

# Índice general

<b>Resumen</b>	<b>1</b>
<b>1. Introducción</b>	<b>2</b>
1.1. Estado del arte . . . . .	2
1.2. Objetivos y metodología . . . . .	3
<b>2. Descripción y exploración de los datos</b>	<b>5</b>
2.1. Series de precipitación . . . . .	5
2.2. Series de temperatura . . . . .	6
<b>3. Modelo Hidrológico</b>	<b>8</b>
3.1. Modelo LEM . . . . .	8
3.1.1. Modelo MELCA . . . . .	10
3.2. Modelo MODSIM . . . . .	10
<b>4. Modelos de aprendizaje profundo</b>	<b>14</b>
4.1. Introducción teórica . . . . .	14
4.2. Redes recurrentes con memoria a largo plazo . . . . .	16
4.3. Topología de los modelos utilizados . . . . .	18
4.3.1. Regularización de la función de coste . . . . .	19
4.3.2. Funciones de activación . . . . .	20
4.3.3. Optimizadores . . . . .	21
4.3.4. Calibración de los modelos . . . . .	21
4.4. Entrenamiento . . . . .	22
4.4.1. Entrenamiento global . . . . .	23
4.4.2. Entrenamiento local . . . . .	24
4.4.3. Entrenamiento secuencial . . . . .	25
4.5. Análisis de componentes principales . . . . .	25
4.6. Coeficiente de Nash-Sutcliffe . . . . .	25
<b>5. Resultados</b>	<b>27</b>
5.1. Validación de los modelos . . . . .	27
5.2. Predicción de los caudales naturales . . . . .	32
5.3. Cálculo de balance hídrico con MODSIM . . . . .	33
<b>6. Discusiones y Conclusiones</b>	<b>37</b>
<b>7. Anexo</b>	<b>39</b>

# Índice de figuras

2.1. Series de precipitación para diferentes regiones de la cuen- ca Chambo. En los paneles superiores se muestra la serie de precipitación correspondiente al año 2019 y los valores me- dios mensuales desde el año 2000 hasta el año 2020. En el panel inferior, se muestran las series de temperatura máxima y mínima diarias. . . . .	7
3.1. Caudales diarios simulados con el modelo MELCA para las cuencas con ids 8, 57 y 75 desde el año 2000 hasta el año 2020.	11
3.2. Esquema conceptual de CHRC con MODSIM . . . . .	13
4.1. Representación gráfica de una neurona artificial (figura toma- da de [1]). . . . .	14
4.2. Tres topologías distintas de red neuronal: (a) multicapa, (b) competitivas, (c) recurrentes. . . . .	15
4.3. Back propagation . . . . .	16
4.4. Red neuronal recurrente con memoria a largo y corto plazo (LSTM). Figura tomada de [2]. . . . .	17
4.5. Recorrido por el funcionamiento de una celda LSTM. Figuras tomadas de [2]. . . . .	18
4.6. Modelo Denso. . . . .	19
4.7. Modelo LSTM1. . . . .	19
4.8. Modelo LSTM2. . . . .	20
4.9. Evolución de la función de loss considerando diferentes opti- mizadores. . . . .	21
4.10. k-fold cross validation (imagen de Gufosowa - WikiMedia).	22
4.11. Porcentaje de varianza explicada acumulada para las prime- ras 9 componentes principales. . . . .	26
5.1. Valores obtenidos para el coeficiente NSE a lo largo de toda la cuenca con los modelos denso y LSTM1 entrenado con PCAs.	28
5.2. Evolución de la función de loss en los conjuntos de entrena- miento y validación para los modelos denso y LSTM1 con PCAs. . . . .	28
5.3. Evolución del coeficiente NSE en función del área (panel su- perior) y la variación de precipitación (panel inferior) para los modelos LSTM1 con y sin PCAs. . . . .	29
5.4. Evolución del coeficiente NSE en función del área (panel su- perior) y la variación de precipitación (panel inferior) para los modelos LSTM1 con PCAs y LSTM1 entrenado localmente.	30

5.5. Función de distribución acumulada para todos los modelos de ANNs a lo largo de todas las sub-cuenca de CHRC. . . . .	31
5.6. Predicciones del caudal de descarga obtenidos con los modelos Denso, LSTM1 con PCAs y LSTM1 local en la subcuenca con id 20. En los paneles inferiores se muestran los grados de ajuste correspondientes. . . . .	32
5.7. Predicciones del caudal de descarga obtenidos con los modelos Denso, LSTM1 con PCAs y LSTM1 local en la subcuenca con id 67. En los paneles inferiores se muestran los grados de ajuste correspondientes. . . . .	34
5.8. Resultados del Modelo LSTM2. En el panel superior se muestran los valores de NSE a lo largo de toda la cuenca y En el panel inferior, los valores predichos en la sub-cuenca con id 15.	35
5.9. Paneles superiores: residuos obtenidos a lo largo de diferentes sub-cuenca. Paneles inferiores: resultados finales obtenidos utilizando el software de gestión de recursos MODSIM, teniendo en cuenta los usos del agua. . . . .	36
7.1. Base de datos relacional con los descriptores de las componentes de una cuenca hidrográfica. . . . .	41
7.2. Ejemplo de consulta de uno de los métodos de la API que contiene la base de datos. . . . .	42
7.3. Ejemplo de consulta del método que calcula el balance hídrico de la cuenca. . . . .	43

# Resumen

En este trabajo se ha evaluado el desempeño de diferentes modelos de redes neuronales para predecir el caudal natural en la cuenca hidrográfica Chambo a partir de series de entrada hidro-climáticas que abarcan un rango de 14 años. Se han considerado redes con diferentes estructuras y se han entrenado de manera local, global y secuencial. Se ha medido el desempeño de los modelos utilizando diferentes métricas y se ha comprobado que en la mayoría de los casos los modelos logran reconstruir de manera correcta los caudales en un rango de 6 años. Con los resultados obtenidos se ha utilizado el software de gestión MODSIM que permite optimizar los usos de los recursos de la cuenca y se ha comprobado que el balance hídrico obtenido con los caudales predichos por los modelos de redes neuronales es el mismo que el obtenido con un modelo hidrológico tradicional. Finalmente se ha desarrollado una aplicación que permite la ejecución del modelo en diferente cuencas hidrográficas.

**Palabras claves:** Redes Neuronales, redes recurrentes, hidrología, predicción, cuencas hidrográficas, aprendizaje automático, aprendizaje profundo

## Abstract

In this work, the performance of different neural network models to predict the natural flow in the watershed Chambo has been evaluated. Models with different configurations have been trained locally, globally and sequentially in hydro-climatic series spanning a 14-year range. The performance of the models, has been measured using different metrics, and it has been verified that in most cases, they succeed in correctly reconstructing the flows over a 6 year range. The obtained results has been used to optimize the use of the resources of the basin with the software MODSIM, and it has been proven that the results obtained by means of neural network models are the same as those obtained with a traditional hydrological model. Finally, an application has been developed that allows the execution of the model in different hydrological basins.

**Key words:** Neural Networks, recurrent networks, hydrology, prediction, hydrographic basin, machine learning, deep learning

# Introducción

## 1.1. Estado del arte

Un problema de larga data en la hidrología a gran escala es el hecho de que los modelos tradicionales, ya sean agregados o basados en procesos físicos, en general no son extrapolables en el espacio, es decir, un modelo calibrado en una cuenca dada no es válido para otra. El desafío principal que deben sortear estos modelos es el de aprender a codificar las diferencias en las características locales para poder extraer información hidrológica de un región a otra, por ejemplo de cuencas calibradas a cuencas sin calibrar [3] [4] [5] [6].

Una pregunta a responder es si realmente existe una teoría hidrológica válida a macro escalas que englobe la cantidad de procesos heterogéneos que ocurren en una cuenca hidrográfica [7]. Más aun, en el artículo *Twenty-three Unsolved Problems in Hydrology (UPH)- a community perspective* publicado en el año 2019 [8] se ha incluido la pregunta de “*¿Cuáles son las leyes hidrológicas válidas en las escalas de una cuenca hidrográfica?*” entre uno de los 23 problemas sin resolver en la hidrología. La respuesta a esta pregunta probablemente venga de la mano los avances de las técnicas de aprendizaje automático y aprendizaje profundo (ML y DP por sus siglas en inglés) que indican que es posible derivar un modelo global a partir de los datos observacionales disponibles que funcione en cuencas hidrológicas a gran escala[7].

Éstos modelos derivados de los datos, que carecen de un modelo físico conceptual asociado, no se basan en el conocimiento previo del sistema hidrológico, sino que aprenden los diferentes comportamientos de las cuencas directamente de las entradas meteorológicas y funcionan mejor cuando se entrena en una gran cantidad de datos proveniente de diferentes cuencas [9][10]. Por el contrario, los modelos tradicionales funcionan mejor cuando se calibran individualmente en cada cuenca y los resultados obtenidos no son transferibles a otras regiones.

Estudios recientes han demostrado que los modelos de DL entrenados en cuencas calibradas predicen mejor en cuencas no calibradas que los modelos tradicionales. Kratzert et al. [11] demostraron que las redes neuronales que incluyen celdas con memoria a largo plazo (LSTM por sus siglas en inglés) superan la performance del bien conocido modelo de sacramento (SAC-SMA) [12] [13] en una serie de cuencas para las cuales el modelo SAC-SMA había sido calibrado de manera individual. El modelo LSTM fue capaz de realizar mejores predicciones del caudal

en dichas cuencas siendo que este nunca antes había visto los datos de entrenamiento de estas cuencas.

Un estudio más reciente[7][14], en donde se han utilizado datos provenientes de varios cientos de cuencas continentales en los Estados Unidos a lo largo de 30 años, demuestra que los modelos de aprendizaje profundo arrojan mejores predicciones para los caudales de descarga en cuencas no aforadas que los obtenidos por modelos tradicionales calibrados con registros de datos en cuencas aforadas. Más aun, estos estudios han demostrado que las redes LSTM son incluso capaces de modelar la presencia de nieve y almacenar esta información en las celdas específicas de memoria sin siquiera haber sido entrenadas directamente en ningún tipo de observación relacionada con nieve más allá de la precipitación total y la temperatura [11]. A partir de estos resultados, se puede presuponer que este enfoque de desarrollar un modelado global es prometedor.

## 1.2. Objetivos y metodología

En este trabajo se han desarrollado modelos de redes neuronales (ANNs) capaces de aprender las diferentes características presentes a lo largo de las 76 sub-cuencas que conforman la cuenca del río Chambbo (CHRC) directamente a partir de datos meteorológicos y predecir los caudales de descarga en cada una de ellas. El objetivo del trabajo es evaluar el desempeño de estos modelos en un entorno calibrado, es decir que no se le pedirá a los modelos que predigan en cuencas para las cuales nunca han visto sus datos de entrenamiento, y determinar si estos modelos, que utilizan datos meteorológicos, son capaces de combinar diferentes partes de la red para predecir los caudales en diferentes puntos a lo largo de toda la cuenca.

En la primera parte del trabajo, el propósito fue generar simulaciones hidrológicas para los caudales naturales de las cuencas que fueron utilizados como las variables objetivo para entrenar los modelos de ANNs. Con este fin se ha utilizado el modelo desarrollado por IHCantabria, denominado MELCA (Modelo de Equilibrio Logístico para Cuenca Andinas) descrito en el capítulo 3 que constituye una adaptación del modelo general denominado LEM (Modelo de Equilibrio Logístico) para cuencas de alta montaña con presencia de páramos y bofedales. Las series meteorológicas utilizadas para simular los caudales con el modelo hidrológico y entrenar los modelos de ANNs son descritas en el capítulo 2. Estas abarcan un período temporal de 20 años e incluyen series de precipitación, series de temperatura máxima y temperatura mínima.

Una vez generados los valores de los caudales simulados por el modelo hidrológico se procedió a entrenar tres configuraciones diferentes de modelos de ANNs descritos en el capítulo 4. Para esto, se han dividido los datos en conjuntos de entrenamiento y validación y se han re-estructurado para que puedan ser utilizados como inputs en las diferentes estructuras de redes consideradas. Para validar los modelos se ha utilizado además de la función de loss, incluida en los algoritmos de las ANNs, el coeficiente de Nash-Stutcliffe (NSE) que es ampliamente utilizado en hidrología. Con el fin de determinar la influencia que los restos o diferencias entre los caudales predichos por las ANNs y los caudales simulados con el modelo MELCA ejercen al determinar el balance hídrico en CHRC, se ha utilizado el software MODSIM que permite optimizar la distribución del agua sobre los cauces de la cuenca teniendo en cuenta los diferentes usos del agua.

Por último, y con el objetivo de poder ejecutar los modelos desarrollados de manera sencilla en otras cuencas hidrográficas, se ha creado una aplicación REST APIs utilizando la librería flask de python. Los detalles del proceso de desarrollo se describen en el anexo 7.

# Descripción y exploración de los datos

En este capítulo se describen las características principales de las series temporales hidro-climáticas utilizadas tanto para entrenar los modelos descritos en el capítulo 4, como para ejecutar el modelo hidrológico MELCA y así generar las variables objetivos, es decir los caudales de descarga de las subcuenca.

## 2.1. Series de precipitación

La cuenca CHRC posee una gran variación en la precipitación en un área geográfica reducida, por este motivo los valores de precipitación recogidos por pluviómetros en las estaciones meteorológicas no son fácilmente extrapolables. Por otro lado los pluviómetros se encuentran a altitudes que se encuentran entre los 2000 y 3000 metros sobre el nivel del mar, mientras que el punto más alto de la cuenca se encuentra a 6288 m.s.n.m, lo que da lugar a que haya una deficiencia de datos en ciertos puntos geográficos. Para completar los datos en estos puntos se ha recurrido a la base de datos de ERA5 [15] y CHELSA [16].

Con el fin de contrastar si las series así obtenidas reflejan correctamente las variaciones del patrón de lluvias con la altura y la pendiente, se han analizado diversas bases de datos globales de precipitación, pero el resultado no ha sido satisfactorio. Por ello, se ha optado por generar las series de manera sintéticas basándose en los patrones combinados de los pluviómetros y de las estaciones de aforos.

El modelo que genera las series de precipitación sintéticas diarias consta de dos niveles, primero se generan series mensuales y luego se generan series diarias desagregando los valores mensuales. Para generar las series mensuales se parte del valor medio anual (determinado mediante la interpolación de los datos de los pluviómetros con la base de datos CHELSA) en la región de interés y del valor medio en el mes más húmedo. El modelo de desagregación a escala mensual asume que las precipitaciones acumuladas en cada mes siguen un comportamiento que puede ser representado por la siguiente distribución "Log-normal" que posee una variación temporal sinusoidal y desviación estándar  $s_1$ :

$$P_m = \exp\left(N\left(a + b1 \cdot \cos\left(\frac{t - \phi_1}{6}\right) + b2 \cdot \cos\left(\frac{t - \phi_2}{12}\right)\right)\right) \quad (2.1)$$

$N(\mu, \sigma)$  es a su vez una distribución Gaussiana con media  $\mu$  y desviación estándar  $\sigma$ . Los valores de las constantes  $a$ ,  $b_1$  y  $b_2$  se obtienen a partir de la precipitación media y máxima, mientras que las fases  $\phi_i$  dependen del régimen de precipitación.

El modelo para crear las series diarias usa el método de cascadas aleatorias multiplicativas [17] para desagregar las series mensuales y posee dos parámetros: denominados  $\sigma_2$  y  $\beta$  que determinan la variabilidad e intermitencia de la lluvia (proporción media de días sin lluvia) y se usan para ajustar el modelo con los valores observados en las series de precipitación disponibles.

## 2.2. Series de temperatura

Para la generación de las series de temperatura se han utilizado datos de 10 termómetros situados en el entorno del área de estudio. Estos datos han sido sometidos a un proceso de curado que por un lado define una frontera para detectar outliers o datos atípicos siguiendo el siguiente criterio:

$$\text{si } X_i > 5 \cdot \sigma_n^2 \text{ es un outlier, donde } \sigma_n^2 = \frac{1}{n} \cdot \sum_{i=1}^n \left( X_i - \bar{X} \right)^2 \quad (2.2)$$

y por el otro lado elimina los datos que presentan un cierto grado de persistencia [18]. Los datos faltantes se han completado de manera similar a cómo se procedió con las series de precipitación utilizando los datos de ERA5.

En la figura 2.1, se muestran a modo de ejemplo las series de precipitación y temperaturas mínimas y máximas generadas para tres diferentes subcuencas localizadas en diferentes puntos de CHRC. Se pueden observar diferentes estacionalidades, la cuenca con id = 57 presenta un régimen costero (donde la precipitación máxima tiene lugar entre Marzo-Abril), mientras que las cuencas con ids= 8 y 75, un régimen amazónico (con un único pico de precipitación en Junio-Julio).

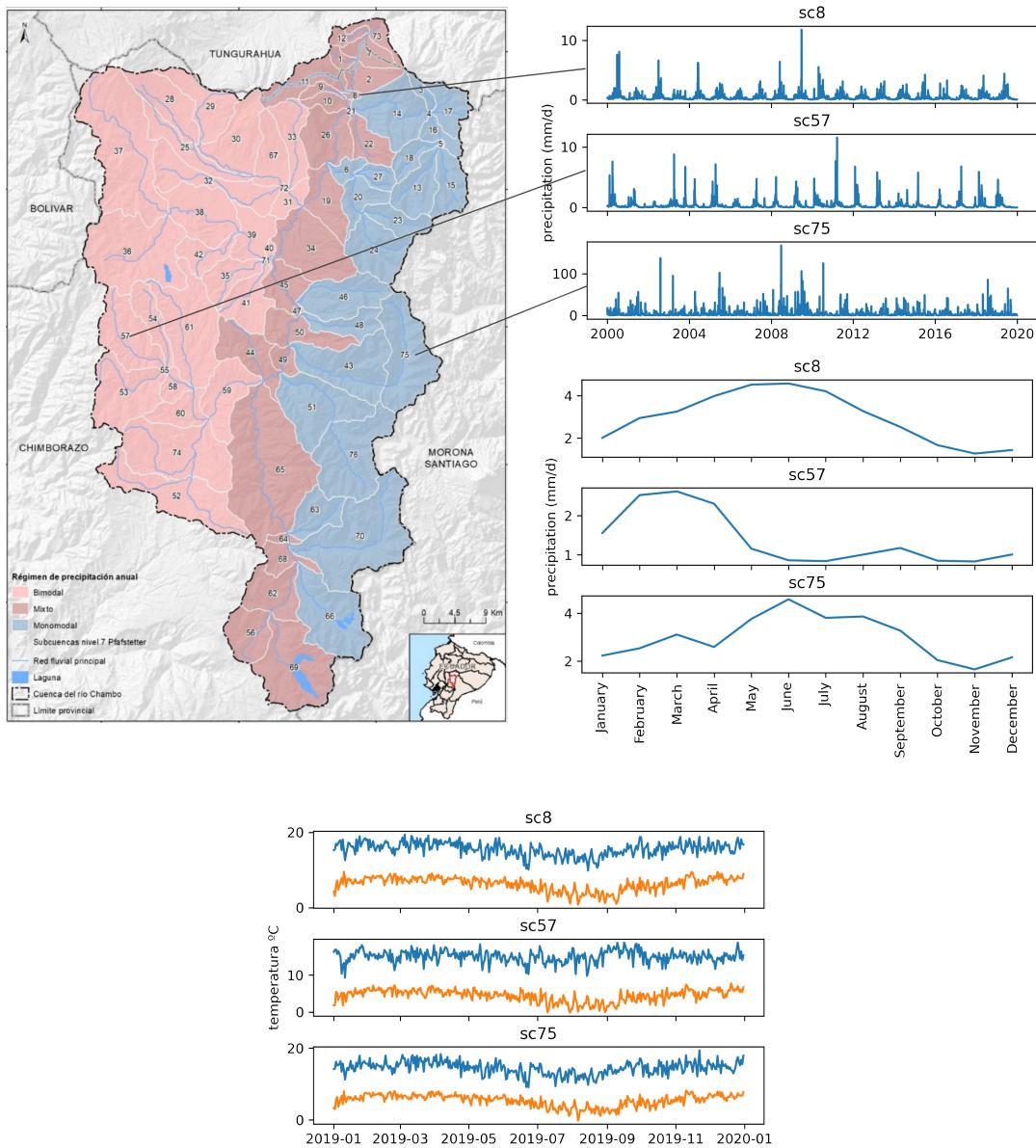


Figura 2.1: Series de precipitación para diferentes regiones de la cuenca Chambo. En los paneles superiores se muestra la serie de precipitación correspondiente al año 2019 y los valores medios mensuales desde el año 2000 hasta el año 2020. En el panel inferior, se muestran las series de temperatura máxima y mínima diarias.

# Modelo Hidrológico

Cómo se ha descrito anteriormente, se ha utilizado el modelo hidrológico MELCA para generar las series de caudales de descarga que han sido utilizadas como variables objetivos al entrenar los modelos de redes neuronales (capítulo 4). Este modelo está basado en LEM y fue desarrollado por IHCantabria. A continuación se describen las características principales de estos modelos.

## 3.1. Modelo LEM

Este modelo simula el proceso hidrológico en una cuenca hidrográfica a partir de las siguientes hipótesis empíricas:

1. Las cuencas hidrográficas son sistemas complejos que persiguen continuamente un equilibrio dinámico, dado por una combinación de factores climáticos ( precipitación y evapotranspiración potencial) y algunas características del terreno (topografía, vegetación, suelo, geología, etc.).

La evolución de la escorrentía ( $R$ ) hacia el equilibrio sigue la ley clásica de crecimiento descrito por la ecuación logística:

$$\frac{dR(t)}{dt} = K \cdot R(t) \cdot \left(1 - \frac{R(t)}{R_{eq}}\right) \quad (3.1)$$

2.  $R_{eq}$  es la escorrentía de equilibrio y se puede expresar como un coeficiente de escorrentía de equilibrio ( $C_{eq}$ ) multiplicado por la precipitación instantánea:  $R_{eq} = P \cdot C_{eq}$ .
3.  $K$  es la tasa de crecimiento y es una función lineal de la precipitación:  $K = P/S_0$ , donde  $S_0$  es una constante con unidades de longitud ( $mm$ ) que representa un espesor característico del suelo.
4. La ecuación logística no considera el tiempo de viaje desde las zonas de producción de escorrentía hasta el punto final de medida del caudal, en la salida de la cuenca. Cuando el intervalo de tiempo de análisis es del mismo orden de magnitud que el tiempo de respuesta de una cuenca, se debe agregar un método de propagación.

La versión estándar del LEM adopta un modelo lineal para el submodelo de enrutamiento y toma la forma del siguiente sistema de ecuaciones:

$$\frac{dR(t)}{dt} = \frac{P(t)}{S_0} \cdot R(t) \cdot \left(1 - \frac{R(t)}{R_{eq}}\right) \quad (3.2)$$

$$\frac{d\hat{P}}{dt} = P(t) - \frac{\hat{P}}{\lambda} \quad (3.3)$$

$$\frac{d\hat{E}}{dt} = E(t) - \frac{\hat{E}}{\lambda} \quad (3.4)$$

$$R_{eq}(t) = P(t) \cdot C_{eq}(\psi); \quad C_{eq}(\psi) = e^{-a \cdot \psi}, \psi = \frac{\hat{E}}{\hat{P}} \quad (3.5)$$

$$\frac{dQ(t)}{dt} = \frac{1}{\tau} \cdot [R(t) - Q(t)] \quad (3.6)$$

Donde  $R$  y  $Q$  son la escorrentía total y la descarga medida en la salida de la cuenca, respectivamente.  $P$  y  $E$  son la precipitación y la evapotranspiración potencial en cada paso de tiempo, mientras que  $\hat{P}$  y  $\hat{E}$  son valores promediados de  $P$  y  $E$  durante un periodo de tiempo característico, respectivamente. Los parámetros del modelo entonces son:

- $\lambda$  (días), el tiempo característico de respuesta de la cuenca.
- $S_0$  (mm), que representa un espesor medio de suelo o una capacidad de almacenamiento característica de la cuenca.
- $a$ , un parámetro adimensional que modifica la forma de la función de equilibrio (típicamente en el rango 0.5-1.5)
- $\tau$  (horas), el parámetro de enrutamiento, que puede considerarse un tiempo de respuesta rápido de la cuenca.

Este sistema de ecuaciones diferenciales ordinarias puede resolverse numéricamente con un esquema explícito incondicionalmente estable, ya que todas las ecuaciones, y en especial la logística, tienen solución analítica.

Como se ha mencionado en el capítulo 2, los datos de precipitación correspondientes a las zonas más altas de la cuenca son escasos. Es por esto que es necesario aplicar dos factores  $f_{cp}$  y  $f_{ce}$  que tienen en cuenta la influencia de la altitud de cada subcuenca y sirven para calibrar el modelo y corregir las series de precipitaciones y evo-transpiración,

respectivamente. Para calibrar el modelo se han utilizado datos de caudales recogidos por diferentes estaciones de aforo proporcionados por MAATE [19] y los valores de caudal promedio aportados por el documento de “*Aportes a la planificación para la gestión integral de los recursos hídricos*” citado en [20]. El modelo es entonces calibrado en cada sub-cuenca de manera individual, ajustando los factores de corrección  $f_{cp}$  y  $f_{ce}$  de modo tal que los caudales mensuales medios simulados se acerquen lo más posible a los valores medidos en las estaciones de aforo.

### 3.1.1. Modelo MELCA

MELCA es un modelo hidrológico semi-distribuido basado en LEM. Este modelo considera varias subcuencas, cada una de ellas con sus parámetros y forzamientos climáticos diferenciados y permite incluir una serie de particularidades asociadas a las cuencas tropicales andinas como la inclusión de páramos y bofedales con sus topologías y el estado de conservación. El modelo convierte la superficie de cada uno de estos ecosistemas andinos en una capacidad equivalente de almacenamiento del suelo. También incluye factores de corrección para la evo-transpiración en zonas de alta montaña, el efecto de glaciares y la aportación de agua atmosférica proveniente de niebla (flujo que es significativo en zonas cuencas andinas tropicales).

En la figura 3.2 se muestran a modo de ejemplo los caudales de descarga diarios simulados con el modelo MELCA para las cuencas con ids 8, 57 y 75. Se puede observar que la cuenca 75, que se encuentra más cerca del amazonas y con un régimen más húmedo, posee una caudal considerablemente mayor.

## 3.2. Modelo MODSIM

Una vez obtenidos los caudales para cada una de las cuencas, ya sea simulados por el modelo LEM o generados por los modelos de redes neuronales, se ha determinado el balance hidrológico de la cuenca mediante la utilización del software MODSIM [21], mundialmente aceptado y empleado para simular operaciones en sistemas hidrológicos como soporte de decisiones. Este modelo permite incorporar simultáneamente la complejidad física, hidrológica y las aspectos institucionales y administrativos del manejo de una cuenca, incluyendo los derechos del agua. Además, brinda soporte para la toma de decisión en el uso del agua entre la agricultura, uso poblacional, industrial, energético y ambiental.

MODSIM es un modelo compuesto por nodos y enlaces que representa el sistema que constituye una cuenca fluvial. Los objetos uti-

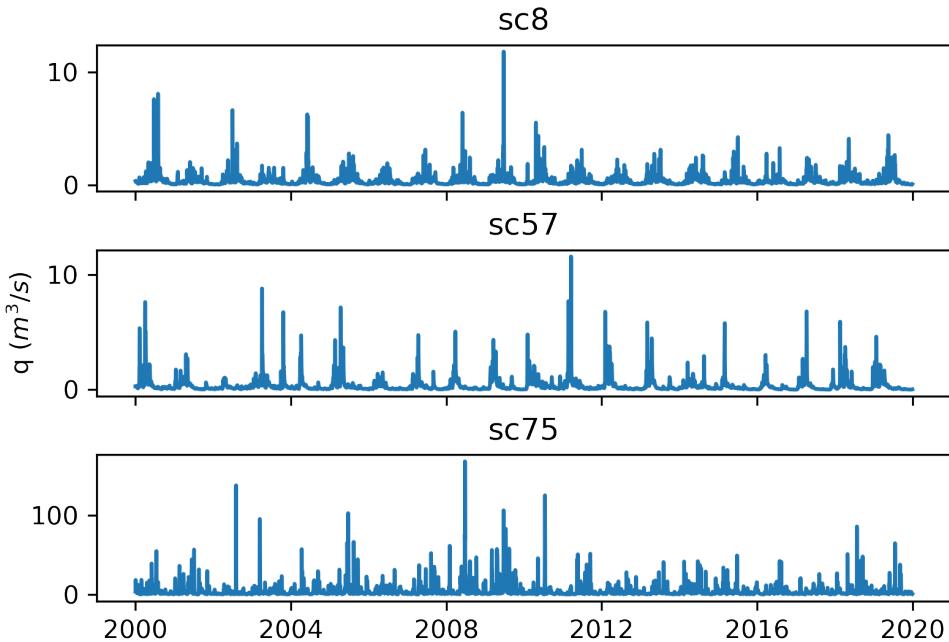


Figura 3.1: Caudales diarios simulados con el modelo MELCA para las cuencas con ids 8, 57 y 75 desde el año 2000 hasta el año 2020.

lizados en el modelo tienen asociados un costo, representado por un número de prioridad relativo. Los nodos con número de prioridad bajo tendrán menos costo y serán satisfechos antes que los nodos con una prioridad mayor. Los objetos en MODSIM no se limitan a representar características físicas e hidrológicas de una cuenca fluvial, sino que también se utilizan para simbolizar elementos artificiales y conceptuales para modelar mecanismos administrativos y legales complejos que rigen la asignación de agua.

Para optimizar la distribución de agua en las redes del cauce, MODSIM minimiza una función objetivo definida de la siguiente forma [?]:

$$\sum_{k \in A} c_k \cdot q_k \quad (3.7)$$

donde  $q_k$  es el caudal en el enlace  $k$ ,  $c_k$  son pesos o las prioridades de derecho de agua por unidad de caudal en el enlace  $k$ .

La minimización de esta función debe conservar el balance de masa, es decir,

$$\sum_{k \in O_i} q_k - \sum_{j \in I_i} q_j = b_{it}(q) \quad (3.8)$$

$$l_{kt} < q_k < u_{kt} \quad (3.9)$$

donde  $O_i/I_i$  es el conjunto de todos los enlaces que se originan/terminan en el nodo  $i$ , y  $b_{it}(q)$  es la ganancia o la pérdida en el nodo  $i$  en el tiempo  $t$ .  $l_{kt}$   $u_{kt}$  son los límites inferior y superior superior especificados, respectivamente, en el flujo en el enlace  $k$  en el tiempo  $t$ .

MODSIM es capaz tener en cuenta en el cálculo la evaporación, los flujos de retorno de las aguas subterráneas, las pérdidas en los canales y los requisitos de flujo en la corriente y resuelve las ecuaciones antes descriptas con el algoritmo de relajación Lagrangiana RELAX-IV [22].

En la figura 5.9, se muestra el esquema conceptual que representa a la cuenca CHRC. Los círculos son nodos sin almacenamiento que representan confluencias y desviaciones de los caudales de la cuenca, las flechas o links representan los caudales fluyentes entre nodos y los cuadrados las demandas consuntivas de agua. Se han considerado 4 tipos de demandas diferente: humana con prioridad 1, industrial, riego y acuicultura con prioridad 2 y energéticas con prioridad 3.

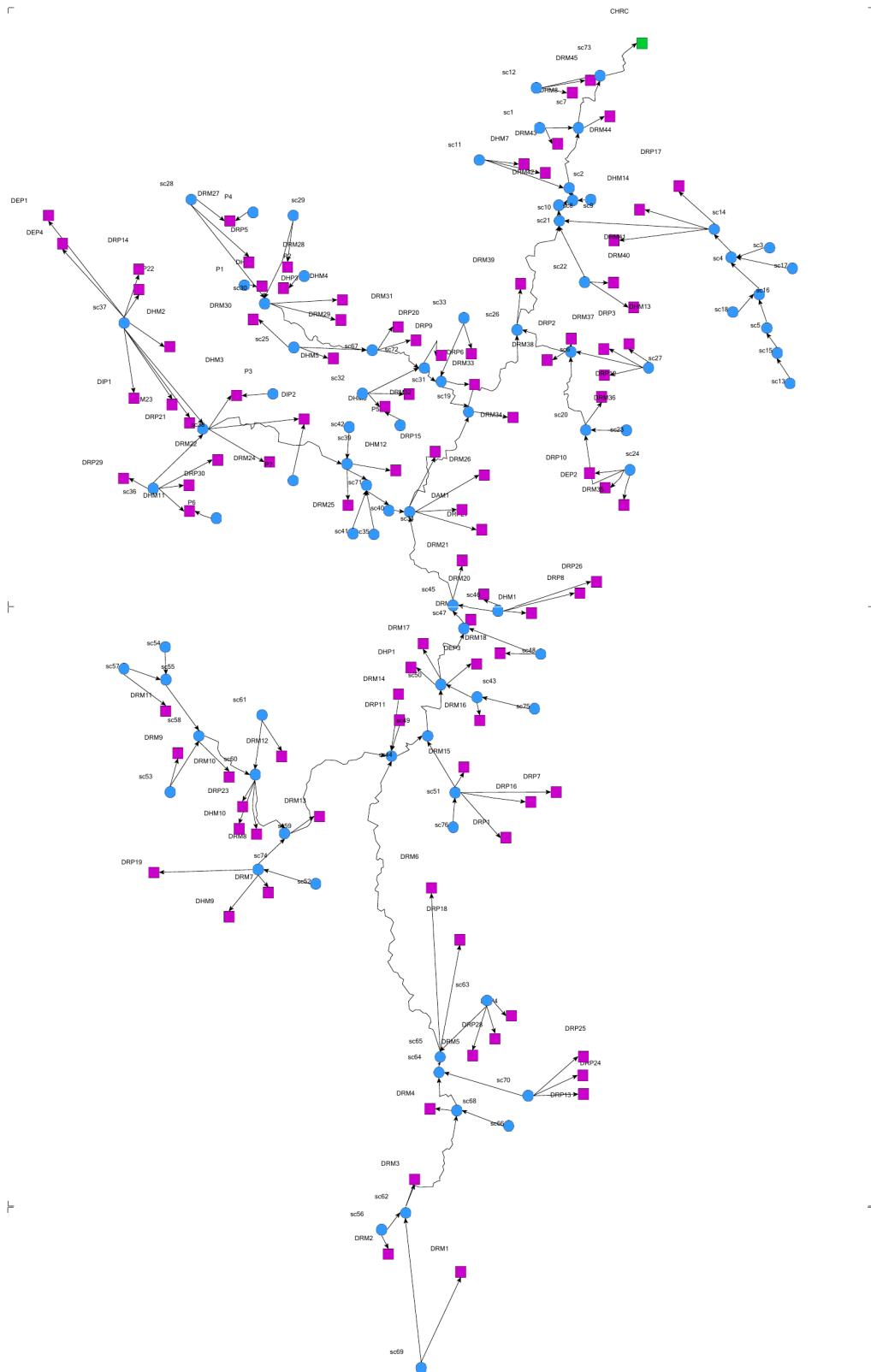


Figura 3.2: Esquema conceptual de CHRC con MODSIM

# Modelos de aprendizaje profundo

## 4.1. Introducción teórica

Las redes neuronales son modelos computacionales cuyo funcionamiento se inspira en el funcionamiento de las neuronas cerebrales reales, su función principal es recibir, procesar y transmitir información a través de señales químicas y eléctricas. En líneas generales, las neuronas se especializan en la recepción de estímulos y generación de impulsos entre ellas mediante conexiones llamadas sinapsis. Cuando una neurona recibe un impulso eléctrico que la activa, esta dispara un impulso que a su vez estimula a otras neuronas a las que está conectada. Algunas de estas neuronas a su vez se activan y se disparan activando a otras y así sucesivamente propagando los estímulos por toda la red. Entre más se practique una tarea, las conexiones se hacen más robustas, esto es el proceso de aprendizaje.

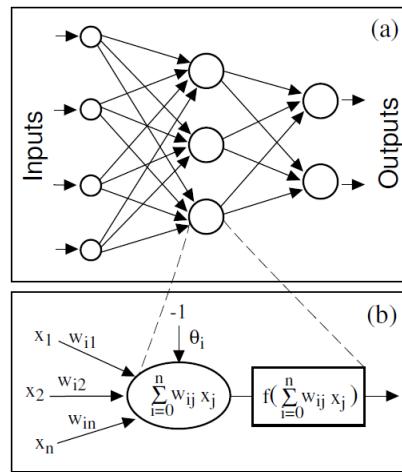


Figura 4.1: Representación gráfica de una neurona artificial (figura tomada de [1]).

Una neurona artificial (Fig. 4.1-b) funciona de una manera muy similar a las neuronas biológicas, ésta recibe datos de entrada que son combinados por la neurona asignando pesos a cada uno de ellos, a los cuales se aplica una función de activación que activa a la neurona si esta supera un umbral dado. La función de activación además de cumplir la función de activar a las neuronas, capta las características no lineales

de los datos.

Una red neuronal (ANNs, de sus siglas en inglés) consiste de un conjunto de neuronas conectadas entre sí (Fig. 4.1-a), que al igual que en el caso de las neuronas reales son capaces de ser entrenadas para aprender a realizar una tarea dada generando diferentes conexiones entre ellas.

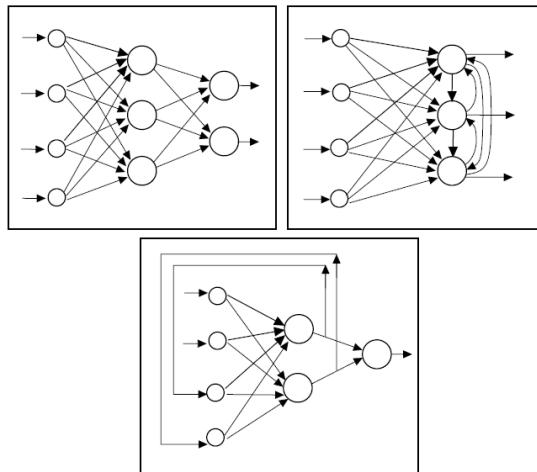


Figura 4.2: Tres topologías distintas de red neuronal: (a) multicapa, (b) competitivas, (c) recurrentes.

La estructura de la red puede tener distintas topologías, en la figura 4.2 se muestran tres modelos de red neuronales distintos: *red multicapa* que solo tiene conexiones entre neuronas de capas consecutivas, *red competitiva* que también posee conexiones entre las neuronas de la última capa y *redes recurrentes* (RNN) que poseen conexiones entre capas no consecutivas. En este trabajo se han utilizado tanto redes multicapa como redes recurrentes, que son apropiadas para problemas de aprendizaje supervisado en donde cada patrón de entrenamiento (entrada de la red)  $X_p = (x_{1p}, \dots, x_{mp})$  tiene asociado un correspondiente patrón de salida  $Y_p = (y_{1p}, \dots, y_{np})$ . El entrenamiento se basa en que la red sea capaz de reproducir los patrones de salida con el menor error posible.

El proceso de aprendizaje de la red consiste en la aplicación de métodos de optimización matemáticos para obtener los pesos  $w_{ij}$  que minimizan una cierta función de coste o *loss*. Los algoritmos más populares se basan en minimizar el error cuadrático medio (RMSE):

$$E(w) = \sum_{j,p} (y_{jp} - \hat{y}_{jp})^2 \quad (4.1)$$

Uno de los algoritmos más simples es el de descenso de gradiente que en cada etapa intenta modificar los pesos de forma incremental de manera de minimizar la función de loss. El incremento de los pesos se obtiene en base al vector opuesto al gradiente del loss, que indica la dirección en la que la función decrece más rápidamente:

$$\Delta w_{ij} = -\alpha \frac{\partial E(w)}{\partial w_{ij}} \quad (4.2)$$

donde  $\alpha$  es la tasa de aprendizaje.

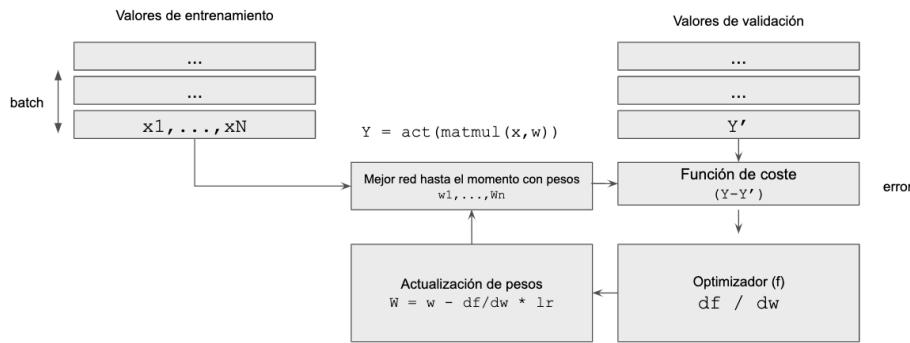


Figura 4.3: Back propagation

El algoritmo de retro-propagación consta de dos pasos, en primer lugar la entrada  $X_p$  se propaga hacia adelante, obteniendo el valor de todas las unidades ocultas y las salidas  $\hat{y}_p$  y por lo tanto el error asociado a éstas. Los valores obtenidos se utilizan para actualizar los pesos de la capa de salida, y éstos se propagan hacia atrás para actualizar los pesos de las capas ocultas. Esto se repite para cada patrón de entrenamiento. En el esquema 4.3 se resume todo el proceso.

## 4.2. Redes recurrentes con memoria a largo plazo

Las RNNs con memoria a largo plazo (LSTM) son redes que poseen conexiones entre capas no consecutivas (Fig. 4.2-c) que además incluyen una conexión extra entre las neuronas dedicadas a almacenar información para largo períodos de tiempo. Esta “memoria a largo” plazo soluciona el problema del desvanecimiento del gradiente que presentan las redes recurrentes tradicionales[14], [2].

En la figura 4.4 se puede observar la estructura de una celda de una red LSTM, donde  $x_t$  es un input dado a un tiempo  $t$ ,  $h_t$  el correspondiente output, las celdas amarillas son capas de neuronas densas y

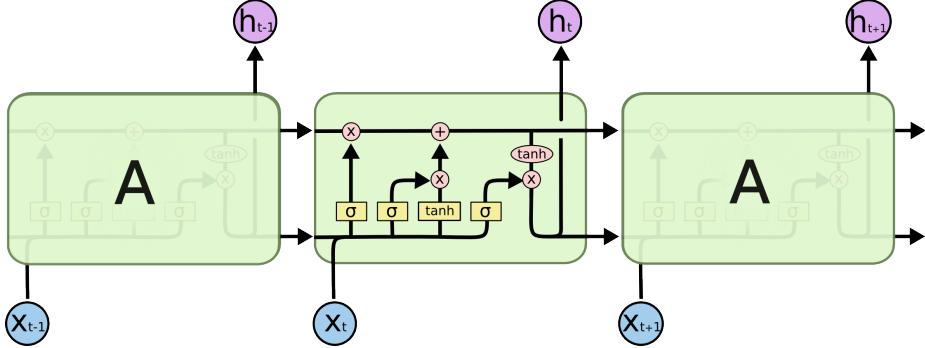


Figura 4.4: Red neuronal recurrente con memoria a largo y corto plazo (LSTM). Figura tomada de [2].

los círculos rosas son puntos en donde se realiza una cierta operación. La clave en estas redes es la línea horizontal superior que recorre la cadena casi sin sufrir modificaciones y permite que la información fluya a través de la red. Mediante las cuatro entradas (celdas amarillas en el esquema), estas redes poseen la habilidad de eliminar o agregar información al estado de la celda y decidir qué parte de la información proveniente de capas anteriores es relevante y qué parte debe ser olvidada [2].

En la figura 4.5, se muestra el recorrido paso a paso del funcionamiento de una celda LSTM. El primer paso (panel 1) es decidir que parte de la información será descartada. Esta decisión se realiza por una sigmoida llamada en inglés “*forget gate layer*”:

$$f_t = \sigma(W_f \cdot [h_{t-1}, x_t] + b_f) \quad (4.3)$$

Esta puerta recibe el input actual  $x_t$  y la salida del tiempo anterior  $h_{t-1}$  y arroja un valor entre 0 y 1, donde 0 significa “olvidar esto completamente” y 1 significa “mantener esto completamente”.

El siguiente paso (paneles 2 y 3) se encarga de actualizar el estado de la celda del valor de  $C_{t-1}$  al nuevo valor  $C_t$  y tiene dos partes, primero se multiplica el antiguo estado por  $f_t$ , y luego se suma el valor del nuevo candidato  $C'_t$ , escalado por un factor que cuantifica la magnitud del cambio:

$$C_t = f_t \cdot C_{t-1} + i_t \cdot C'_t \quad (4.4)$$

donde,

$$i_t = \sigma(W_i \cdot [h_{t-1}, x_t] + b_i) \quad (4.5)$$

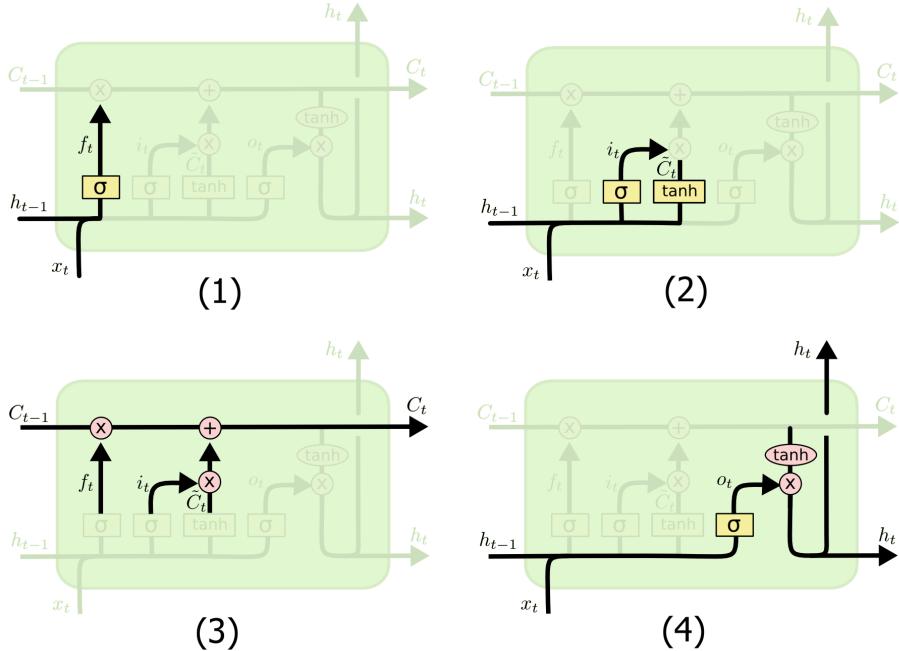


Figura 4.5: Recorrido por el funcionamiento de una celda LSTM. Figuras tomadas de [2].

$$C'_t = \tanh(W_C \cdot [h_{t-1}, x_t] + b_C) \quad (4.6)$$

Finalmente, se calcula la salida  $h_t$  de la siguiente manera (panel 4):

$$o_t = \sigma(W_o \cdot [h_{t-1}, x_t] + b_o) \quad (4.7)$$

$$h_t = o_t \cdot \tanh(C_t) \quad (4.8)$$

### 4.3. Topología de los modelos utilizados

Se han considerados 3 modelos que poseen una estructura básica que consta de una capa de entrada, dos o tres capas ocultas y una capa de salida:

- 1. Modelo Denso:** En primer lugar se ha considerado un modelo que consta de dos capas densas, en la figura 4.6 podemos ver un resumen con las principales características de las mismas.

2. **Modelo LSTM1:** En segundo lugar se ha considerado un modelo que contiene una capa LSTM y una segunda capa densa (Fig. 4.7).
3. **Modelo LSTM2:** Por último se ha considerado un tercer modelo que consta de dos capas LSTM y una capa densa (Fig. 4.8) que se ha entrenado de manera secuencial.

Layer (type)	Output Shape	Param #
<hr/>		
flatten_3 (Flatten)	(None, 9)	0
dense_31 (Dense)	(None, 200)	2000
dense_32 (Dense)	(None, 200)	40200
dense_33 (Dense)	(None, 76)	15276
<hr/>		
Total params: 57,476		
Trainable params: 57,476		
Non-trainable params: 0		

---

Figura 4.6: Modelo Denso.

Model: "sequential_13"		
Layer (type)	Output Shape	Param #
<hr/>		
lstm_10 (LSTM)	(None, 200)	168000
dense_29 (Dense)	(None, 200)	40200
dense_30 (Dense)	(None, 76)	15276
<hr/>		
Total params: 223,476		
Trainable params: 223,476		
Non-trainable params: 0		

---

Figura 4.7: Modelo LSTM1.

#### 4.3.1. Regularización de la función de coste

Se han considerado dos tipos de regularizaciones, L1 y L2 que penalizan el loss añadiendo los siguientes términos para cada caso:

$$loss + \lambda \sum |\omega_{ij}| \quad (L1) \quad (4.9)$$

```

Model: "sequential_15"
=====
Layer (type)          Output Shape         Param #
=====
lstm_11 (LSTM)        (None, 1, 100)      41600
lstm_12 (LSTM)        (None, 100)         80400
dense_34 (Dense)      (None, 100)         10100
dense_35 (Dense)      (None, 1)           101
=====
Total params: 132,201
Trainable params: 132,201
Non-trainable params: 0

```

Figura 4.8: Modelo LSTM2.

$$loss + \lambda \sum \omega_i^2 j \quad (L2) \quad (4.10)$$

La función de estos términos extra es la de reducir el valor de los parámetros haciendo incluso que la influencia de algunas variables de entrada sea nula en la salida de la red, lo que da lugar a una selección de variables de forma natural y reduce el posible sobreajuste de los datos. El valor de  $\lambda$  varía entre 0 y 1 y controla la magnitud de la reducción de los parámetros.

#### 4.3.2. Funciones de activación

En todas las capas se ha considerado la misma función de activación  $f$ , y se han entrenado los modelos considerando las opciones mostradas en 4.1.

$f(x) = \frac{1}{e^{-x}+1}$ (sigmoide)	$f(x) = \begin{cases} 0 & \text{si } x \leq 0 \\ x & \text{si } x > 0 \end{cases}$ (relu)
$f(x) = \frac{e^{2x}-1}{e^{2x}+1}$ (tanh)	$f(x) = x$ (linear)

Cuadro 4.1: Funciones de activación

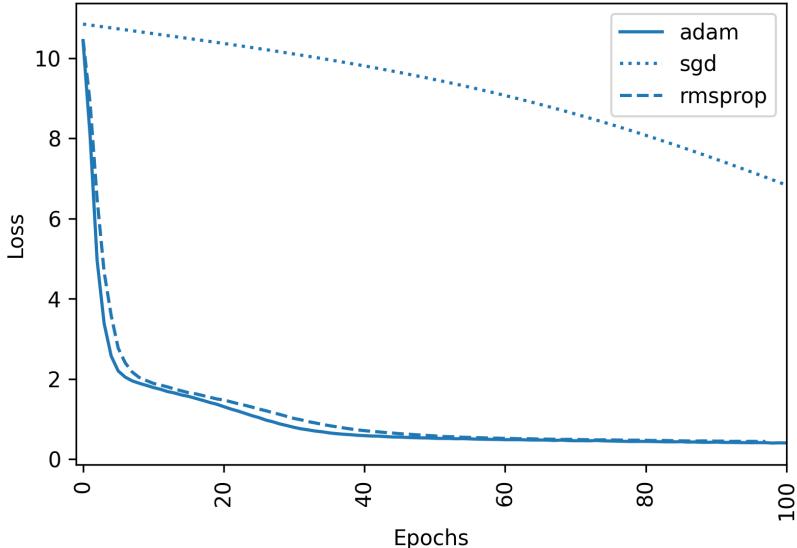


Figura 4.9: Evolución de la función de loss considerando diferentes optimizadores.

#### 4.3.3. Optimizadores

Dado el gran número de diferentes pesos y observaciones, el cálculo de la derivada parcial de la función de coste respecto a cada uno de los pesos de la red para cada observación es inviable. Existen diferentes métodos que optimizan este proceso y agilizan los cálculos. En este trabajo se ha hecho una exploración inicial manual para evaluar la performance de los optimizadores ”*Stochastic Gradient Descent*” (SGD) [23], ”*Adaptive moment estimation*” (Adam) [24] y *Root Mean Square Propagation* (RMSprop) [23].

En la figura 4.9 se muestra a modo de ejemplo la evolución de la función de loss considerando cada uno de los optimizadores al entrenar el modelo LSTM1, se puede ver que la taza de aprendizaje con adam y rmsprop es considerablemente mayor que con sgd, ya que en los primeros casos la función de loss alcanza su valor de equilibrio en aproximadamente 60 épocas, mientras que sgd necesita más de 200.

#### 4.3.4. Calibración de los modelos

Los modelos se calibran realizando una búsqueda exhaustiva de hyper-parámetros utilizando el con el método ”*GridSearchCV*” disponible en la librería scikit-learn [25]. Éste método crea una grilla con los hyper-parámetros del modelo y optimiza sus valores realizando una búsqueda



Figura 4.10: k-fold cross validation (imagen de Gufosowa - WikiMedia).

con validación cruzada, en donde el conjunto de datos se divide en datos de entrenamiento y de validación (train/test en la figura 4.11). Cómo este conjunto se divide y cuantas veces se realiza el proceso, se controla mediante el número de pliegues o folds, en general denotado con la letra  $k$ . El modelo utiliza el primer fold en la primera iteración como conjunto de test y los siguientes  $k-1$  como conjunto de train. Este proceso se repite  $k$  veces para cada uno de las posibles combinaciones de hyper-parámetros presentes en la grilla y arroja como resultado final un valor promedio del escore utilizado para evaluar la calidad del modelo.

En la tabla 4.2 se muestran los hyper-parámetros explorados para entrenar los diferentes modelos. En el caso del entrenamiento global, realizado para los modelos denso y LSTM1 PCA, tres de estos parámetros (el número de neuronas, la función de activación y  $\lambda$ ) han sido optimizados mediante el método GridSearchCV.

#### 4.4. Entrenamiento

Los modelos han sido entrenados durante un número de épocas o pasos igual a 200 y se ha utilizado el callback *"early stopping"* con paciencia 3, que interrumpe la ejecución cuando el valor del loss en el conjunto de validación aumenta durante tres iteraciones sucesivas. Se han considerado tres métodos diferentes de entrenamiento que se describen a continuación.

Modelo	Parámetros	valores
Denso	solver Neuronas activación alpha Regularización	rmsprop, adam, sgd 10, 80, 200 sigmoid,relu,tanh,linear 0.0001,0.0002,0.0003 l1,l2 $\lambda = 0,0001, 0,001, 0,1$
LSTM1 PCA	solver Neuronas activación alpha Regularización	rmsprop, adam, sgd 10, 80, 200 sigmoid,relu,tanh,linear 0.0001,0.0002,0.0003 l1,l2 $\lambda = 0,0001, 0,001, 0,1$
LSTM1 loc	solver Neuronas activación alpha Regularización	rmsprop, adam, sgd 10, 80, 200 sigmoid,relu,tanh,linear 0.0001,0.0002,0.0003 l1,l2 $\lambda = 0,0001$
LSTM2	solver Neuronas activación alpha Regularización	rmsprop, adam, sgd 10, 80, 200 sigmoid,relu,tanh,linear 0.001 l1,l2 $\lambda = 0,0000001$

Cuadro 4.2: Hyper parámetros explorados en los diferentes modelos.

#### 4.4.1. Entrenamiento global

Los modelos denso y LSTM1 han sido entrenados sobre una matriz que contiene las entradas o variables características (series temporales de precipitación, temperatura máxima y temperatura mínima) para todas las subcuencas que constituyen la cuenca hidrológica Chambo:

$$X_{comp} = \begin{bmatrix} p_{1,1} & T_{1,1}^{max} & T_{1,1}^{min} & p_{1,2} & T_{1,2}^{max} & T_{1,2}^{min} & \dots & p_{1,nc} & T_{1,nc}^{max} & T_{1,nc}^{min} \\ \dots & \dots \\ p_{nt,1} & T_{nt,1}^{max} & T_{nt,1}^{min} & p_{nt,2} & T_{nt,2}^{max} & T_{nt,2}^{min} & \dots & p_{nt,n} & T_{nt,n}^{max} & T_{nt,n}^{min} \end{bmatrix}$$

Donde  $n_t$  es el número de pasos temporales y  $nc$  es el número de subcuencas. Las variables objetivo son los caudales naturales simulados con el modelo hidrológico LEM y se encuentran almacenados en una matriz  $Y_{comp}$  estructurada de la siguiente manera:

$$Y_{comp} = \begin{bmatrix} q_{1,1} & q_{1,2} & q_{1,2} \dots & q_{1,nc} \\ \dots & \dots & \dots & \dots \\ q_{nt,1} & q_{nt,2} & q_{nt,2} \dots & q_{nt,nc} \end{bmatrix}$$

El paso de tiempo que se ha considerado es mensual y el rango temporal total es de 20 años (desde el año 2000 hasta el año 2020). Por otro lado la cuenca se encuentra compuesta por 76 subcuenas, entonces las dimensiones de  $X_{comp}$  e  $Y_{comp}$  son (229,228) y (229,76), respectivamente. Finalmente, los datos han sido divididos en conjuntos de train y test tomando los primeros 160 como conjunto de train y los últimos 69 como conjunto de test.

Para el caso de capas LSTM es necesario agregar una dimensión a las matrices de entrada que toma en cuenta la correlación temporal de las muestras. La estructura de las matrices de entrada toman la forma (*muestras, pasos de tiempo, características*). En el modelo LSTM1 se considera un paso de tiempo por cada muestra, por lo cual las matrices toman la dimensión (229,1,228) y (229,1,76), respectivamente.

#### 4.4.2. Entrenamiento local

Por otro lado el modelo LSTM1 también se ha entrenado individualmente cuenca por cuenca, para lo cual se han utilizado matrices  $X_{loc}$  e  $Y_{loc}$  con las siguientes formas:

$$X_{loc,id} = \begin{bmatrix} p_{1,id} & T_{1,id}^{max} & T_{1,id}^{min} \\ \dots & \dots & \dots \\ p_{nt,id} & T_{nt,id}^{max} & T_{nt,id}^{min} \end{bmatrix}$$

$$Y_{loc,id} = \begin{bmatrix} q_{1,id} \\ \dots \\ q_{nt,id} \end{bmatrix}$$

Donde  $id$  es el número de la subcuenca.

En este último caso, las matrices poseen las dimensiones (229,1,3) y (229,1,1), respectivamente. La división entre los conjuntos de entrenamiento y test se ha hecho de la misma manera que en el apartado anterior.

#### 4.4.3. Entrenamiento secuencial

Finalmente, el modelo LSTM2 se ha entrenado de manera secuencial, utilizando únicamente los valores simulados de los caudales en la matriz  $Y_{loc,id}$ . En este enfoque, la red utiliza los valores de los caudales en tiempos anteriores  $q_{t-1}, q_{t-2}, q_{t-n}$  para predecir el valor actual  $q_t$ . La variable  $n$  también llamada "look back" se puede elegir arbitrariamente y se puede ajustar haciendo una busqueda con el método de cross-validation.

### 4.5. Análisis de componentes principales

El espacio de variables predictoras del conjunto de entrenamiento que contiene la información de las 76 subcuenca posee una dimensión: (160,228). Sin embargo, existen correlaciones entre las variables que provocan que muchas de estas aporten información poco relevante o redundante [26]. Estas correlaciones pueden distorsionar el proceso de aprendizaje de los modelos [1] y es por eso que se ha optado por reducir el número de variables mediante el análisis de componentes principales[27].

Este método realiza una transformación del espacio predictor a un espacio vectorial cuya base son los auto-vectores de la matriz de covariancia. En este espacio, los valores en la diagonal de dicha matriz son las varianzas en la dirección de cada uno de los vectores de la base o componentes principales. La reducción de dimensiones se realiza escogiendo las direcciones que captan la mayor variación de los datos originales, es decir las PCA con las mayores varianzas.

Luego de realizar este análisis se ha encontrado que se puede explicar el 96 % de la varianza del conjunto de datos original considerando sólo las primeras 9 componentes (ver figura ??). Por lo tanto la reducción del espacio predictor es considerable, ya que ha pasado de ser (160,228) a (160,9).

### 4.6. Coeficiente de Nash-Sutcliffe

Además de utilizar la función de loss para validar los modelos durante el entrenamiento, también se ha utilizado el coeficiente Nash-Sutcliffe (NSE) [28] para evaluar la performance de los mismos en el conjunto de test. Este coeficiente es uno de los más utilizados en hidrología y se define de la siguiente manera:

$$NSE = 1 - \frac{\sum_i^n (y_i - \hat{y}_i)^2}{\sum_i^n (\bar{y} - \hat{y}_i)^2} \quad (4.11)$$

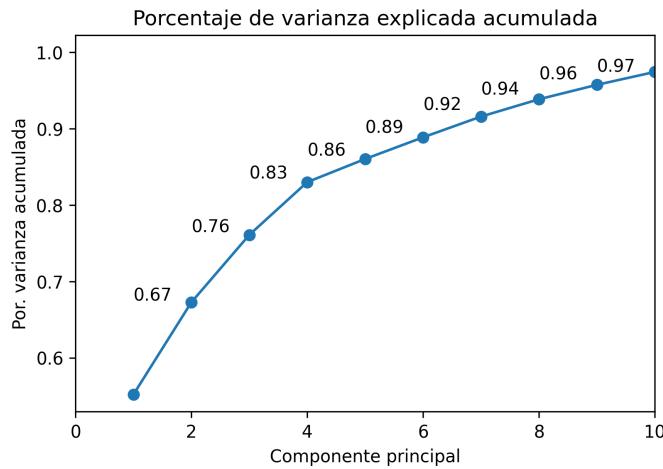


Figura 4.11: Porcentaje de varianza explicada acumulada para las primeras 9 componentes principales.

Calidad del ajuste	NSE
Excelente	$0,75 < NSE \leq 1,00$
Bueno	$0,65 < NSE \leq 0,75$
Aceptable	$0,5 < NSE \leq 0,65$
No aceptable	$NSE \leq 0,5$

Cuadro 4.3: Calidad de los ajustes en función del coeficiente NSE [28].

Los valores de NSE varían entre  $-\infty$  y 1, siendo este último valor el correspondiente a un ajuste perfecto. En la tabla 4.3 se indica la relación entre los valores de NSE y la calidad de los ajustes. Este coeficiente equivale a una combinación lineal del error medio cuadrático, la desviación y el coeficiente de correlación entre las series.

# Resultados

En este capítulo se muestran los resultados obtenidos al entrenar los modelos previamente descritos, el objetivo es predecir los caudales de descarga naturales en cada una de las sub-cuencas que conforman CHRC para un evento de precipitación dado. La performance predictiva de los modelos es evaluada en el conjunto de test, en la tabla 5.1 se muestran los hyper-parámetros utilizados para cada modelo obtenidos tras realizar la exploración descrita en 4.3.4.

## 5.1. Validación de los modelos

Como se ha mencionado en la sección 4.6, se ha utilizado el coeficiente de Nash-Sutcliffe para validar los modelos y hacer un análisis un poco más profundo sobre cómo es su performance en los diferentes puntos de la cuenca. En la figura 5.1 se muestran los valores obtenidos de NSE para todas las sub-cuencas de Chambo considerando el modelo denso y el modelo LSTM1 entrenado de manera global en el espacio de las componentes principales. Las líneas horizontales muestran los umbrales correspondientes a ajustes excelentes y aceptables.

El modelo LSTM1 posee una mejor performance general a lo largo de toda la cuenca, en este caso se puede considerar que el ajuste para el 68 % de las sub-cuencas es excelente, mientras que el 11 % de los ajustes son buenos y el 16 % aceptables. Por otro lado, si bien la performance general del modelo denso es buena (el 68 % es mayor a 0,65), el porcentaje de fallos asciende al 13 %. Se puede observar que en las subcuenca con ids 5, 26, 31 y 66 el coeficiente NSE adquiere valores negativos, mientras que el modelo LSTM1 realiza muy buenos ajustes. La mejor performance del modelo LSTM1 se debe a que las celdas de memoria permiten una interpretación a lo largo del eje del tiempo de los diferentes procesos que ocurren en cada una de las cuencas, lo

Modelo	optimizador	Neuronas	Activación	alpha	reg
Denso	rmsprop	200	relu	0.0002	l2, 0.0001
LSTM1 PCA	rmsprop	200	linear	0.0002	l2, 0.0001
LSTM1 loc	adam	200	relu	0.0001	l1, 0.0001
LSTM2	adam	200	relu	0.001	l2, 0.0000001

Cuadro 5.1: Hyper parámetros utilizados en los diferentes modelos.

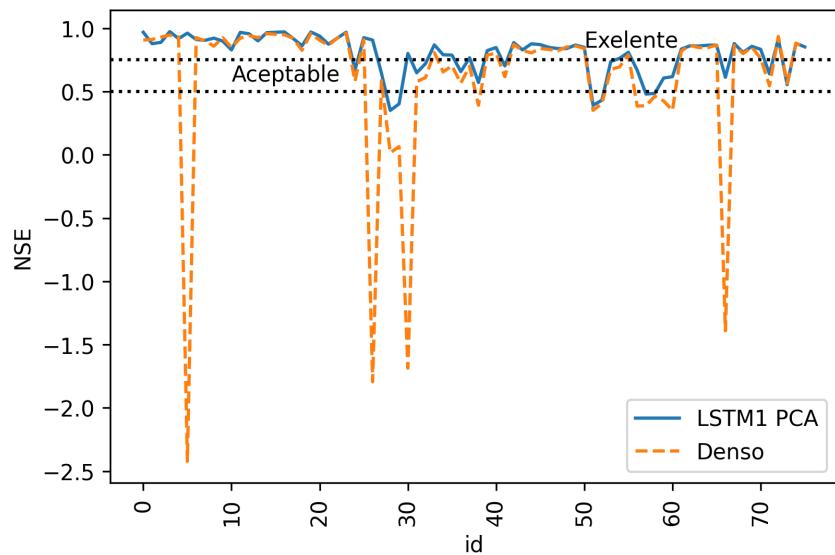


Figura 5.1: Valores obtenidos para el coeficiente NSE a lo largo de toda la cuenca con los modelos denso y LSTM1 entrenado con PCAs.

que permite captar más información sobre la relación entre eventos de precipitación y descarga.

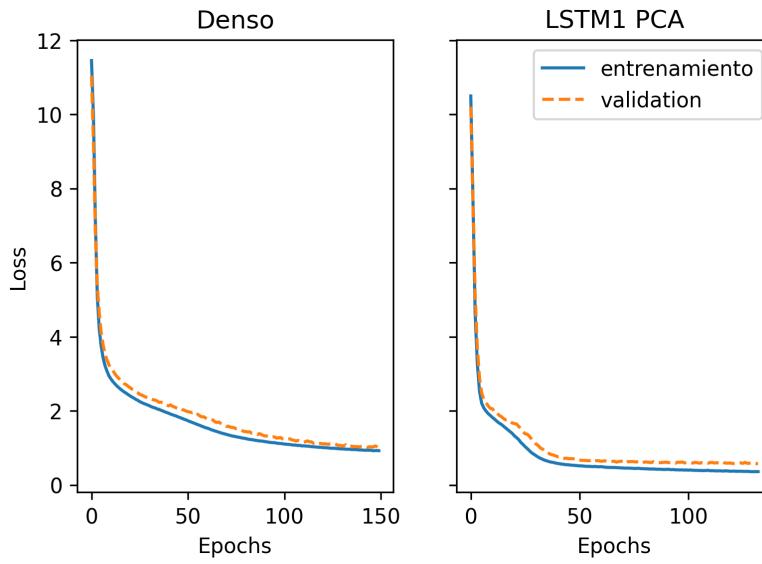


Figura 5.2: Evolución de la función de loss en los conjuntos de entrenamiento y validación para los modelos denso y LSTM1 con PCAs.

En la figura 5.2 se muestran los valores de la función de loss obtenidos en cada época o iteración durante el entrenamiento de los modelos denso y LSTM1 PCA. Durante el entrenamiento se ha dividido el conjunto de datos de manera que el 85 % de los mismos han sido utilizados para entrenar el modelo y el 15 % restante para validarla y así evitar el sobre ajuste monitorizando la evolución del loss. Además se ha implementado el callback *"early stopping"* que interrumpe la ejecución cuando el loss en el conjunto de validación comienza a aumentar.

Si se comparan ambos modelos, podemos observar que en el modelo LSTM1 PCA aprende más rápido ya que en aproximadamente 50 épocas el valor de loss alcanza su valor de equilibrio, en el caso del modelo denso, esto ocurre luego de 150 épocas. Por otro lado, y en concordancia con los resultados arrojados por el coeficiente NSE, el valor final del loss en el modelo LSTM1 PCA (entrenamiento: 0.3587 , validación: 0.5787) es menor que en el modelo denso (entrenamiento: 0.9245 validación: 1.0060) los que indica un mejor ajuste global.

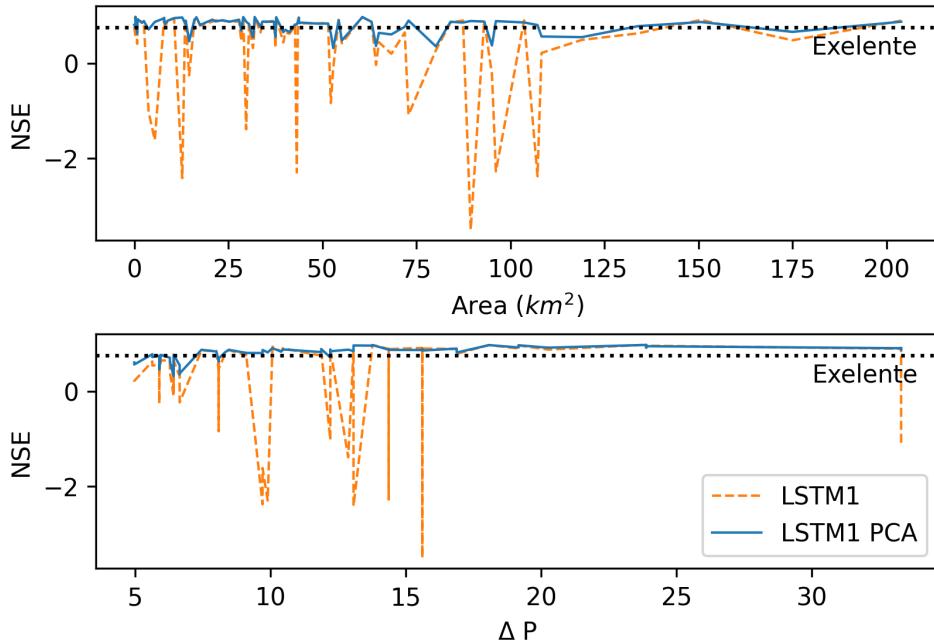


Figura 5.3: Evolución del coeficiente NSE en función del área (panel superior) y la variación de precipitación (panel inferior) para los modelos LSTM1 con y sin PCAs.

En la figura 5.3, se muestra la comparación entre los valores de NSE obtenidos para el modelo LSTM1 con PCAs (curva continua) y sin

PCAs (curva a rayas) en función del área de la cuenca (panel superior) y de la variación de precipitación (panel inferior). Los resultados obtenidos cuando consideramos las 288 características de entrada muestran una gran fluctuación de los valores de NSE y en general peores predicciones. Una razón por la que las componentes principales funcionan mejor puede deberse a que éstas agregan la información proveniente de todo el dominio del espacio predictor [26]. En línea con este argumento, en estas figuras también se puede observar que el área de las sub-cuenca y la variación de precipitación son factores relevantes que condicionan la calidad de los ajustes. El modelo que no incluye componentes principales tiende a fallar más en las cuencas pequeñas ( $Area < 120 \text{ km}^2$ ) y áridas ( $\Delta P < 20 \text{ mm/d}$ ). Esto puede deberse a que en estos casos los valores de la precipitación y caudales son más pequeños y el loss es en general menor que el loss para una sub-cuenca con una descarga grande. Esto último provoca que el modelo deje de aprender demasiado pronto y se genere un sobre peso para las sub-cuenca más grandes y húmedas mientras que la performance en las más pequeñas y áridas disminuye [14].

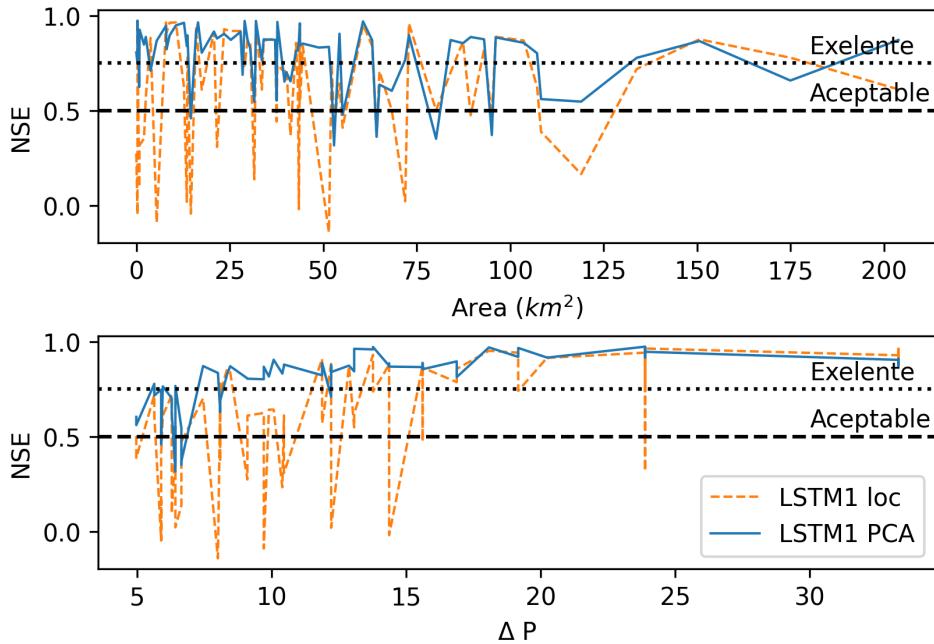


Figura 5.4: Evolución del coeficiente NSE en función del área (panel superior) y la variación de precipitación (panel inferior) para los modelos LSTM1 con PCAs y LSTM1 entrenado localmente.

Este efecto es también visible cuando entrenamos el modelo LSTM1 de manera local, como puede verse en la figura 5.4 en la mayoría de los casos, la performance del modelo global es mejor que la del modelo local. Este resultado se encuentra alineado con estudios que demuestran que los modelos LSTM entrenados en un gran número de sub-cuencas son capaces de vincular Las características de las mismas para aprender un modelo global que a su vez es capaz de reflejar explícitamente las similitudes y diferencias de las cuencas a nivel local [14].

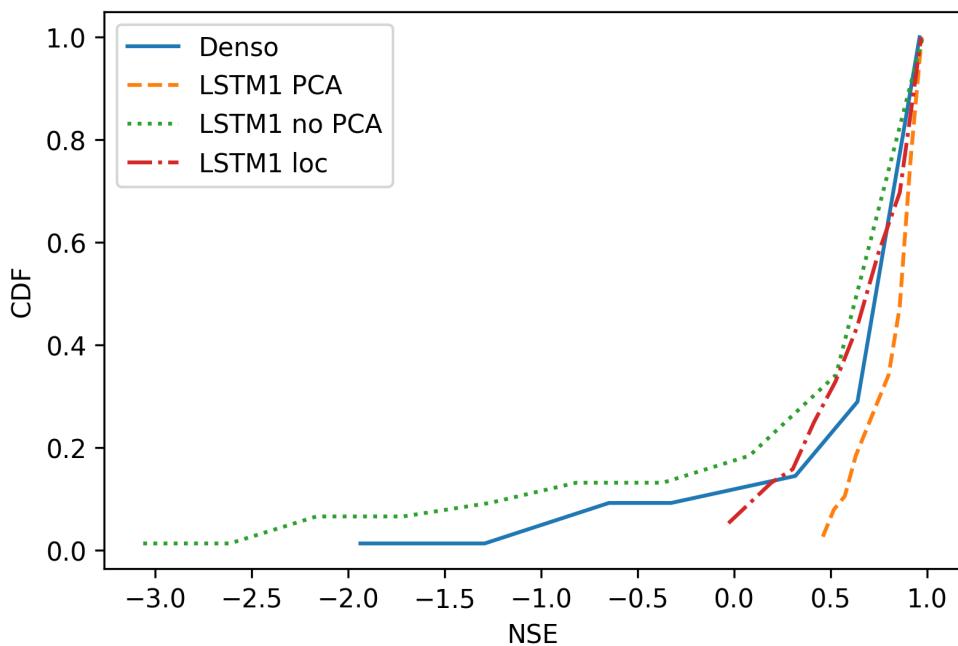


Figura 5.5: Función de distribución acumulada para todos los modelos de ANNs a lo largo de todas las sub-cuencas de CHRC.

En la figura 5.5, se muestra una comparación de la función de distribución acumulada para todos los modelos sobre las 76 sub-cuencas de CHRC. En esta figura se puede ver claramente que el modelo que mejor funciona en general es el LSTM1 PCA. El modelo denso es el segundo mejor en algunos sectores y el que muestra la peor performance en toda la cuenca es el modelo que usa el espacio predictor sin hacer el análisis de componentes principales.

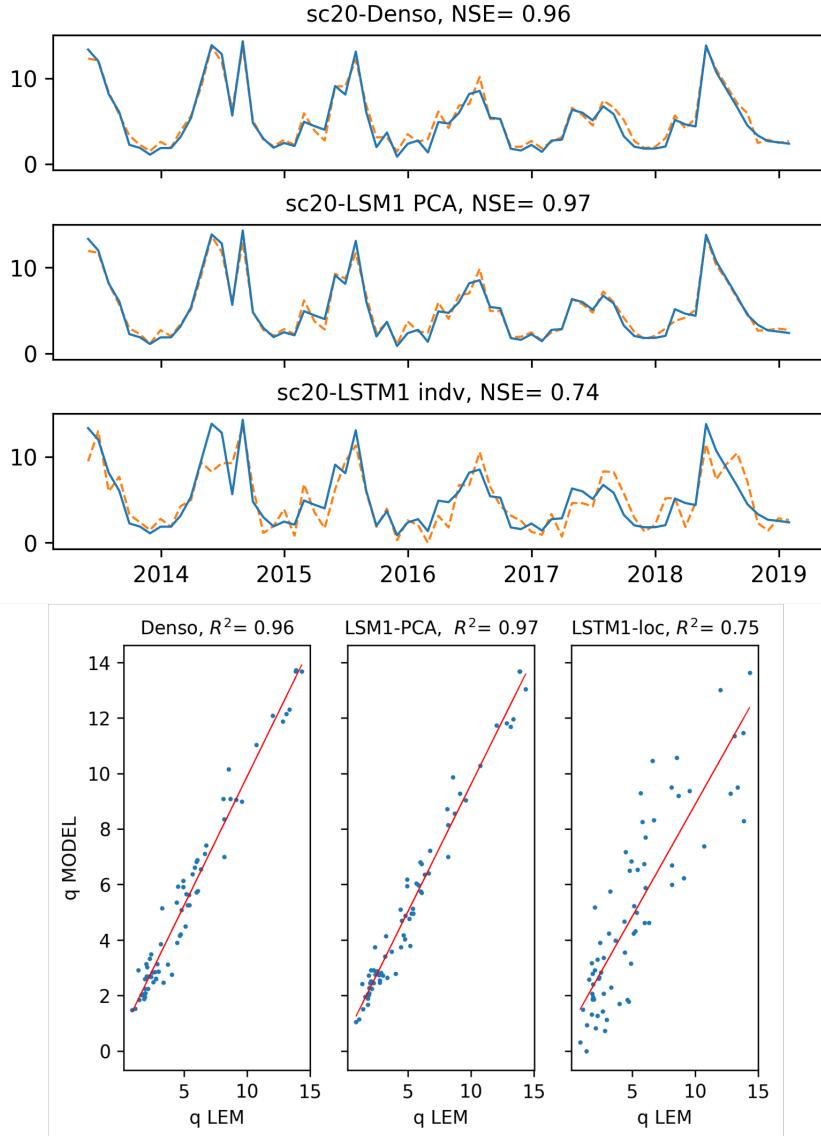


Figura 5.6: Predicciones del caudal de descarga obtenidos con los modelos Denso, LSTM1 con PCAs y LSTM1 local en la subcuenca con id 20. En los paneles inferiores se muestran los grados de ajuste correspondientes.

## 5.2. Predicción de los caudales naturales

En las figuras 5.6 se muestran los resultados obtenidos para los caudales en la sub-cuenca con id 20 y los grados de ajustes correspondientes obtenidos con los diferentes modelos. Las curvas continuas son

los valores simulados con el modelo MELCA y las curvas a rayas son los valores predichos con los modelos denso, LSTM1 PCA y LSTM1 loc en el conjunto de test. En este caso la performance de los modelos LSTM1 PCA y denso es excelente, con valores de NSE iguales a 0.97 y 0.96, respectivamente, mientras que el modelo entrenado localmente es un poco más baja. Como se ha explicado en sesiones anteriores, los modelos entrenados de manera global con componentes principales, aprenden en un espacio que contiene información agregada proveniente de todo el espacio predictor. El ajuste del modelo LSTM1 PCA es aún mejor porque la presencia de neuronas recurrentes permiten contemplar la correlación de los datos en la dimensión temporal.

En la figura 5.7 se muestra a modo de ejemplo un caso en el que el modelo LSTM1 loc falla al predecir los valores en el conjunto de test, mientras que los modelos Denso y LSTM1 PCA arrojan un ajuste aceptable ( $NSE > 0,6$ ). En este caso, el modelo LSTM1 loc ha sido capaz de reflejar cierta tendencia de los datos pero aún así la calidad del ajuste es baja ( $NSE < 0,5$ ). El modelo LSTM1 PCA, que posee ambas ventajas, la de considerar componentes principales y redes recurrentes, es capaz de predecir los valores de los caudales en el conjunto de test con una calidad buena ( $NSE = 0,7$ ).

En la figura 5.8 se muestran los resultados obtenidos para el modelo LSTM2 entrenado secuencialmente. en el panel superior se muestran los valores del coeficiente NSE a lo largo de toda la cuenca y en el panel inferior los resultados obtenidos para el mejor ajuste. Se puede observar que la performance del modelo es en general bastante pobre, sólo unos pocos puntos sobrepasan en umbral de calidad aceptable. Los mayores problemas que presenta esta aproximación es que por un lado los errores cometidos en cada una de las predicciones se propagado y acumulado a lo largo del tiempo y por otro lado se pierde completamente la información otorgada por las series hidro-climáticas de entrada, por lo cual el modelo no es capaz de aprender ningún patrón relacionado con la respuesta que las diferentes sub-cuenca tienen frente a las series de precipitación.

### 5.3. Cálculo de balance hídrico con MODSIM

Con el fin de determinar si los resultados obtenidos con redes neuronales pueden ser utilizados para simular operaciones en la cuenca CHRC, se ha determinado el balance hidrológico utilizando el software MODSIM (introducido en 3.2) con los caudales simulados por el modelo hidrológico MELCA y las predicciones de los modelos de redes neuronales en el conjunto de test.

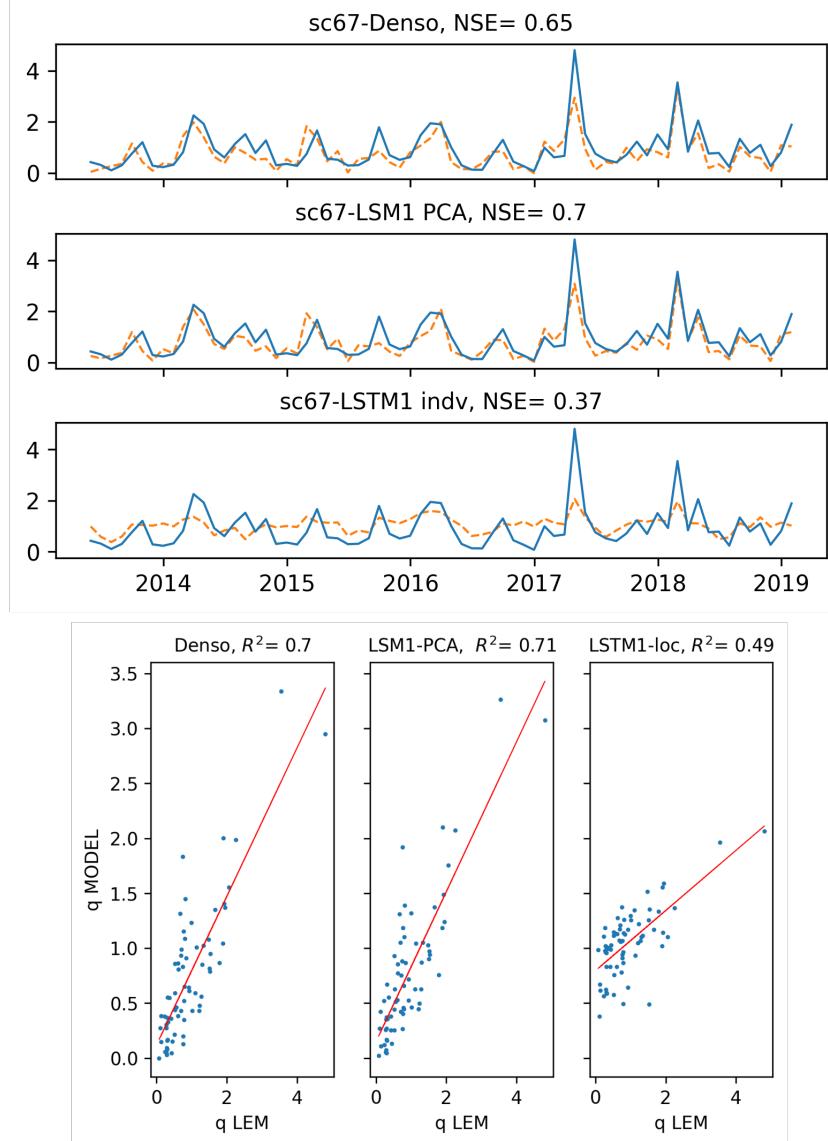


Figura 5.7: Predicciones del caudal de descarga obtenidos con los modelos Denso, LSTM1 con PCAs y LSTM1 local en la subcuenca con id 67. En los paneles inferiores se muestran los grados de ajuste correspondientes.

En el panel superior de la figura 5.9 se muestra el residuo ( $R$ ), es decir la diferencia entre los valores de los caudales predichos respecto a los caudales de referencia (en este caso simulados con el modelo MELCA), para diferentes sub-cuenca distribuidas a lo largo de toda la cuenca hidrográfica Chambo. En los paneles inferiores se muestran los

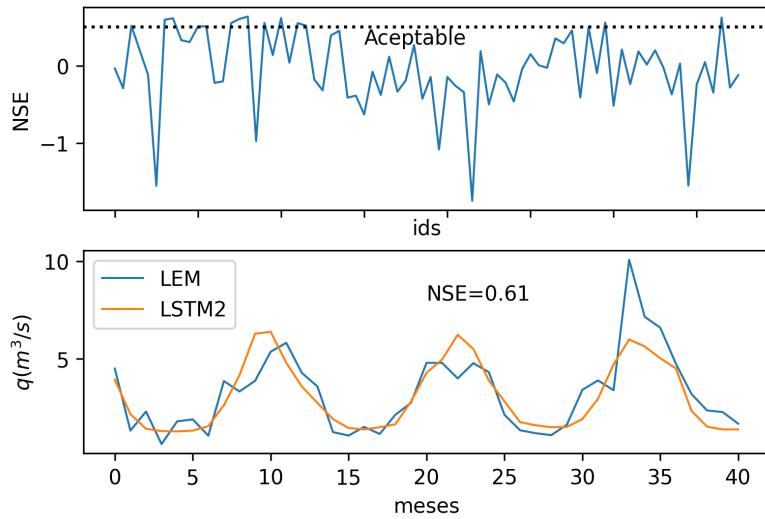


Figura 5.8: Resultados del Modelo LSTM2. En el panel superior se muestran los valores de NSE a lo largo de toda la cuenca y En el panel inferior, los valores predichos en la sub-cuenca con id 15.

valores de los caudales arrojados por MODSIM para la sub-cuenca que se encuentra a la salida de CHRC (aguas abajo) y para una de las subcuenca que se encuentra a mayor altitud (aguas arriba). Las curvas continuas corresponden al resultado obtenido con los caudales simulados con el modelo MELCA, y la curva a rayas es el resultado obtenido cuando consideramos los valores predichos por el modelo LSTM1 PCA. Se puede apreciar que los resultados arrojados por MODSIM son los mismos para ambos casos, es decir, las pequeñas variaciones mostradas en el panel superior no influyen en el resultado final.

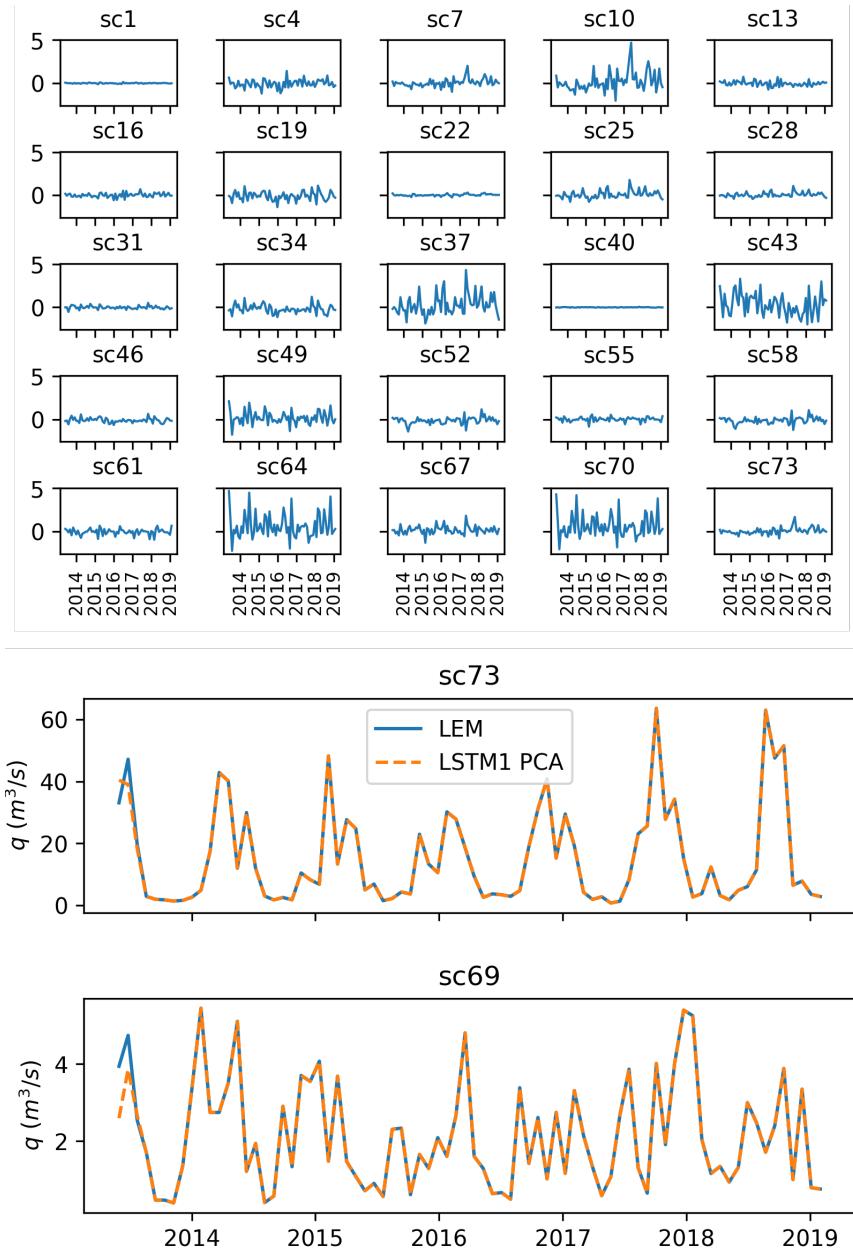


Figura 5.9: Paneles superiores: residuos obtenidos a lo largo de diferentes sub-cuencas. Paneles inferiores: resultados finales obtenidos utilizando el software de gestión de recursos MODSIM, teniendo en cuenta los usos del agua.

## Discusiones y Conclusiones

Los resultados descritos en el capítulo anterior demuestran que el modelo que realiza los mejores ajustes es el LSTM1 cuando es entrenado en el espacio de las componentes principales. Aproximadamente un 70 % de los ajustes obtenidos con el mismo son excelentes mientras que esta cifra desciende al 60 % para los modelos denso y LSTM1 entrenado localmente. Si bien los últimos modelos poseen una buena performance en la mayoría de las sub-cuencas, presentan mucha variación en el coeficiente de NSE, alcanzando incluso valores negativos en algunos puntos para los cuales el modelo LSTM1 PCA arroja resultados que son desde aceptables hasta excelentes. Por otro lado el experimento realizado con el modelo LSTM2, que considera un entrenamiento secuencial utilizando solamente los valores de la serie de caudales de descarga de las sub-cuencas, ha demostrado tener una mala performance en la mayoría de los puntos.

Los resultados antes mencionados nos llevan a concluir que la inclusión de las celdas de memoria LSTM y la presencia de componentes principales mejoran notablemente la performance de los modelos en cuencas hidrográficas. Por un lado las celdas LSTM permiten almacenar información de eje temporal sobre los diferentes procesos que ocurren en las sub-cuencas, y así captar más información sobre la relación entre eventos de precipitación y descarga. Por otro lado, las componentes principales reducen considerablemente la dimensión del espacio predictor y combinan la información proveniente de todo el dominio del mismo, lo que soluciona problemas de sobre ajustes y el problema des desvanecimiento del loss presente en las cuencas más pequeñas y áridas. Finalmente, una vez entrenados los modelos, se ha demostrado que los resultados obtenidos con redes LSTM pueden ser utilizados para simular operaciones en la cuenca CHRC, ya que se ha comprobado que el resultado del balance hidrológico obtenido con MODSIM es el mismo que los obtenidos con los valores simulados por el modelo LEM.

Todos estos resultados apuntan en la misma línea que estudios recientes en los cuales se concluye que las redes neuronales generalmente requieren una gran cantidad de datos de entrenamiento y que los ajustes que se obtienen al entrenar modelos de aprendizaje profundo en una sola sub-cuenca no suelen ser fiables [14]. Esto supone una gran diferencia con el modelado y calibrado hidrológico tradicional que normalmente demuestra una mejor performance cuando los modelos se calibran de forma independiente para cada sub-cuenca. Ésta propiedad de los

modelos clásicos presenta problemas, ya que se ha observado que los parámetros obtenidos por extrapolaciones basadas en valores calibrados en cuencas de referencia pueden dar lugar a espacios de parámetros poco realistas[29]. Los modelos LSTM en cambio, demuestran tener la capacidad de aprender simultáneamente relaciones de series temporales y espaciales en el mismo marco predictivo, lo que evita muchos problemas que actualmente se encuentran asociados con la estimación y transferencia de parámetros de modelos hidrológicos tradicionales [14], [7].

Una conclusión importante es entonces que las celdas LSTM son capaces de generar un modelo único a partir de grandes conjuntos de datos capaz de reflejar los comportamientos hidrológicos regionales específicos de cada sub-cuenca ya que estos modelos vinculan las características locales de las sub-cuencas y aprenden un modelo general a partir de los datos combinados de todas ellas. Es por esto que concluimos que la principal virtud de las redes neuronales no es simplemente el hecho de que ajusten bien sino su capacidad de aprendizaje y flexibilidad para ser utilizadas en una variedad de lugares y condiciones diferentes.

Por último, como posibles mejoras para trabajos futuros se propone concatenar los descriptores estáticos de las sub-cuencas, como por ejemplo el área, la longitud del cauce, etc., al espacio predictor. Otra mejora podría ser definir una función objetivo al entrenar los modelos que no dependa del valor medio de los caudales de descarga de las diferentes cuencas. Kratzert etal. [14] proponen por ejemplo usar directamente una definición global del coeficiente de Nash-Stutcliffe durante el entrenamiento de los datos. En este caso, si bien se perdería la linealidad entre las métricas del RMSE y el NSE, esto permitiría contemplar el hecho de que las variancias de los datos observacionales difieren en distintas cuencas y así evitar el sobre-peso asociado a las cuencas más grandes y húmedas.

## Anexo

Con el fin de poder ejecutar fácilmente los modelos de ANNs desarrollados y calcular el balance hídrico de la cuenca, se han creado dos aplicaciones REST APIs utilizando la librería flask de python [30]. La idea es que mediante la correcta estructuración de los datos de entrada necesarios para la ejecución de los modelos, estos se puedan ejecutar en diferentes cuencas. Para esto se han creado diferentes clases y métodos en python que constituyen el core de la API, estos métodos o recursos se hacen accesibles a través de la interfaz de la API, en la que se exponen los métodos y las URLs disponibles para acceder y/o manipular cada uno de ellos.

El proceso que se ha seguido para crear esta API es el siguientes:

- En primer lugar se ha construido una base de datos relacional que contiene los descriptores las cuencas hidrográficas (Fig. 7.2), así como también todos los datos referentes a los diferentes usos de agua. Esta base de datos fue realizada con el software PostgreSQL un sistema de código abierto de administración de bases de datos del tipo relacional cuyas consultas se basan en SQL[31]. Por otro lado las series temporales que actúan como inputs de los modelos no admiten consultas relacionales y por lo tanto se han almacenado de forma separada en archivos de excel en un directorio local.
- Una vez estructurados los datos en sus diferentes formatos, se ha desarrollado una serie de métodos en python para hacer todo tipo de consultas, agregaciones, actualizaciones, y borrado de los datos. Para esto se utilizó la librería psycopg2, una de las librerías más populares que sirven para integrar el lenguaje de PostgreSQL en python y permite crear métodos que hagan consultas en la base de datos creada con dicho software.
- El tercer paso es crear una API que contenga todos los métodos que faciliten el acceso a los datos, entre los métodos creados se destacan los que permiten recorrer la navegación de la cuenca, encontrar para una cuenca dada, las cuencas aguas arriba que vierten en ella, todas las demandas y aportaciones de agua, etc (Fig. 7.2).
- Una vez creara la API que contiene los métodos que permiten la consulta de la base de datos y las series temporales, se ha

creado otra aplicación que ejecuta los modelos desarrollados (Fig. 7.3). Para esto se ha creado una clase python, que posee varios métodos, que ejecutan los modelos, y calculan el balance hídrico de la cuenca. En este proceso se ha utilizado la librería request que permite hacer peticiones HTTP POST y GET a la base de datos que contiene las tablas con las series temporales y los descriptores necesarios para la ejecución de los modelos.

- Por último, una vez que se ha accedido a todos los datos necesarios para ejecutar los modelos que simulan/predicen los caudales naturales de las cuencas, y los datos asociados a todas las demandas y aportaciones de agua, se procede a calcular el balance hídrico de la cuenca siguiendo los pasos:
  1. para una subcuenca dada, se agregan los resultados de las simulaciones de caudal aguas arriba siguiendo el cauce de navegación de la cuenca,
  2. se averiguan todas las demandas que toman agua de dicha cuenca y todas las demandas que retornan agua a la misma,
  3. se calcula el balance como:  $q_{ac} - q_{dem} + q_{ret}$ , donde  $q_{ac}$  el caudal natural agregado aguas arriba,  $q_{dem}$  el caudal agregado de todas las demandas y  $q_{ret}$  el retorno total de todas las demandas.

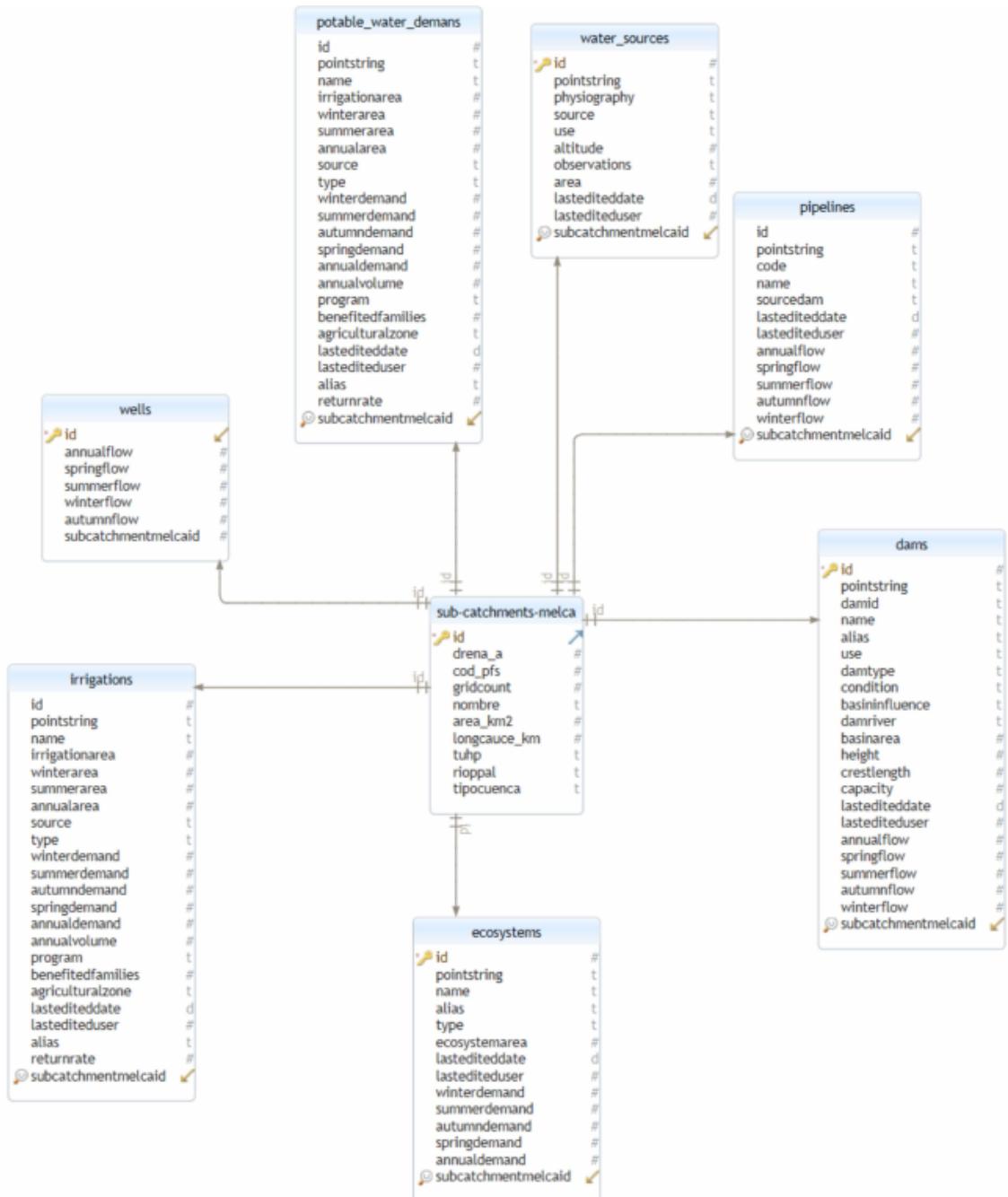


Figura 7.1: Base de datos relacional con los descriptores de las componentes de una cuenca hidrográfica.

GET /Hydrographies/potable\_water\_demands/{Sub\_Catchment\_MELCA\_id}

**Parameters**

Name	Description
Sub_Catchment_MELCA_id * required	string (path)
10	

**Responses**

Curl

```
curl -X 'GET' \
  'http://127.0.0.1:5050/Hydrographies/potable_water_demands/10' \
  -H 'accept: application/json'
```

Request URL

```
http://127.0.0.1:5050/Hydrographies/potable_water_demands/10
```

Server response

Code	Details
200	Response body <pre>[     {         "agriculturalzone": "agriculturalzone",         "alias": "0",         "annualArea": 0,         "annualDemand": 0,         "annualVolume": 0,         "autumnArea": 0,         "benefitedFamilies": 0,         "id": 10,         "initialContract": 0,         "lastEditedDate": "Mon, 21 Sep 2020 00:00:00 GRT",         "lastEditedUser": 0,         "name": "name",         "pointString": "pointString",         "program": "program",         "returnRate": 0,         "source": "source",         "springArea": 0,         "summerArea": 0,         "summerDemand": 0,         "type": "type",         "winterArea": 0,         "winterDemand": 0     } ]</pre> <p><a href="#">Copy</a> <a href="#">Download</a></p>

Figura 7.2: Ejemplo de consulta de uno de los métodos de la API que contiene la base de datos.

POST /CHAMBO Models/Calculate\_resultant\_flows

**Parameters**

Name	Description
<b>payload</b> * required	object (body)
	{ "sub_catchment_id": "10", "initial date": "00-01-01", "final date": "19-12-31" }

**Cancel**

Parameter content type  
**application/json**

**Execute** **Clear**

**Responses**

Response content type **application/json**

**Curl**

```
curl -X 'POST' \
'http://127.0.0.1:5000/CHAMBO Models/Calculate_resultant_flows' \
-H 'accept: application/json' \
-H 'Content-Type: application/json' \
-d '{
    "sub_catchment_id": "10",
    "initial date": "00-01-01",
    "final date": "19-12-31"
}'
```

**Request URL**

**http://127.0.0.1:5000/CHAMBO Models/Calculate\_resultant\_flows**

**Server response**

Code	Details
200	Response body

```
{
  "demanda_ecosistematica": {
    "annualFlow": 16.347974441550782,
    "autumnFlow": 17.20138569022858,
    "springFlow": 6.8443053188138485,
    "summerFlow": 8.964003131524498,
    "winterFlow": 16.162182675649404
  },
  "demandas": {
    "annualFlow": 28.669999999999995,
    "autumnFlow": 28.669999999999995,
    "springFlow": 28.669999999999995,
    "summerFlow": 28.669999999999995,
    "winterFlow": 28.669999999999995
  },
  "qprec": {
    "annualFlow": 66.710773632109398,
    "autumnFlow": 92.22723268324454,
    "springFlow": 57.4777439591494,
    "summerFlow": 50.52073661668714,
    "winterFlow": 86.63525202812987
  },
  "resultado_final": {
    "annualFlow": 42.322732632109398,
    "autumnFlow": 59.37721205324454,
    "springFlow": 13.0371439591494,
    "summerFlow": 26.08073661668714,
    "winterFlow": 62.19525202812987
  }
}
```

**Download**

Figura 7.3: Ejemplo de consulta del método que calcula el balance hídrico de la cuenca.

## Bibliografía

- [1] Gutierrez, J. M., R. Cano, A. S. Cofino y C. M. Sordo: *Redes Probabilísticas y Neuronales en la Ciencias Atmosféricas*. 2004.
- [2] Olah, C.: *Understanding LSTM networks*. <https://colah.github.io/posts/2015-08-Understanding-LSTMs/>.
- [3] Blöschl, G. y Murugesu Sivapalan: *Scale issues in hydrological modelling: A review*. Hydrological Processes, 9:251 – 290, Abril 1995.
- [4] Razavi, Tara y Paulin Coulibaly: *Streamflow Prediction in Ungauged Basins: Review of Regionalization Methods*. Journal of Hydrologic Engineering, 18:958–975, Agosto 2013.
- [5] Hrachowitz, Markus, Hubert Savenije, G. Blöschl, Jeffrey McDonnell, Murugesu Sivapalan, John Pomeroy, Berit Arheimer, T. Blume, M.P. Clark, Uwe Ehret, F. Fenicia, J.E. Freer, Alexander Gefan, H.V. Gupta, D. Hughes, Rolf Hut, Alberto Montanari, Saket Pande, Doerthe Tetzlaff y Christophe Cudennec: *A decade of Predictions in Ungauged Basins (PUB)—a review*. Hydrological Sciences Journal/Journal des Sciences Hydrologiques, 58:1–58, Junio 2013.
- [6] Blöschl, G., Murugesu Sivapalan, Thorsten Wagener, Alberto Viglione y H. Savenije: *Runoff Prediction in Ungauged Basins. Synthesis Across Processes, Places and Scales*. Runoff Prediction in Ungauged Basins: Synthesis Across Processes, Places and Scales, páginas 1–465, Enero 2011.
- [7] Nearing, G. S., F. Kratzert, A. K. Sampson, C. S. Pelissier, D. Klotz, J. M. Frame, C. Prieto y H. V. Gupta: *What Role Does Hydrological Science Play in the Age of Machine Learning?* Water Resources Research, 57(3):e2020WR028091, 2021.
- [8] Blöschl, Günter, Marc Bierkens, Antonio Chambel, Christophe Cudennec, Georgia Destouni, Aldo Fiori, James Kirchner, Jeffrey McDonnell, Hubert Savenije, Murugesu Sivapalan, Christine Stumpf, Elena Toth, Elena Volpi, Gemma Carr, Claire Lupton, Luis Salinas, Borbála Széles, Alberto Viglione, Hafzullah Aksoy y Yongqiang Zhang: *Twenty-three Unsolved Problems in Hydrology (UPH) – a community perspective*. Hydrological Sciences Journal/Journal des Sciences Hydrologiques, 64:1141–1158, Junio 2019.

- [9] Mizukami, Naoki, Oldrich Rakovec, A. Newman, Martyn Clark, A. Wood, Hoshin Gupta y Rohini Kumar: *On the choice of calibration metrics for “high-flow” estimation using hydrologic models.* Hydrology and Earth System Sciences, 23:2601–2614, Junio 2019.
- [10] Peters-Lidard, Christa, Martyn Clark, Luis Samaniego, Niko Verhoest, Tim van Emmerik, Remko Uijlenhoet, Kevin Achieng, T. Franz y Ross Woods: *Scaling, Similarity, and the Fourth Paradigm for Hydrology.* Hydrology and Earth System Sciences, 21:3701–3713, Julio 2017.
- [11] Kratzert, Frederik, Daniel Klotz, Mathew Herrnegger y Sepp Hocheiter: *A glimpse into the Unobserved: Runoff simulation for ungauged catchments with LSTMs*, Diciembre 2018.
- [12] Georgakakos, Konstantine: *A Generalized Stochastic Hydrometeorological Model for Flood and Flash-Flood Forecasting: 1. Formulation.* Water Resources Research - WATER RESOUR RES, 22, Diciembre 1986.
- [13] Hameed, Maysoun: *Evaluating Global Sensitivity Analysis Methods for Hydrologic Modeling over the Columbia River Basin*, Abril 2015.
- [14] Kratzert, F., D. Klotz, G. Shalev, G. Klambauer, S. Hocheiter y G. Nearing: *Towards learning universal, regional, and local hydrological behaviors via machine learning applied to large-sample datasets.* Hydrology and Earth System Sciences, 23(12):5089–5110, 2019. <https://hess.copernicus.org/articles/23/5089/2019/>.
- [15] <https://www.ecmwf.int/en/forecasts/datasets/reanalysis-datasets/era5>.
- [16] <https://chelsa-climate.org/>.
- [17] Molnar, Peter y Paolo Burlando: *Preservation of rainfall properties in stochastic disaggregation by a simple random cascade model.* Atmospheric Research, 77:137–151, Septiembre 2005.
- [18] Estévez, Javier, Pedro Gavilán y Juan Giraldez: *Guidelines on validation procedures for meteorological data from automatic weather stations.* Journal of Hydrology, 402:144–154, Mayo 2011.
- [19] Ministerio del Ambiente, Agua y Transición Ecológica. <https://www.ambiente.gob.ec/>.

- [20] Rodríguez Ros, Javier: *Aportes a la planificación para la gestión integral de los recursos hídricos Contribución del Comité de la SubCuenCA del río ChAmbo*, Julio 2015.
- [21] <http://modsim.enr.colostate.edu/>.
- [22] Bertsekas, Dimitri, Paul Tseng y Massachusetts Systems: *RELAX-IV : a faster version of the RELAX code for solving minimum cost flow problems*. Septiembre 2022.
- [23] Ruder, Sebastian: *An overview of gradient descent optimization algorithms*, 2016. <https://arxiv.org/abs/1609.04747>.
- [24] Kingma, Diederik P. y Jimmy Ba: *Adam: A Method for Stochastic Optimization*, 2014.
- [25] Pedregosa, Fabian, Gael Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, Jake Vanderplas, Alexandre Passos, David Cournapeau, Matthieu Brucher, Matthieu Perrot, Edouard Duchesnay y Gilles Louppe: *Scikit-learn: Machine Learning in Python*. Journal of Machine Learning Research, 12, Enero 2012.
- [26] Javier Diez-Sierra, Manuel del Jesus: *Long-term rainfall prediction using atmospheric synoptic patterns in semiarid climates with statistical and machine learning methods*. 2020.
- [27] Abdi, Hervé y Lynne J. Williams: *Principal component analysis*. WIREs Computational Statistics, 2(4):433–459, 2010. <https://wires.onlinelibrary.wiley.com/doi/abs/10.1002/wics.101>.
- [28] Moriasi, Daniel, Jeff Arnold, Michael Van Liew, Ron Bingner, R.D. Harmel y Tamie Veith: *Model Evaluation Guidelines for Systematic Quantification of Accuracy in Watershed Simulations*. Transactions of the ASABE, 50, Mayo 2007.
- [29] Mizukami, Naoki, Martyn Clark, A. Newman, A. Wood, Ethan Gutmann, Bart Nijssen, Oldrich Rakovec y Luis Samaniego: *Toward seamless large domain parameter estimation for hydrologic models*. Water Resources Research, 53, Agosto 2017.
- [30] *Flask-RESTful's User Guide*. <https://flask-restful.readthedocs.io/en/latest/>.
- [31] *PostgreSQL*. <https://www.postgresql.org/>.