'Covid-19 LSTM prediction in Spain'

FELIX ANGEL MARTINEZ MUELA* and MIGUEL DE LA CAL BRAVO*, Universidad de Castilla La Mancha, Spain

Proyecto para la asignatura del Máster en Ingeniería Informática llamada Desarrollo de Sistemas Inteligentes, en la cual, se empleará *Matlab* para realizar predicciones de la evolución del coronavirus *Covid-19* a corto plazo mediante la técnica *LSTM*.

CCS Concepts: • **Computing methodologies** → **Machine learning**; **Machine learning algorithms**.

Additional Key Words and Phrases: Prediction, Covid-19, LSTM, Spain

ACM Reference Format:
Felix Angel Martinez Muela and Miguel de la Cal Bravo. 2020. 'Covid-19 LSTM prediction in Spain'. *ACM Trans. Graph.* 0, 0, Article 0 (May 2020), 4 pages.

1 INTRODUCCIÓN

En este trabajo se realiza una serie de predicciones sobre la evolución del nuevo coronavirus *Covid-19* en las distintas comunidades autónomas de España. Las predicciones serán realizadas a corto plazo, a un día vista y siete días vista.

Para ello, se empleará la técnica *LSTM* sobre cada una de las CCAA, ajustando sus parámetros con la finalidad de reducir los errores y obtener predicciones de las diferentes variables lo más cercanas a la realidad posible.

Finalmente, las predicciones obtenidas de todas las CCAA para el periodo de días entre el 15 de abril y 30 de abril, serán exportadas a ficheros *.csv*.

2 LSTM

LSTM proviene de las siglas en Inglés: *Long Short-Term Memory*, lo que nos da una idea de que se trata de un método de inteligencia artificial que tiene en cuenta situaciones cercanas, teniendo en cuenta también las situaciones más alejadas del presente y que también tendrán influencia en el resultado.

En este apartado, se explicará brevemente cómo utilizar la técnica **Long Short-Term Memory (LSTM)** para realizar predicciones a corto plazo sobre la evolución del virus.

LSTM como podemos apreciar en la imagen 2 , se trata de una Red Neuronal Recurrente con propagación hacia atrás, lo cual implica que las neuronas de capas anteriores se ven afectadas por las neuronas posteriores. Esto último se realiza para que la red pueda recordar estados previos y poder decidir cuál será el siguiente.

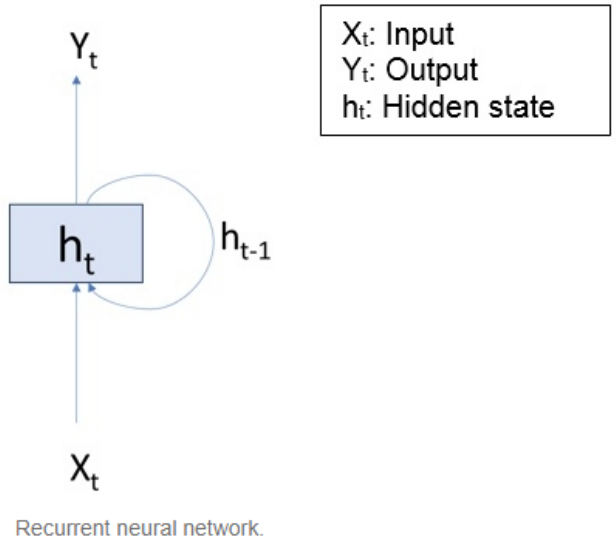

Fig. 1. Redes Neuronales Recurrentes o RNN

A diferencia de las redes neuronales estándar, las *LSTM* puede aprender dependencias largas, por lo que será buenas para predecir a largo plazo. Como podemos apreciar en la imagen 2, nos encontramos con la estructura de un bloque *LSTM*.

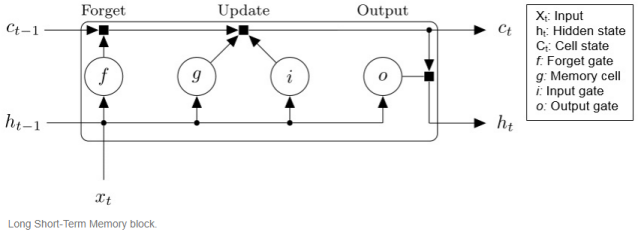

Fig. 2. Bloque LSTM

Los pesos de la puerta de entrada controlan como influye un nuevo valor en la celda. Los pesos de la puerta del olvido y la puerta de salida, controlan como un valor permanece en la celda y el valor que se le da para calcular la activación del bloque *LSTM*.

2.1 Variables a predecir

En este trabajo, se predecirán un total de cinco variables diferentes, siendo éstas las siguientes:

*Both authors contributed equally to this research.

Authors' address: Felix Angel Martinez Muela, FelixAngel.Martinez@alu.uclm.es; Miguel de la Cal Bravo, Miguella.Cal@alu.uclm.es, Universidad de Castilla La Mancha, Ciudad Real, Ciudad Real, Spain, 13005.

0730-0301/2020/5-ART0 $15.00
https://doi.org/

- `casos`: número de casos detectados en total.
- `altas`: número de altas epidemiológicas totales.
- `hospitalizados`: número de personas hospitalizadas en total.
- `uci`: número de personas hospitalizadas en la UCI.
- `defunciones`: número de defunciones en total.

2.2 Explicación de la técnica

Una vez conocemos las variables que trataremos de predecir, realizaremos una explicación con los fundamentos de la técnica que emplearemos para predecir dichas variables. Recordemos antes de nada, que se lo que se pretende es realizar predicciones a corto plazo, para lo cual *LSTM* se trata de una buena técnica para llevar este proceso a cabo.

2.3 Hiperparámetros

Se denomina hiperparámetros, a aquellos valores que se asignan antes de comenzar el aprendizaje, en nuestro caso nos centraremos en los hiperparámetros que afectan a *LSTM*, que son:

- `Optimizador`: se utilizará *Adam* como optimizador del algoritmo.
- `MaxEpochs`: número máximo de etapas para el entrenamiento de la red.
- `InitialLearnRatel`: factor de aprendizaje inicial de la red.
- `LearnRateSchedule`: ajuste de la tasa de aprendizaje durante el entrenamiento.
- `LearnRateDropPeriod`: número de épocas para disminuir la tasa de aprendizaje.
- `LearnRateDropFactor`: factor para reducir la tasa de aprendizaje.

En las secciones siguientes, hablaremos de estos hiperparámetros y cuales seleccionaremos para su optimización en el algoritmo.

2.3.1 Selección Hiperparámetros. Para saber que parámetros son los mejores para nuestro modelo, teniendo en cuenta que tenemos un total de diecinueve comunidades a predecir cinco variables por cada una.

Para obtener cuales son los mejores hiperparámetros se han realizado una serie de algoritmos, llamados: **OptimizacionHiperparametros.m** y **LSTM_Hyperparam.m**.

- `OptimizacionHiperparametros.m`: Clase en *Matlab* para probar un conjunto definido de iteracciones de entrenamiento y de ratio de aprendizaje, de esta manera se va a ejecutar mediante fuerza bruta un conjunto de iteracciones y de ratios de aprendizaje sobre los datos de las comunidades autónomas, llamando a *LSTM_Hyperparam.m*, para obtenido la predicción de dichos días, comprobar el error de esta. Se ejecutará para los valores determinados, y nos quedaremos con el mejor valor, tanto de iteracciones de entrenamiento, como de nivel de aprendizaje.
- `LSTM_Hyperparam.m.`: se trata de una clase *Matlab*, muy similar a *LSTM.m* la cual recibe también como parámetros de entrada los valores a probar, mencionados en *Optimización-Hiperparametros.m.*

En la imagen 3 tenemos como se calcula el error para una variable dentro de una comunidad autónoma, y para el cálculo de siete días. Esto es parte del la obtención de la selección de hiperparámetros.

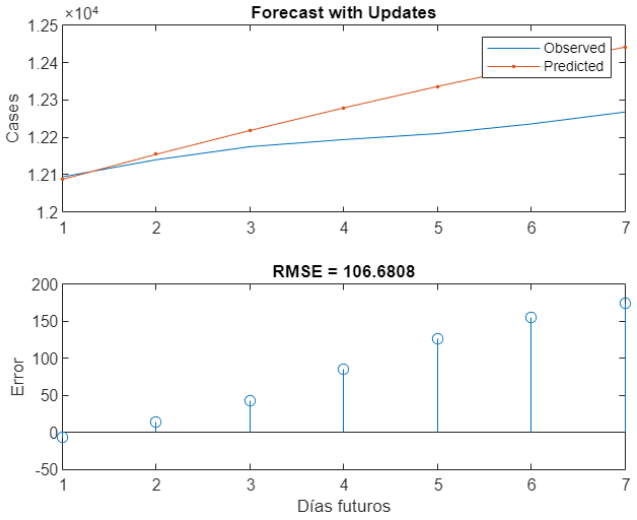

Fig. 3. Gráficas con el error cometido en las predicciones

Los valores de los hiperparámetros seleccionados finalmente para realizar las predicciones son los que se muestran en la Tabla 1:

Table 1. Valores elegidos para los hiperparámetros de LSTM

Hiperparámetro	Nombre variable
Optimizador	*adam*
MaxEpochs	**140**
GradientThreshold	1
InitialLearnRate	0.005
LearnRateSchedule	*piecewise*
LearnRateDropPeriod	5
LearnRateDropFactor	0.05

2.4 Entrenamiento

Una de las etapas que debemos realizar con nuestra red neuronal recurrente *LSTM* es entrenarla. Entrenarla en este caso consiste en dadas unas fechas en las que ha sucedido el *Covid-19*, con unos datos resultantes del mismo, darle esos datos a la red sabiendo el resultado de éstos. Esta parte está definida en la clase *Matlab LSTM.m*

2.5 Validación

La validación consiste en una vez que hemos entrenado la red con los datos de entrenamiento, procedemos a ver cómo de buena es nuestra red prediciendo valores que ya conocemos su resultado, pero que no han sido empleados para el entrenamiento. La validación ha resultado útil para la obtención de unos hiperparámetros óptimos, ya que se ha comparado el resultado predicho, con los datos reales y así poder calcular un error medio cuadrático.

3 RESULTADOS OBTENIDOS

Tras aplicar la técnica de *LSTM* y haber extraído los resultados de cada una de las variables para cada comunidad autónoma, pasaremos a analizar los datos resultantes de las predicciones.

Debemos tener en cuenta que la obtención de los hiperparámetros óptimos, es una ardua tarea computacionalmente, y para la cual, nosotros como alumnos de máster no podemos soportar. Para suplir esta deficiencia se ha optado por el uso de *Matlab Online*, ya que tiene mayor capacidad de cómputo que nuestros portátiles.

La parte de ejecución de *LSTM* también ha supuesto una carga computacional alta, ya que es un algorítmo pesado y que tarda alrededor de 5 minutos para la predicción en un día y para todas las comunidades autónomas, por lo que para calcular las 16 predicciones, lleva en un tiempo estimado de 1 hora y 15 minutos.

Los principales resultados obtenidos han sido el conjunto de los 15 días predichos a partir del 16 de abril en adelante, en formato .csv como se muestra en la imagen 4.

	A	B	C	D	E	F	G
1	CCAA	Fecha	AcumulatedPRC	Hospitalized	Critical	Deaths	AcumulatedRecoveries
2	AN	16/04/2020	110.594.639	51.930.918	6.939.726	10.310.784	33.070.088
3	AN	17/04/2020	111.942.334	5239.46	7.050.388	10.703.201	34.375.454
4	AN	18/04/2020	113.432.246	52.788.384	7.155.768	11.100.376	3.547.019
5	AN	19/04/2020	114.949.951	53.123.491	7.254.351	11.484.025	36.305.752
6	AN	20/04/2020	116.448.213	53.414.946	7.346.106	11.842.905	3.692.802
7	AN	21/04/2020	117.899.492	53.672.866	7.431.347	12.171.123	37.387.341
8	AN	22/04/2020	119.288.838	53.904.277	7.510.507	12.466.304	37.727.183
9	AR	16/04/2020	42.566.753	1.953.465	2.914.276	6.102.367	11.470.585
10	AR	17/04/2020	42.967.344	19.717.783	2.943.144	6.360.695	1.183.793
11	AR	18/04/2020	43.212.891	19.797.006	2.969.471	6.618.216	12.198.717
12	AR	19/04/2020	43.349.795	19.814.153	2.993.788	6.866.129	12.541.725

Fig. 4. Ejemplo de predicción de datos exportados a csv

En la figura 5 nos encontramos como se mezclan los datos obtenidos, con respecto los predichos, pudiéndonos dar cuenta de que el patrón predicho puede ir muy de la mano del futuro que se dará para esas fechas.

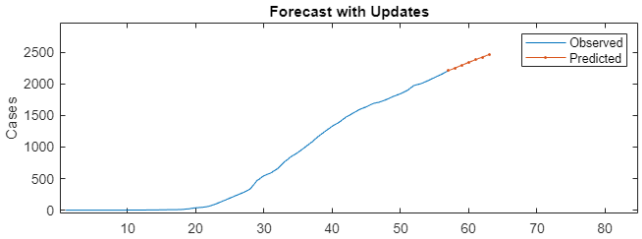

Fig. 5. Ejemplo predicción

4 CONCLUSIONES

Para terminar, daremos nuestras conclusiones y opinión personal sobre la técnica de predicción utilizada. Por un lado, nos parece que se trata de una técnica que a corto plazo (uno a siete días vista) puede funcionar bastante bien, ya que realiza predicciones de forma muy monótona con respecto a lo observado (sin generar picos o dientes de sierra). Sin embargo, a medida que queramos predecir datos a más largo plazo, detectamos que no generará predicciones tan ciertas conociendo la evolución normal que tiene una pandemia, debido al comportamiento descrito anteriormente.

Por otro lado, y al tener que predecir bastantes variables para un total de diecinueve CCAA dieciséis días diferentes, los tiempos de cómputo para predecir los datos serán muy elevados si no disponemos de un computador con muy buenos recursos. Además, gracias al código de nuestro *script LSTM_Hyperparam.m*, podemos realizar diferentes combinaciones de los hiperparámetros para obtener los mejores resultados, eso sí, aumentando significativamente de nuevo los tiempos de cómputo para predecir los datos.

Este *script*, en caso de tener un computador muy potente, creemos que sería capaz de sacar las mejores predicciones con la técnica *LSTM* sin invertir tanto tiempo, y así de esta manera podríamos probar un amplio abanico de combinaciones con sus hiperparámetros y obtener la mejor de todas ellas.

5 REFERENCIAS

A continuación, se muestran las referencias empleadas para realizar tanto el desarrollo del proyecto, como la presente memoria:

- Temario asignatura Desarrollo de Sistemas Inteligentes, perteneciente al Máster en Ingeniería Informática de la UCLM
- https://es.mathworks.com/help/deeplearning/ug/long-short-term-memory-networks.html
- https://es.mathworks.com/discovery/lstm.html
- https://covid19.esi.uclm.es/model?model=lstm
- https://www.bioinf.jku.at/publications/older/2604.pdf
- https://es.mathworks.com/help/deeplearning/ug/time-series-forecasting-using-deep-learning.html
- https://towardsdatascience.com/learning-rate-schedules-and-adaptive-learning-rate-methods-for-deep-learning-2c8f433990d1
- https://es.mathworks.com/help/deeplearning/ref/trainingoptions.html

A EJECUCIÓN

El proyecto ha sido realizado en lenguaje *Matlab*, para el cual se han desarrollado los siguientes scripts y funciones:

(1) **main.m**
(2) **OptimizacionHiperparametros.m**
(3) **LSTM_Hyperparam.m**
(4) **HistoricDataSpain.m**
(5) **LSTM.m**

En el primer fichero que ejecutaremos, llamado **main.m**, encontramos un *main* para llamar a las funciones *HistoricDataSpain* y realizar posteriormente las predicciones con la técnica para el intervalo de días del 15 de abril (indicado como 0 en el bucle) y el 30 de abril (indicado como 15 en el bucle). Estas predicciones serán realizadas para las cinco variables, en las diecinueve CCAA de España para cada uno de los días hasta siete días vista, exportando finalmente los resultados predichos en cada día a un fichero *.csv*.

En el segundo fichero llamado **OptimizacionHiperparametros.m**, se realizan pruebas combinando los parámetros de épocas y learning rate, con el fin de lograr la mejor parametrización de la técnica *LSTM* y obtener mejores predicciones.

En el tercer fichero llamado **LSTM_Hyperparam.m**, se realizan pruebas combinando los parámetros de épocas y learning rate, con el

fin de lograr la mejor parametrización de la técnica *LSTM* y obtener mejores predicciones.

Los ficheros **HistoricDataSpain.m** y **LSTM.m** han sido obtenidos de la asignatura en *Campus Virtual*. El primero de ellos se encarga de descargar los datos publicados en el *ISCII* y formatearlos para su posterior tratamiento, mientras que el segundo se encarga de aplicar la función *LSTM* con los parámetros seleccionados.

Es importante recordar que, los parámetros de **MaxEpochs** y **InitialLearnRate** han sido modificados del fichero **LSTM.m**, en función de nuestras pruebas realizadas a modo de prueba y error, quedándonos con aquellas con los hiperparámetros con los que obtuvimos menor error.