de decisión no es lineal. Para esto se utilizan las funciones Kernel, que proyectan los datos en un espacio dimensionalmente mayor, sustituyendo el producto escalar por otro tipo de operación no

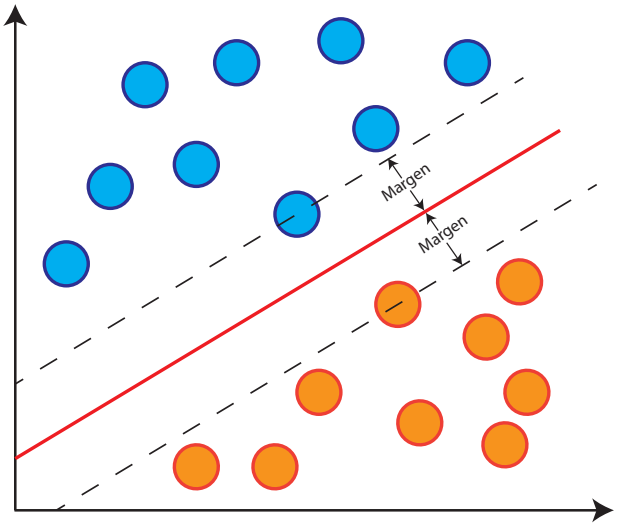


Figura 4.12: Funcionamiento algoritmo SVM

lineal. Un ejemplo puede ser utilizar un kernel basado en la tangente hiperbólica $(\vec{x_i} \cdot \vec{x_j}) = tanh(k\vec{x_i} \cdot \vec{x_j})$. De esta forma se pueden conseguir regiones de decisión cuya forma no sea lineal.

Existen otras implementaciones, como soluciones basadas en "*soft decisions*", que no se van a tratar aquí.

Al tratarse de un algoritmo supervisado necesita datos etiquetados para su entrenamiento, en este trabajo se usará un 15 % de los datos para entrenar el modelo.

### 4.5.5. Redes neuronales recurrentes (RNN)

Hoy en día las redes neuronales son posiblemente la técnica más utilizada en el campo del *Machine Learning* debido a la capacidad que tienen a ajustarse a los datos de entrenamiento, así como a su capacidad para poder descubrir patrones no perceptibles a través de otras técnicas.

Las redes neuronales hacen uso de una arquitectura compuesta por una gran cantidad de neuronas, organizadas normalmente en capas. Estas neuronas, realizan una operación sencilla como puede ser multiplicar por un peso y sumarle un valor.

Este resultado se usa como valor de entrada para una función no lineal. Las funciones utilizadas deben ser derivables, característica necesaria para que se pueda computar el gradiente, elemento clave para el proceso de entrenamiento de estas redes.

Algunas de las funciones no lineales más utilizadas son la función ReLu, sigmoide o

la tangente hiperbólica, entre otras. En la Figura 4.13 se muestra un ejemplo de cada una de ellas.

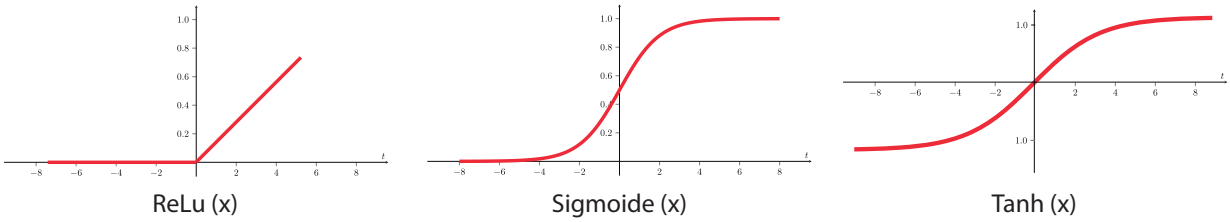


Figura 4.13: Funciones más utilizadas en redes neuronales

Estas redes utilizan métodos iterativos, como el descenso de gradiente, para llegar a una solución subóptima[6].

Dentro de este campo existe un tipo de redes, llamado redes neuronales recurrentes, las cuales además de realizar las operaciones anteriores guardan y transmiten internamente un valor de estado. Este valor ayuda a la red a "aprender" estructuras muy útiles para series temporales.

Concretamente en este trabajo se han usado neuronas del tipo Long Short Term Memory (LSTM), que realizan una serie de operaciones internas que mejoran el rendimiento respecto a la estructura de neurona simple.

La estructura utilizada se muestra en la Figura 4.14 está compuesta por las siguientes capas:

- Capa LSTM recurrente compuesta por 100 neuronas.
- Capa LSTM recurrente compuesta por 100 neuronas.
- Capa *fully-conected* de 50 neuronas.
- Neurona de salida

Este modelo, a diferencia de los anteriores, predice un valor concreto de congestión en lugar de un nivel. A partir de los valores predichos se puede conocer el nivel de congestión que le asigna el modelo. Esto se lleva a cabo para poder realizar una comparativa entre los diferentes modelos.

Este modelo puede extenderse para tratar no solo de deducir el nivel de congestión actual, sino tratar de predecir el estado de congestión en diferentes instantes del futuro. Esta característica puede ser muy útil de cara al control de la congestión.

[6]Se denomina solución subóptima a aquellas soluciones alcanzas donde el valor obtenido no se garantiza que sea el máximo global, ya que el algoritmo puede haber convergido hacia un mínimo local

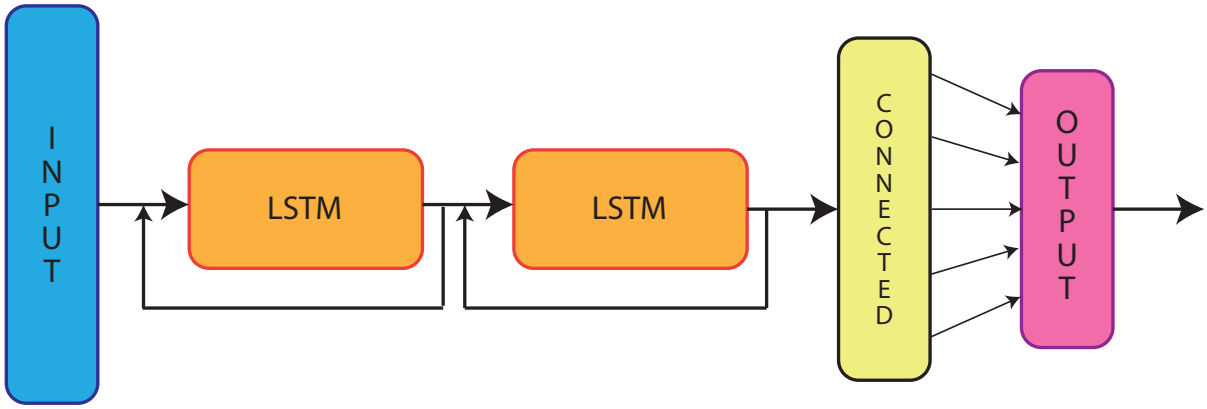


Figura 4.14: Estructura de la red neuronal utilizada

# 5 Resultados

Este capítulo describe los métodos utilizados para medir el rendimiento de los diferentes algoritmos propuestos. Así como el funcionamiento de cada uno de ellos explicado en detalle.

## 5.1. Variables utilizadas y algoritmos empleados

En la Sección 4.3 se ha realizado un análisis de la "utilidad" de cada uno de los parámetros de entrada con relación a la predicción del nivel de la congestión del sistema.

Se han realizado diferentes pruebas con diferentes combinaciones de variables de entrada con objeto de determinar aquellas que mejores resultados producen. Los resultados se degradan a medida que se aumenta la distancia entre emisor y receptor y/o se aumenta la tasa binaria, aunque en situaciones de distancias muy cortas el rendimiento tampoco es bueno.

Por este motivo los resultados que se van a presentar comprenden distancias entre 30 y 50 metros y tasas de transmisión desde 10 hasta 1000 Mbps.

Las variables utilizadas son el **delay** entre paquetes y la **desviación típica del IaT**. La combinación de estas 2 métricas es la que mejores resultados ofrece, aunque en algunos algoritmos se puede estudiar el alimentar al mismo con más variables, como puede ser el

modelo basado en RNNs.

Los modelos utilizados se encuentran resumidos a continuación.

- Clasificación binaria:
  - K-means, clustering no supervisado
  - SVC, clasificación supervisada
- Clasificación en 4 niveles:
  - Clustering jerárquico
  - SVC, clasificación supervisada
  - Cadenas ocultas de Markov (HMM)
- Predicción del valor de congestión:
  - Redes neuronales recurrentes

## 5.2. Métricas utilizadas para medir el rendimiento

La manera de cuantificar los resultados obtenidos de un modelo no es una acción trivial. Esta se realiza de manera diferente para el caso de la clasificación binaria, la clasificación multinivel y la predicción del valor exacto.

Para el caso de la clasificación las métricas se basan en los resultados de la matriz de confusión, donde se contrasta las predicciones realizadas con los verdaderos valores de los datos.

| | VALOR ESTIMADO | |
|---|---|---|
| VALOR REAL | True Negative | False Positive |
| | False Negative | True Positive |

Figura 5.1: Componentes de la matriz de confusión para el caso binario

En la Figura 5.1 se muestra la estructura de la matriz de confusión de un clasificador binario. Se puede observar que tiene 4 campos mediante los cuales se pueden utilizar métricas ya establecidas como son:

- *Precisión*: Este valor se puede entender como el porcentaje de veces que realmente el resultado es positivo cuando el modelo indica que es positivo. $Precision = \frac{TP}{TP+FP}$

- *Recall*: Mide el porcentaje de valores positivos correctos entre estos mismos y los valores negativos erróneos. $Recall = \frac{TP}{TP+FN}$
- *F1-Score*: Es una forma de combinar las dos métricas anteriores en una sola, suele usarse cuando las 2 métricas anteriores tienen el mismo peso $F1 = 2 \times \frac{Prec \times Recall}{Prec + Recall}$

Cuando se realiza una clasificación multinivel es más complicado medir el rendimiento.

En este caso se ha pensado en aplicar una penalización diferente en función de la distancia en niveles del valor correcto. De esta forma se calcula una pseudo-precisión donde los valores correctos son los más importantes, seguidos de los valores que se hayan predicho a distancia 1 y finalizando con los que se encuentran a distancia 2, siendo la distancia el número de niveles que difieren el valor predicho y el real. Así, la pseudo-precisión queda definida como sigue.

$$Pseudo - prec(S_i) = \frac{1}{S_i} \sum_{i=0}^{2} \frac{Spred_i}{2^i}$$

$$Pseudo{-}prec(S_i) = \frac{Spred_i + 0.5 \times (Spred_{i-1} + Spred_{i+1}) + 0.25 \times (Spred_{i-2} + Spred_{i+2})}{S_i}$$

Por último, para el caso de la predicción de valores se ha utilizado como métrica el Error Cuadrático Medio (MSE), que mide la diferencia entre la secuencia temporal predicha y la real. Con objeto de poder comparar este último método con los anteriores.

También se ha discretizado la secuencia de salida en niveles para poder aplicar las métricas anteriores y poder compararlas, esto se realiza con objeto de comparar los distintos modelos. Cabe destacar que el hecho de discretizar la salida solo tiene sentido en términos comparativos ya que se reduce la exactitud del modelo.

## 5.3. Evaluación de los algoritmos

### 5.3.1. Clasificación binaria

Para la clasificación binaria, en primer lugar, se ha utilizado el algoritmo k-means Como se ha comentado anteriormente este algoritmo no tiene en cuenta la relación temporal entre muestras más allá de los estadísticos que alimentan al algoritmo.

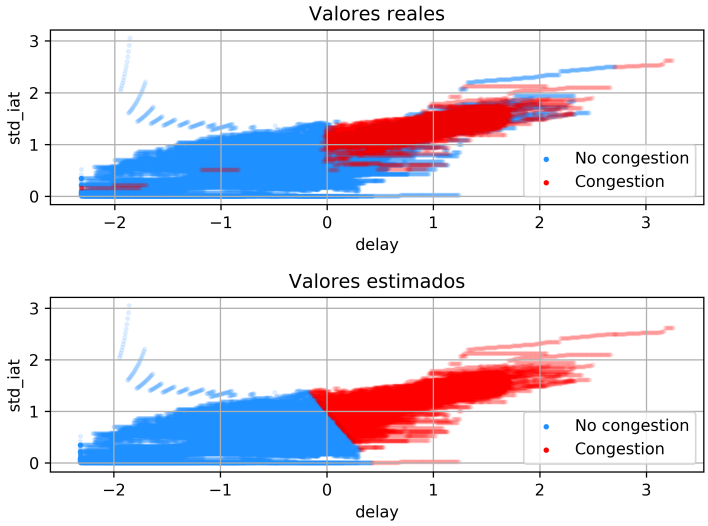


Figura 5.2: Representación tanto real como de los valores obtenidos con el algoritmo k-means

En la Figura 5.2 se puede ver una representación, en función de las variables de entrada, de los niveles predichos por el algoritmo k-means para una clasificación binaria con una tasa de envío de 30 Mbps y una distancia de 40 m.

En esta figura se puede apreciar cómo con el uso de este algoritmo aparecen 2 regiones perfectamente diferenciadas en los datos predichos. Sin embargo, esto no es lo que ocurre realmente. Tal como se puede ver en el gráfico superior, los valores de "No Congestión" se solapan con los valores de "Congestión" en determinadas zonas. Para complementar este resultado, en la Figura 5.3 se representa la evolución temporal del nivel de congestión, tanto el real como el predicho por k-means. Como se puede ver, el algoritmo tiene un rendimiento bueno.

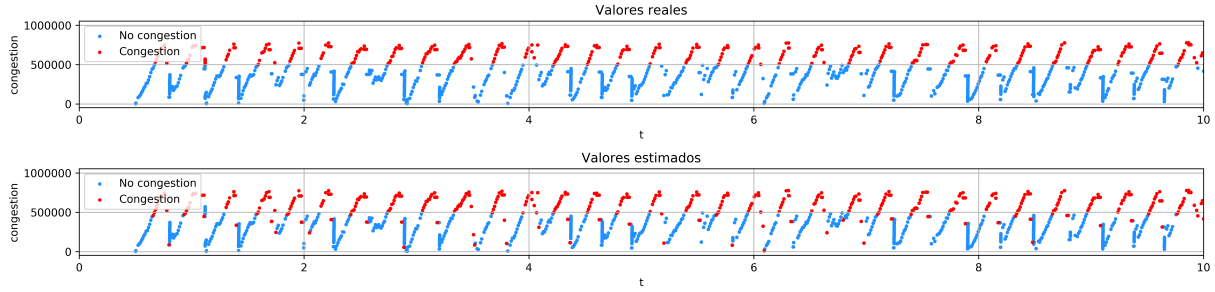


Figura 5.3: Niveles de congestión a lo largo del tiempo *Kmeans*

A fin de realizar un análisis más formal, seguidamente se analiza el rendimiento del algoritmo k-means usando las métricas anteriormente descritas. Estos resultados están comparados con el rendimiento que ofrece el algoritmo supervisado SVC sobre el mismo

escenario, los resultados para una distancia de 40 m se pueden ver en las Figuras 5.4 y 5.5.

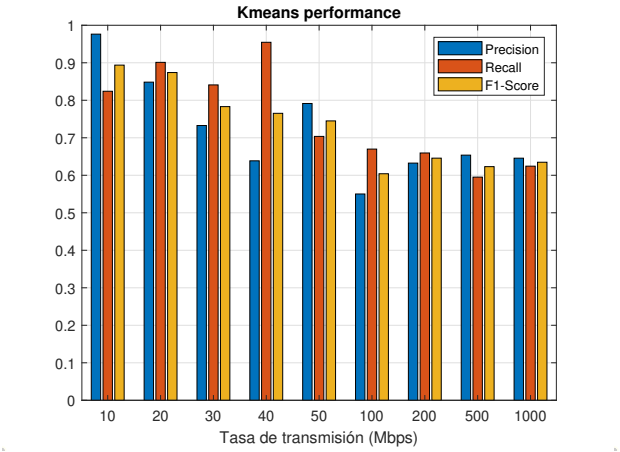


Figura 5.4: Rendimiento Kmeans (40 m)

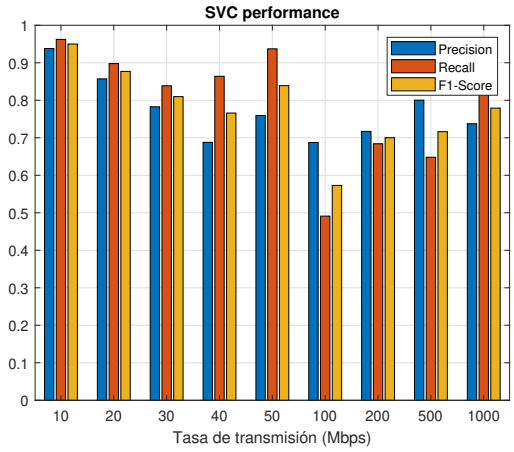


Figura 5.5: Rendimiento SVC (40 m)

Se puede ver cómo los resultados concuerdan con lo esperado, como ya se ha visto antes el nivel de correlación desciende a medida que se incrementa la tasa binaria. Por lo que es consecuente que también descienda la capacidad del modelo en escenarios más incorrelados.

También se observa cómo el rendimiento del algoritmo supervisado sobrepasa ligeramente al algoritmo no supervisado en la mayoría de escenarios. Este comportamiento tiene sentido ya que este segundo algoritmo hace uso de las etiquetas de los datos además de los *inputs* del modelo.

### 5.3.2. Clasificación en 4 niveles

#### Clustering jerárquico

Se ha utilizado el algoritmo K-means con un número de clusters igual a 2 de forma recurrente. De esta forma, en el caso de que el modelo interprete mal el nivel de congestión es probable que el estado real este contiguo al predicho.

En la Figura 5.6 se puede ver la matriz de confusión resultante de aplicar este modelo para un escenario de tasa de envío de 30 Mbps y una distancia de 40 m.

Se puede observar cómo el porcentaje de valores predichos correctamente (diagonal de la matriz) sigue siendo relativamente alto, también se puede ver cómo el resto de niveles predichos se encuentran contiguos al estado real.

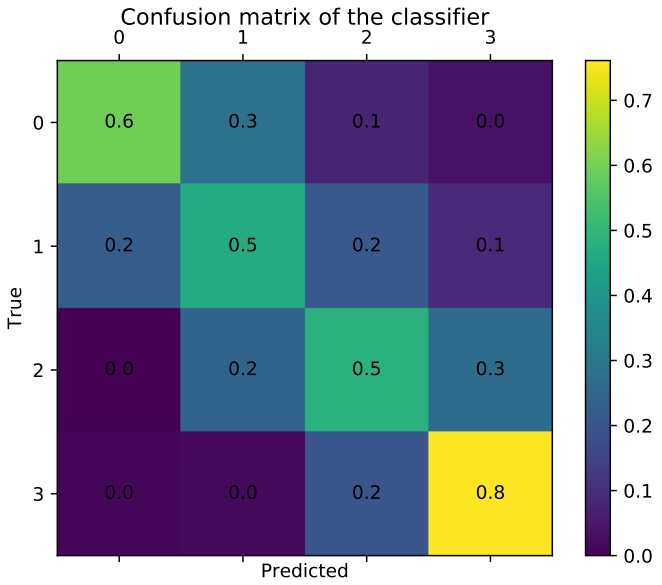


Figura 5.6: Matriz de confusión usando clustering jerárquico (BW=30, dist=40 m)

Para medir el rendimiento se va a utilizar la métrica descrita anteriormente para diferentes tasas binarias y distancia de 30, 40 y 50 m.

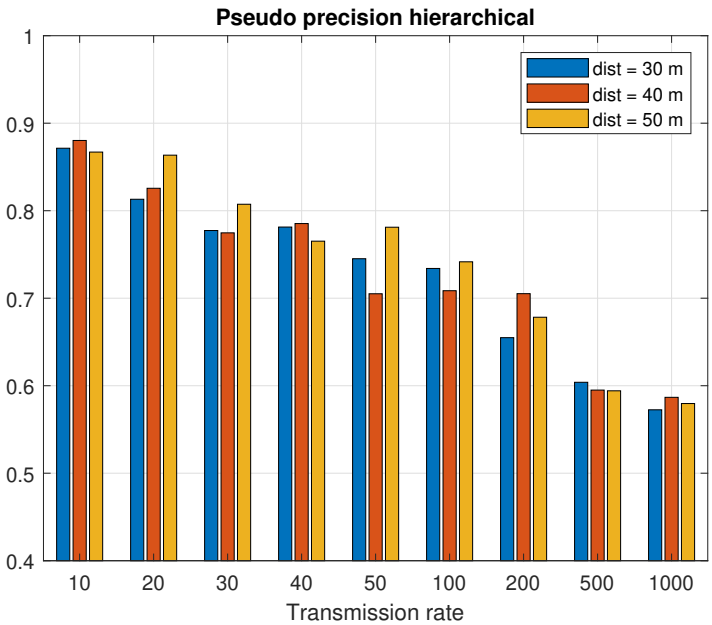


Figura 5.7: Rendimiento clustering jerárquico

En la Figura 5.7 se puede observar cómo esta métrica del rendimiento también se reduce a medida que se aumenta la tasa binaría. Sin embargo, parece ser menos sensible a cambios en la distancia. Esto puede ser debido a la propia forma en la que está definida la métrica.

### Cadenas ocultas de Markov (HMM)

Este modelo, en primer lugar, a través de un algoritmo iterativo, calcula los diferentes parámetros de la cadena de Markov como son las probabilidades de transición entre estados. También se encarga de calcular la media y varianza de las probabilidades de emisión de las métricas de entrada.

Una vez calculados estos parámetros del modelo, el sistema predice la secuencia de estados de la cadena de Markov de acuerdo a las variables observables.

La matriz de transición asociada al modelo para el mismo escenario que en los casos anteriores es la siguiente:

$$\begin{bmatrix} 0.99583 & 0.00383 & 0.00034 & 0 \\ 0.00485 & 0.99414 & 0.00101 & 0 \\ 0.00027 & 0.00632 & 0.98713 & 0.00628 \\ 0 & 0.00003 & 0.00435 & 0.99562 \end{bmatrix} \tag{5.1}$$

Matriz de transición entre estados

Los parámetros que modelan las probabilidades de emisión gaussianas inferidos por el modelo se indican a continuación:

$$\begin{bmatrix} 1.203 & 1.418 \\ -1.211 & 0.601 \\ 0.331 & 1.043 \\ -0.268 & 0.515 \end{bmatrix} \tag{5.2}$$

Media inferida de los *inputs*

$$\begin{bmatrix} 0.111 & 0.019 \\ 0.189 & 0.070 \\ 0.098 & 0.022 \\ 0.111 & 0.038 \end{bmatrix} \tag{5.3}$$

Varianza inferida de los *inputs*

En la Matriz 5.1 se puede ver cómo el modelo infiere que la transición entre estados se hace de manera que siempre se desplaza a un estado contiguo o se mantiene en el estado actual. Las prestaciones de este modelo pueden verse en la Figura 5.8.

El rendimiento del modelo es muy similar al anterior, aunque para altas tasas de transmisión se comporta ligeramente peor. Esta pérdida de rendimiento se acentúa en los escenarios de menor distancia para tasas altas.

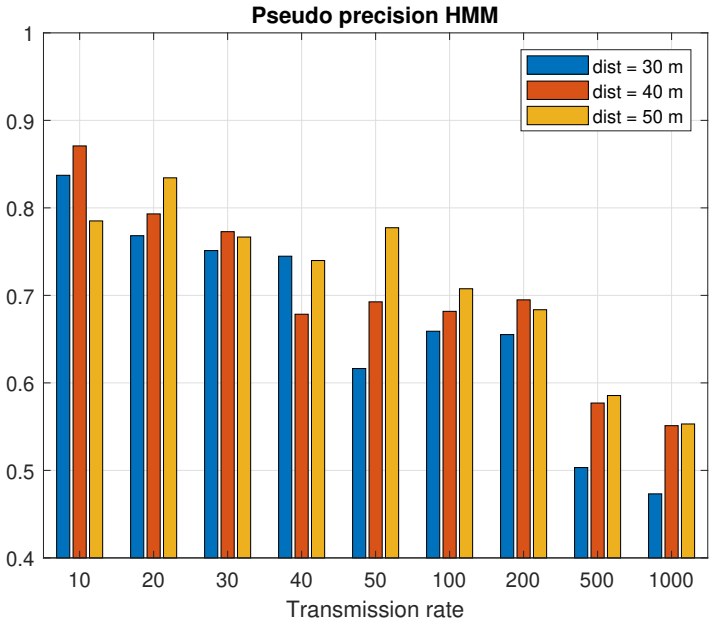


Figura 5.8: Pseudo-precisión del algoritmo HMM

### SVC

Además de los dos modelos presentados para la clasificación en 4 niveles, al igual que en el caso binario, estos resultados se han comparado con los obtenidos a través del algoritmo de aprendizaje supervisado SVC. El rendimiento obtenido en este caso es el que se muestra en la Figura 5.9.

Aproximadamente tiene la misma forma que las gráficas de rendimiento de los 2 modelos anteriores. Al igual que ocurre en el caso binario, el comportamiento del algoritmo de aprendizaje supervisado es ligeramente superior a las soluciones basadas en aprendizaje no supervisado.

Además, en la Figura 5.10, también se muestra una representación temporal de los valores de congestión junto con los niveles de congestión reales y predichos.

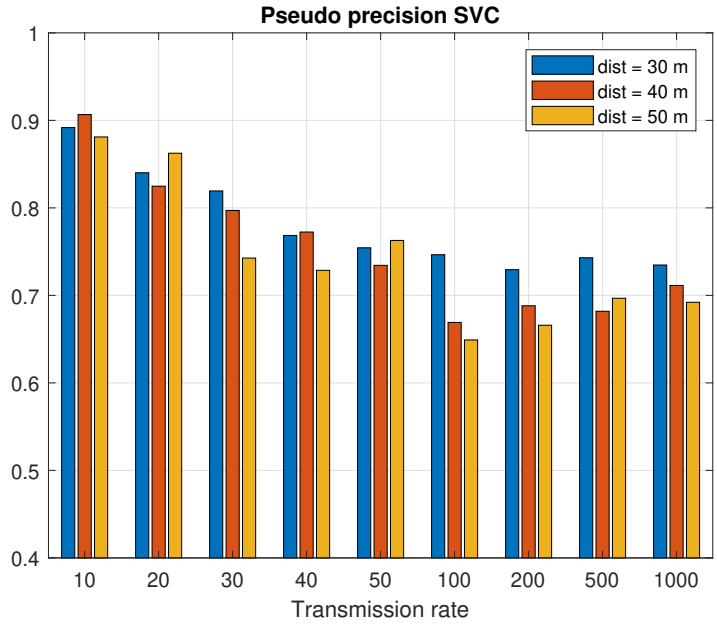


Figura 5.9: Rendimiento clustering supervisado SVC

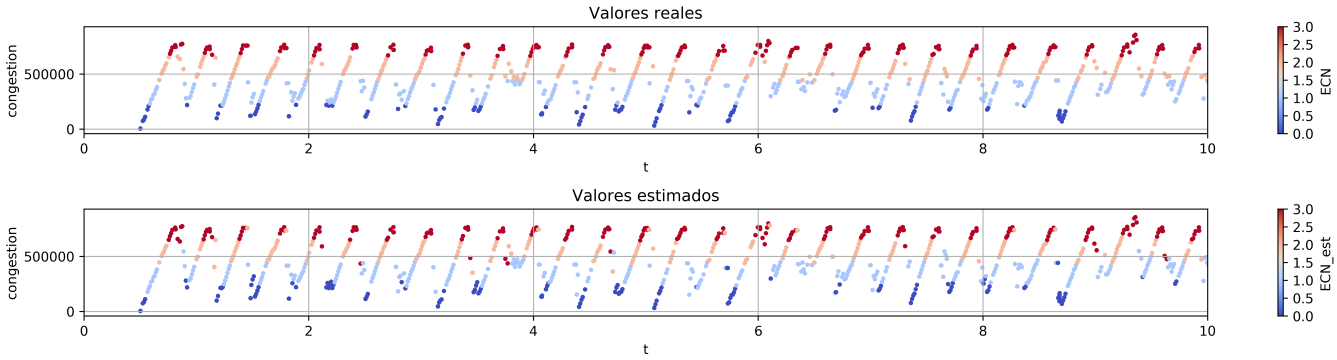


Figura 5.10: Representación temporal de los niveles asignados (SVC)

Se puede apreciar cómo los valores reales y los estimados difieren más que en el caso de la clasificación binaria, esta confusión parece darse especialmente con los niveles de los extremos. Este comportamiento es el esperado, ya que al aumentar el número de niveles discretos la precisión del modelo baja.

### 5.3.3. Predicción del valor de la congestión (RNN)

A diferencia de los casos anteriores este algoritmo predice un valor concreto en lugar de un nivel discreto. Para medir el rendimiento de este algoritmo se ha usado el MSE.

Los valores de MSE obtenidos para los diferentes escenarios se pueden ver en la Figura 5.11. Es importante destacar que estos valores se han obtenido reentrenando el mismo modelo sobre diferentes escenarios. Es posible que, para determinados escenarios, como los de tasas de transmisión altas, sea necesario cambiar ligeramente la arquitectura del modelo o realizar algún paso previo adicional para obtener un rendimiento superior. Sin embargo, aquí se ha utilizado la misma arquitectura de red neuronal para todos los escenarios[1].

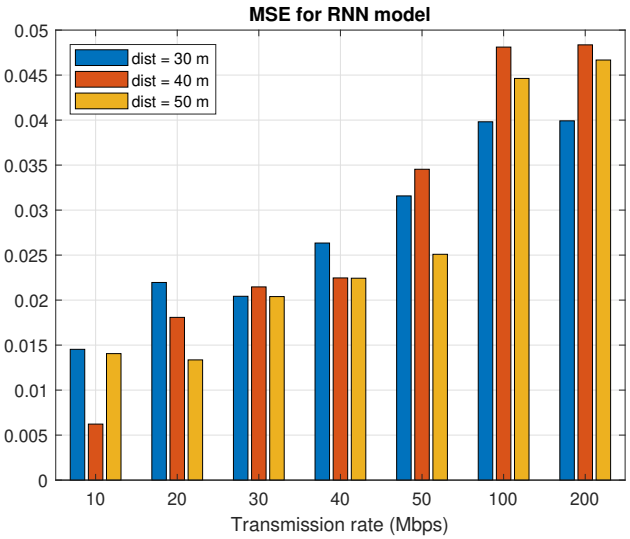


Figura 5.11: Valor de MSE para el modelo basado en redes neuronales

Se aprecia, al igual que en todos los casos anteriores cómo a medida que se aumenta la tasa de transmisión el valor de congestión se vuelve más difícil de predecir.

En principio, el problema que presenta este tipo de soluciones es el alto tiempo de entrenamiento que se requiere para obtener buenos resultados. Sin embargo, se ha realizado una prueba con un modelo entrenado sobre un escenario concreto (distancia de 40 m y tasa de envío de 40 Mbps) testeando su rendimiento sobre otros escenarios, tal como se muestra en la Figura 5.12

[1]No se han probado todos los escenarios debido al gran tiempo de entrenamiento que requiere esta solución

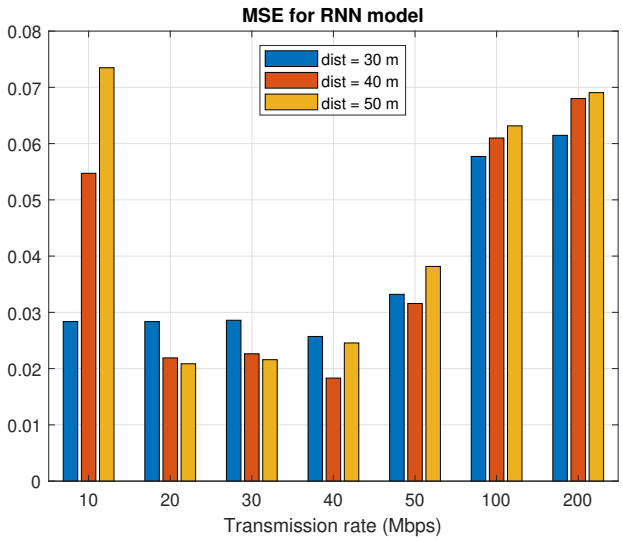


Figura 5.12: Valor de MSE para el modelo usando escenarios diferentes

Se puede apreciar cómo en los escenarios similares el rendimiento se mantiene aproximadamente igual, a pesar de haber sido entrenado con datos de otro entorno diferente.

Los resultados de esta prueba plantean, como una posible solución, un modelo más complejo basado en redes neuronales el cual pueda suministrarse pre-entrenado, y funcione bien en numerosas situaciones.

Se puede ver en la Figura 5.13 una comparación entre los valores reales que toma la congestión y los valores que predice el modelo. Los valores registrados en las trazas parecen tener cierto carácter discreto, ya que se producen cambios bruscos en el valor que el modelo no es capaz de predecir con mucha precisión.

A pesar de este hecho, tanto esta figura como los valores de MSE demuestran un alto grado de similitud entre ambas secuencias

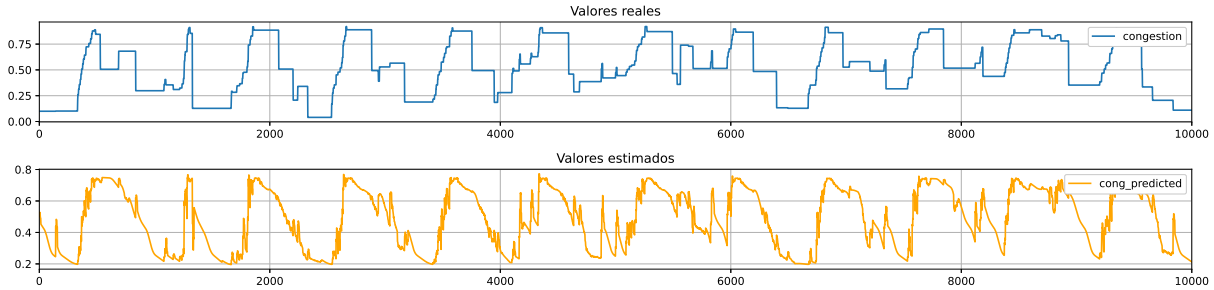


Figura 5.13: Comparativa valores reales y predicción del modelo basado en RNN

Por último, para intentar comparar el rendimiento de este modelo con los anteriores se ha discretizado la salida predicha del modelo y se ha comparado con los niveles discretos asignados. De esta manera, en la Figura 5.14 se puede calcular la misma métrica usada en los casos anteriores.

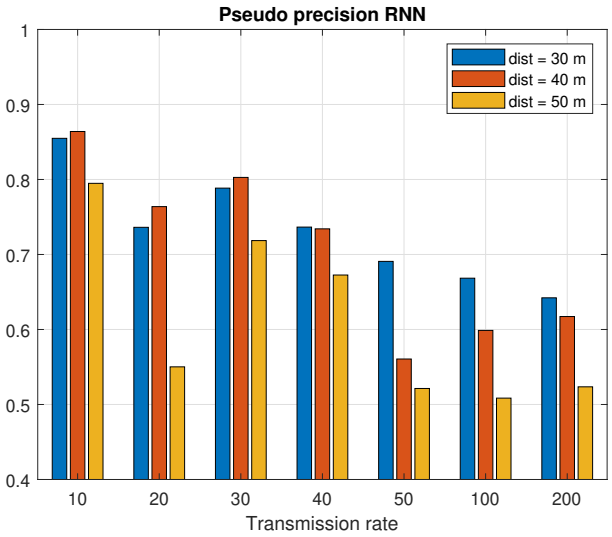


Figura 5.14: Rendimiento del modelo basado en redes neuronales

A pesar de tener un rendimiento bueno en la tarea de predicción al discretizar estos valores para poder compararlos con los obtenidos con las otras soluciones propuestas se aprecia cómo el rendimiento de la solución basada en redes neuronales es inferior al que proporcionan otras soluciones más sencillas.

## 5.4. Comparativa del rendimiento de los modelos

Con objeto de resumir todos los resultados y gráficas presentadas a lo largo de este capítulo la Figura 5.15 muestra el rendimiento de las diferentes soluciones propuestas en lo que a términos de rendimiento se refiere, utilizando la métrica indicada anteriormente para un escenario con distancia igual a 40 m.

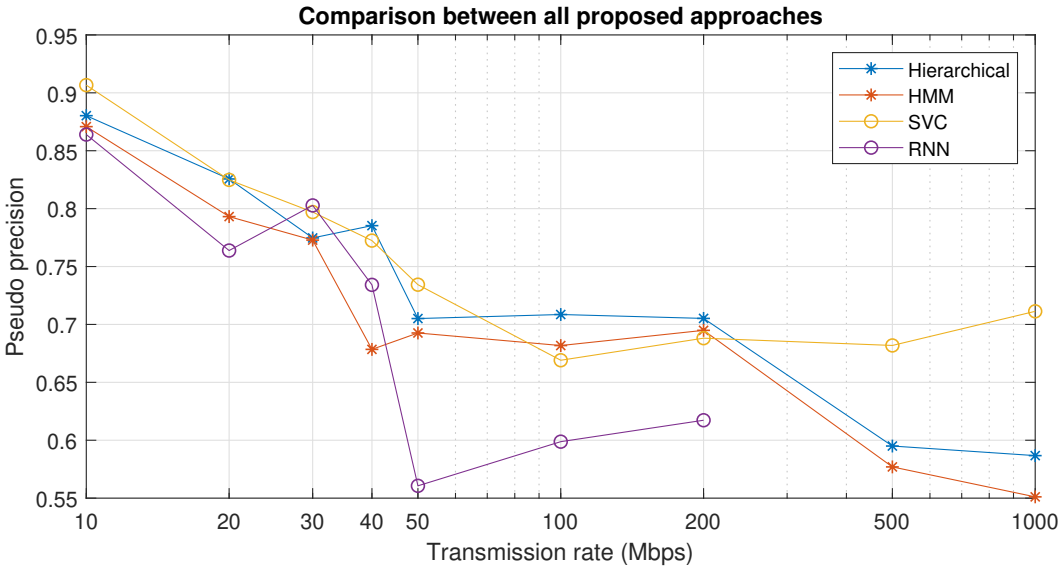


Figura 5.15: Rendimiento de las diferentes soluciones propuestas

En primer lugar, el hecho más notable es que la red neuronal, a pesar de tener unos valores de MSE buenos, utilizando la métrica descrita en este trabajo, presenta un rendimiento inferior al del resto de soluciones propuestas. Este hecho se hace especialmente notable a medida que aumentamos la tasa de transmisión.

Por otra parte, se aprecia que la solución jerárquica ofrece el mejor rendimiento entre las soluciones de aprendizaje no supervisado, aunque no existe gran diferencia entre esta y el modelo de cadenas ocultas de Markov.

Ambos modelos tienen un comportamiento similar con el algoritmo supervisado SVC excepto para tasas de transmisión altas, donde este último presenta el mejor rendimiento entre todos los comentados.

# 6

# Conclusiones y líneas futuras

En este capítulo se resume y analiza todo lo expuesto en los apartados anteriores, poniendo especial atención a los resultados obtenidos en el Capítulo 5. Por otra parte, se hará una pequeña estimación del posible uso futuro de estas técnicas y su extensión.

## 6.1. Conclusiones

El sector de la telefonía móvil va a sufrir un cambio nunca visto hasta ahora con la adopción de las redes 5G. Este hecho va a ser especialmente notable en las redes de alta frecuencia o de mmW.

Las soluciones tradicionales, basadas principalmente en TCP, han demostrado ser ineficientes en entornos altamente variables como puede ser el entorno inalámbrico. Este hecho se agrava a medida que se avanza hacia entornos más variantes.

Es por eso por lo que hay numerosos trabajos que tratan de encontrar una solución más adecuada para entornos inalámbricos ([8, 9, 23]). Sin embargo, pocos estudios se centran en cómo las redes mmW afectan a los algoritmos de control de congestión.

En este trabajo se ha hecho un análisis del uso potencial de técnicas más innovadoras, como son las soluciones basadas en *Machine Learning*, en escenarios donde las soluciones tradicionales no ofrecen un rendimiento adecuado.

Se han analizado algunos de los datos disponibles por los nodos a nivel de capa de transporte para conocer su utilidad a la hora de predecir la congestión.

Seguidamente, se ha comprobado cómo algunas de las métricas capturadas están altamente relacionadas con los valores a predecir.

Con estas métricas se ha realizado el tratamiento de los datos adecuado para poder alimentar los diferentes algoritmos utilizados. Se han propuesto diferentes soluciones para modelar la congestión analizando los resultados obtenidos en cada una de ellas.

Se aprecia en los resultados como soluciones simples, basadas en aprendizaje no supervisado ofrecen rendimientos altos para predicción binaria.

Para casos más complejos como predicción multinivel, o predicción del valor de congestión, se han estudiado técnicas más elaboradas. De entre los diferentes algoritmos analizados se ha visto que la solución jerárquica basada en el algoritmo *Kmeans* es la que mejor rendimiento ofrece. Además, esta solución tiene un coste computacional relativamente bajo, comparado con el resto de algoritmos analizados.

Por otra parte, la solución más compleja basada en redes neuronales ha funcionado bien para escenarios donde la tasa de transmisión es baja. Sin embargo, a medida que han empeorado las condiciones del canal su rendimiento ha decrecido en mayor medida que para el resto de las soluciones.

Es interesante remarcar el rendimiento obtenido utilizando una red neuronal pre-entrenada sobre escenarios similares al de entrenamiento, ya que puede ser un primer paso hacia una solución global basada en un modelo pre-entrenado.

Por último, este trabajo ha contribuido en parte a la elaboración del artículo titulado **"Learning Congestion over Millimeter-Wave Channels"** que se encuentra bajo revisión en la conferencia internacional *IEEE WiMob 2020*.

## 6.2. Líneas futuras

Tras ver el posible valor que tiene este tipo de técnicas sobre este tipo de escenarios se han identificado una sería de aspectos que se podrían abordar en el futuro.

El uso de estas técnicas sobre escenarios más complejos, los cuales se adecúen mejor a un entorno real, con topologías MIMO, múltiples usuarios o usuarios en movimiento, es posiblemente el siguiente paso por seguir en el estudio de estas técnicas.

Por otra parte, al tratarse de una solución intensiva utilizada por usuarios móviles, debe caracterizarse el consumo de energía que estas soluciones implican en los terminales finales.

Con los resultados obtenidos con el uso de RNN pueden existir arquitecturas más complejas que se adecúen a cualquier tipo de escenario y que puedan ser distribuidas parcial o totalmente pre-entrenada. Esto solucionaría el mayor inconveniente de esta técnica que reside en el elevado coste de entrenamiento.

Estas soluciones además pueden adaptarse no solo para predecir el nivel de congestión actual sino para predecir múltiples instantes temporales futuros.

En la rama de aprendizaje no supervisado, visto el rendimiento de modelos sencillos como el *Kmeans* utilizado de manera jerárquica, se pueden utilizar soluciones similares que tengan en cuenta más aspectos como es el caso de Gaussian Mixture Models (GMM) que tienen en cuenta la covarianza de las métricas entre sí.

# Bibliografía


[1] Ali A Zaidi, R Baldemair, M Andersson, S Faxér, V Molés-Cases, and Z Wang. Designing for the future: the 5g nr physical layer. *Ericsson Technology Review*, pages 1–13, 2017.

[2] Liu and Bahadori. A survey on granger causality, a computational view. 2004.

[3] GSMA. 5g spectrum gsma public policy position. *Information Theory, IEEE Transactions on*, 2020.

[4] Yong Niu, Yong Li, Depeng Jin, Li Su, and Athanasios V Vasilakos. A survey of millimeter wave communications (mmwave) for 5g: opportunities and challenges. *Wireless networks*, 21(8):2657–2676, 2015.

[5] 3GPP. NR; Medium Access Control (MAC) protocol specification. Technical Specification (TS) 38.321, 3rd Generation Partnership Project (3GPP), 2017.

[6] 3GPP. NR; Radio Link Control (RLC) protocol specification. Technical Specification (TS) 38.322, 3rd Generation Partnership Project (3GPP), 2017.

[7] 3GPP. NR; Packet Data Convergence Protocol (PDCP) specification. Technical Specification (TS) 38.323, 3rd Generation Partnership Project (3GPP), 2017.

[8] Yasir Zaki, Thomas Pötsch, Jay Chen, Lakshminarayanan Subramanian, and Carmelita Görg. Adaptive congestion control for unpredictable cellular networks. In *Proceedings of the 2015 ACM Conference on Special Interest Group on Data Communication*, pages 509–522, 2015.

[9] Keith Winstein, Anirudh Sivaraman, and Hari Balakrishnan. Stochastic forecasts achieve high throughput and low delay over cellular networks. In *Presented as part of the 10th {USENIX} Symposium on Networked Systems Design and Implementation ({NSDI} 13)*, pages 459–471, 2013.

[10] Neal Cardwell, Yuchung Cheng, C Stephen Gunn, Soheil Hassas Yeganeh, and Van Jacobson. Bbr: Congestion-based congestion control. *Queue*, 14(5):20–53, 2016.

[11] Kadangode Ramakrishnan, Sally Floyd, David Black, et al. The addition of explicit congestion notification (ecn) to ip. 2001.

[12] Mohammad Alizadeh, Albert Greenberg, David A Maltz, Jitendra Padhye, Parveen Patel, Balaji Prabhakar, Sudipta Sengupta, and Murari Sridharan. Data center tcp (dctcp). In *Proceedings of the ACM SIGCOMM 2010 conference*, pages 63–74, 2010.

[13] Stephen Bensley, Lars Eggert, Dave Thaler, Praveen Balasubramanian, and Glenn Judd. Datacenter tcp (dctcp): Tcp congestion control for datacenters. *Internet Draft*, 2017.

[14] Brian Trammell, Mirja Kühlewind, Damiano Boppart, Iain Learmonth, Gorry Fairhurst, and Richard Scheffenegger. Enabling internet-wide deployment of explicit congestion notification. In *International Conference on Passive and Active Network Measurement*, pages 193–205. Springer, 2015.

[15] Ravi Netravali, Anirudh Sivaraman, Somak Das, Ameesh Goyal, Keith Winstein, James Mickens, and Hari Balakrishnan. Mahimahi: Accurate record-and-replay for {HTTP}. In *2015 {USENIX} Annual Technical Conference ({USENIX}{ATC} 15)*, pages 417–429, 2015.

[16] 3GPP. Study on channel model for frequency spectrum above 6 GHz. Technical Report (TR) 38.900, 3rd Generation Partnership Project (3GPP), 2017.

[17] Leandros Tassiulas and Anthony Ephremides. Stability properties of constrained queueing systems and scheduling policies for maximum throughput in multihop radio networks. In *29th IEEE Conference on Decision and Control*, pages 2130–2132. IEEE, 1990.

[18] David A Dickey and Wayne A Fuller. Distribution of the estimators for autoregressive time series with a unit root. *Journal of the American statistical association*, 74(366a):427–431, 1979.

[19] Heino Bohn Nielsen. Non-stationary time series and unit root tests. *Unpublished Lecture Notes for Econometrics*, 2, 2005.

[20] Clive WJ Granger. Investigating causal relations by econometric models and cross-spectral methods. *Econometrica: journal of the Econometric Society*, pages 424–438, 1969.

[21] Lawrence R Rabiner. A tutorial on hidden markov models and selected applications in speech recognition. *Proceedings of the IEEE*, 77(2):257–286, 1989.

[22] Thad Starner and Alex Pentland. Real-time american sign language recognition from video using hidden markov models. In *Motion-based recognition*, pages 227–243. Springer, 1997.

[23] Mo Dong, Tong Meng, Doron Zarchy, Engin Arslan, Yossi Gilad, Brighten Godfrey, and Michael Schapira. {PCC} vivace: Online-learning congestion control. In *15th {USENIX} Symposium on Networked Systems Design and Implementation ({NSDI} 18)*, pages 343–356, 2018.

# Lista de acrónimos

**ECN** Explicit Congestion Notification

**RED** Random Early Detection

**BDP** Bandwidth Delay Product

**DCTCP** Data Centers TCP

**RTT** Round Trip Time

**IaT** Inter Arrival Time

**NR** New Radio

**TDT** Television Digital Terrestre

**IoT** Internet of Things

**CP-OFDM** Cyclic Prefix OFDM

**UL** Up link

**DL** Down link

**D2D** Device to device

**eMMB** enhanced Mobile Broadband

**URLLC** Ultra Reliable Low Latency Communications

**mMTC** massive Machine Type Communications

**RLC** Radio Link Control

**AM** Acknowledged mode

**UM** Unacknowledged mode

**TM** Transparent mode

**SDU** Service Data Unit

**HARQ** Hibrid Automatic Repeat Request

**PDCP** Packet Data Convergence Protocol

**VoD** Video on Demand

**LOS** Line of Sight

**UDP** User Datagram Protocol

**TCP** Transport Control Protocol

**MCS** Modulation and Coding Scheme

**NG** New Generation

**MSS** Maximum Segment Size

**RTO** Retransmission TimeOut

**BDP** Bandwidth Delay Product

**ML** Machine Learning

**SINR** Signal Interference Noise Ratio

**PCA** Principal Component Analysis

**SVM** Support Vector Machine

**SVC** Support Vector Clasiffication

**RNN** Redes neuronales recurrentes

**WCSS** Within-cluster Sum of Squares

**MIMO** Multiple Input Multiple Output

**LSTM** Long Short Term Memory

**MSE** Error Cuadrático Medio

**GMM** Gaussian Mixture Models

**HMM** Hidden Markov Model

# Métricas clasificación binaria

| | 30 | | | 40 | | | 50 | | |
|---|---|---|---|---|---|---|---|---|---|
| | Prec. | Recall | F1-Score | Prec. | Recall | F1-Score | Prec. | Recall | F1-Score |
| 10 | 0.926 | 0.907 | 0.916 | 0.976 | 0.824 | 0.894 | 0.983 | 0.840 | 0.906 |
| 20 | 0.682 | 0.991 | 0.808 | 0.848 | 0.901 | 0.874 | 0.957 | 0.840 | 0.895 |
| 30 | 0.637 | 0.986 | 0.774 | 0.733 | 0.841 | 0.783 | 0.809 | 0.875 | 0.841 |
| 40 | 0.657 | 0.984 | 0.788 | 0.639 | 0.955 | 0.765 | 0.700 | 0.856 | 0.770 |
| 50 | 0.609 | 0.896 | 0.725 | 0.791 | 0.704 | 0.745 | 0.693 | 0.835 | 0.757 |
| 100 | 0.621 | 0.757 | 0.682 | 0.550 | 0.670 | 0.604 | 0.341 | 0.759 | 0.471 |
| 200 | 0.743 | 0.548 | 0.631 | 0.632 | 0.659 | 0.646 | 0.618 | 0.660 | 0.638 |
| 500 | 0.703 | 0.422 | 0.528 | 0.654 | 0.595 | 0.623 | 0.672 | 0.515 | 0.583 |
| 1000 | 0.652 | 0.388 | 0.487 | 0.646 | 0.624 | 0.635 | 0.622 | 0.490 | 0.548 |

Métricas rendimiento Kmeans binario

| | 30 | | | 40 | | | 50 | | |
|---|---|---|---|---|---|---|---|---|---|
| | Prec. | Recall | F1-Score | Prec. | Recall | F1-Score | Prec. | Recall | F1-Score |
| 10 | 0.922 | 0.945 | 0.933 | 0.938 | 0.962 | 0.950 | 0.942 | 0.956 | 0.949 |
| 20 | 0.775 | 0.916 | 0.840 | 0.857 | 0.898 | 0.877 | 0.915 | 0.939 | 0.927 |
| 30 | 0.751 | 0.920 | 0.827 | 0.783 | 0.839 | 0.810 | 0.802 | 0.912 | 0.853 |
| 40 | 0.748 | 0.926 | 0.827 | 0.688 | 0.864 | 0.766 | 0.700 | 0.861 | 0.772 |
| 50 | 0.689 | 0.779 | 0.731 | 0.759 | 0.937 | 0.839 | 0.708 | 0.837 | 0.767 |
| 100 | 0.681 | 0.656 | 0.668 | 0.688 | 0.491 | 0.573 | 0.894 | 0.016 | 0.031 |
| 200 | 0.755 | 0.751 | 0.753 | 0.717 | 0.684 | 0.700 | 0.711 | 0.651 | 0.680 |
| 500 | 0.871 | 0.740 | 0.800 | 0.800 | 0.648 | 0.716 | 0.717 | 0.775 | 0.745 |
| 1000 | 0.848 | 0.660 | 0.742 | 0.737 | 0.826 | 0.779 | 0.724 | 0.745 | 0.734 |

Métricas rendimiento SVC binario

# Métricas clasificación multinivel (N=4)

| | 30 | 40 | 50 |
|---|---|---|---|
| 10 | 0.871 | 0.880 | 0.867 |
| 20 | 0.813 | 0.826 | 0.863 |
| 30 | 0.777 | 0.775 | 0.807 |
| 40 | 0.781 | 0.785 | 0.765 |
| 50 | 0.745 | 0.705 | 0.781 |
| 100 | 0.734 | 0.709 | 0.742 |
| 200 | 0.655 | 0.705 | 0.678 |
| 500 | 0.604 | 0.595 | 0.594 |
| 1000 | 0.572 | 0.587 | 0.580 |

Rendimiento Kmeans jerárquico

| | 30 | 40 | 50 |
|---|---|---|---|
| 10 | 0.837 | 0.871 | 0.785 |
| 20 | 0.768 | 0.793 | 0.834 |
| 30 | 0.751 | 0.773 | 0.767 |
| 40 | 0.745 | 0.678 | 0.740 |
| 50 | 0.616 | 0.693 | 0.777 |
| 100 | 0.659 | 0.682 | 0.708 |
| 200 | 0.655 | 0.695 | 0.684 |
| 500 | 0.503 | 0.577 | 0.586 |
| 1000 | 0.473 | 0.551 | 0.553 |

Rendimiento HMM

| | 30 | 40 | 50 |
|---|---|---|---|
| 10 | 0.892 | 0.907 | 0.881 |
| 20 | 0.840 | 0.825 | 0.862 |
| 30 | 0.819 | 0.797 | 0.743 |
| 40 | 0.768 | 0.772 | 0.729 |
| 50 | 0.754 | 0.734 | 0.763 |
| 100 | 0.746 | 0.669 | 0.649 |
| 200 | 0.729 | 0.688 | 0.666 |
| 500 | 0.743 | 0.682 | 0.697 |
| 1000 | 0.735 | 0.711 | 0.692 |

Rendimiento SVC

| | 30 | 40 | 50 |
|---|---|---|---|
| 10 | 0.855 | 0.864 | 0.795 |
| 20 | 0.736 | 0.764 | 0.550 |
| 30 | 0.788 | 0.803 | 0.719 |
| 40 | 0.737 | 0.734 | 0.673 |
| 50 | 0.691 | 0.561 | 0.521 |
| 100 | 0.668 | 0.599 | 0.508 |
| 200 | 0.642 | 0.617 | 0.524 |
| 500 | | | |
| 1000 | | | |

Rendimiento RNN