Red Neural Recurrente para la Desambiguación de Entidades en Datos de Medios Sociales

Cristina Zuheros, Siham Tabik, Ana Valdivia, Eugenio Martínez-Cámara y Francisco Herrera
Instituto Andaluz de Investigación en Ciencias de Datos e Inteligencia Computacional,
Universidad de Granada, 18071 Granada, España
Email: {czuheros, siham, avaldivia}@ugr.es, {emcamara, fherrera}@decsai.ugr.es

***Resumen*—Un reto particular en el ámbito del Procesamiento del Lenguaje Natural es la desambiguación de palabras polisémicas. La gran disponibilidad, diversidad y rapidez cambiante de los datos en línea fuerzan el desarrollo de sistemas de desambiguación con una dependencia reducida de los recursos lingüísticos. Se parte de la base que la codificación del contexto de una entidad específica no requiere del uso de recursos lingüísticos externos como bases de conocimiento, por lo que se propone una arquitectura de red neuronal basada en el uso de las Redes Neuronales Recurrentes *Long Short-Term Memory* para codificar el contexto de una entidad, concretamente el modelo *Two k-Contextual Windows*. Se considera un problema real que requiere la desambiguación de la entidad *Granada*. Se genera y proporciona un corpus etiquetado de comentarios extraidos de medios sociales escritos en inglés, los cuáles se utilizan para evaluar nuestra propuesta. Los resultados demuestran la validez de nuestra hipótesis.**

I. Introducción

La mayoría de las palabras tienen más de un significado posible ya que el lenguaje humano es inherentemente ambiguo. Los humanos somos capaces de identificar el sentido de una palabra analizando el contexto en el que esta aparece. Actualmente, una enorme cantidad de información, incluyendo textos, imágenes y vídeos, está disponible en distintas plataformas de Internet como son *Twitter*, *Instagram*, *Tripadvisor* y *Booking*, entre otros [1] y [2]. La gran cantidad de información que puede obtenerse de esas fuentes es muy valiosa en distintos campos. Por ejemplo, saber lo que se piensa sobre una entidad, un producto, un evento, una persona o un lugar, nos permite usar dicha información para mejorar un servicio y aumentar el número de clientes o turistas. Extraer esta información sobre una entidad con un significado específico de una enorme cantidad de datos requiere filtrar aquellas menciones que hacen referencia a otros posibles significados de la entidad en cuestión.

La desambiguación del sentido de la palalabra (*Word Sense Disambiguation* - WSD) es la tarea del Procesamiento del Lenguaje Natual que se encarga de la asignación computacional del sentido o significado correcto de una palabra dependiendo de su contexto [3] y [4]. Esta tarea requiere una gran cantidad de información y conocimiento. Actualmente, se vuelve cada vez más desafiante, ya que los sentidos de las palabras son muy diversos e incluso cambiantes, especialmente en escenarios donde la vinculación de un nuevo sentido a una entidad es impredecible. Además, los sistemas supervisados de WSD dependen altamente de las características anotadas a mano, las cuales también se ven afectadas por el problema anteriormente mencionado. Por tanto, la tarea de desambiguación se ve limitada por el cuello de botella de la adquisición de conocimiento y la naturaleza siempre cambiante del lenguaje.

El estado del arte en WSD queda determinado principalmente por modelos supervisados y actualmente por métodos de aprendizaje profundo basados en un tipo específico de redes neuronales recurrentes conocido como modelo *Long Short-Term Memory* (LSTM), como el modelo *Targeted Two k-Contextual Windows* [5]. Sin embargo, esos modelos neuronales no han sido evaluados en *corpora* obtenidos de medios sociales. Este tipo de datos presentan unas características propias como el uso informal del lenguaje y el uso de *emojis* para la expresión de emociones [6]. Estas características específicas hacen más desafiante la tarea.

Las contribuciones que se hacen en este trabajo se pueden resumir en: (1) nueva arquitectura neuronal basada en LSTM para la desambiguación de entidades, llamada *Two k-Contextual Windows*. Consideramos como caso de estudio la desambiguación de la entidad *Granada*. Y (2) nuevo corpus etiquetado manualmente cada uno de los sentidos de la entidad objeto de estudio. El corpus está compuesto por textos publicados en *Twitter* e *Instagram*.

Dicho documento está estructurado de la siguiente manera: la Sección II abarca la definición de WSD como tarea de clasificación y un breve análisis sobre los trabajos más cercanos. La descripción del modelo propuesto se muestra en la Sección III, y la generación del corpus utilizado para la evaluación se presenta en la Sección IV. El marco experimental se muestra en la Sección V, mientras que los resultados experimentales se presentan en la Sección VI. Por último, en la Sección VII se presentan las conclusiones y los trabajos futuros.

II. Contexto

En esta Sección se exponen las bases para entender nuestra propuesta. En la Sección II-A se define la tarea de clasificación de WSD junto con los *word embeddings*. En la Sección II-B se presenta la LSTM RNN y en la Sección II-C se revisan los trabajos más relacionados.

II-A. Aprendizaje de secuencias para WSD y embeddings

WSD puede definirse como la identificación automática del sentido más adecuado de una palabra a la misma en función de su contexto. Puede verse como una tarea de clasificación.

Sea T una secuencia de n palabras $\{w_1, \ldots, w_n\}$, la tarea de desambiguación consiste en encontrar una función A de palabras a sentidos, tal que $A(w_j) \subseteq sentido_D(w_j)$ siendo $sentido_D(w_j)$ el conjunto de sentidos posibles de la palabra (w_j) en un diccionario de sentidos D.

En este trabajo, T es el conjunto de comentarios extraídos de *Twitter* e *Instagram*, la palabra a desambiguar w_t es *Granada* y D es el conjunto de posibles significados que se detallan en la Sección IV-A. Se propone representar mediante vectores de *word embeddings* la secuencia de palabras $\{w_1, \ldots, w_n\}$ de cada secuencia de entrada T y codificarlas posteriormente con una LSTM RNN.

Un *embedding* es una representación de un objeto topológico en un espacio determinado, de forma que se conservan sus propiedades de conectividad o algebraicas [7] y [8]. Un vector *word embedding* es la representación del espacio semántico ideal de palabras en un espacio vectorial continuo. Matemáticamente, se define un *word embedding* w_j, como un vector d-dimensional $\mathbf{we}_j^\mathsf{T} = (we_1, \ldots, we_d) \in \mathbb{R}^d$. Por tanto, se define $WE_{1:n} \in \mathbb{R}^{d \times n}$ como la matriz de *word embeddings* del conjunto de palabras $\{w_1, \ldots, w_n\}$. Algunos de los modelos más destacados para realizar *word embeddings* son *word2vec* [9] y *GLoVe* [10].

II-B. Red neuronal de memoria de corto y largo plazo

LSTM es un tipo de RNN con la capacidad de representar entradas secuenciales de tamaño arbitrario en un vector de tamaño fijo y prestar atención a las propiedades estructuradas de las entradas [11]. Dado que se pretende desambiguar una palabra en una entrada secuencial de tamaño arbitrario, LSTM cumple con los requisitos de nuestro problema.

A continuación, se definen las bases de RNN que también son la base de LSTM. Una RNN se define como una función R que se aplica recursivamente a una secuencia de palabras, concretamente en nuestro caso una secuencia de *word embeddings* $(\mathbf{we}_1, \ldots, \mathbf{we}_n)$. La función R toma como entrada un vector de estado $\mathbf{s}_{j-1}$ y un vector $\mathbf{we}_j$ y genera un nuevo vector estado $\mathbf{s}_j$. Este vector estado es proyectado en un vector salida $\mathbf{y}_j$ mediante una función determinística $O(\cdot)$. La Ecuación 1 resume dicha definición.

$$\begin{aligned} RNN(WE_{1:n}; \mathbf{s}_o) &= \mathbf{y} \\ \mathbf{y}_j &= O(\mathbf{s}_j) \\ \mathbf{s}_j &= R(\mathbf{s}_{j-1}, \mathbf{we}_j) \end{aligned} \tag{1}$$

Las RNNs presentan el problema del desvanecimiento del gradiente [12] y [13], lo que significa que la multiplicación repetida de parámetros en la función R puede hacer que los valores de esos parámetros se desvanezcan o exploten, lo que hace más difícil el entrenamiento de la red neuronal. LSTM fue la primera RNN con una arquitectura diseñada para resolver dicho problema [12] y [13].

LSTM [14] separa el vector estado (s_j) en dos mitades, siendo una mitad considerada como "celdas de memoria" y la otra como memoria de trabajo. El mecanismo de LSTM decide en cada entrada de la secuencia qué cantidad de la nueva entrada (we_j) debe escribirse en la celda de memoria y qué cantidad del contenido de la celda de memoria debe olvidarse [15].

Desde un punto de vista lingüístico, la operación recursiva de LSTM significa que cada palabra (w_j) está codificada con el significado de las palabras anteriores ($R_{LSTM}(\mathbf{s}_{j-1}$, $\mathbf{we}_j)$). Sin embargo, el contexto de una palabra no sólo depende de las palabras localizadas antes de esa palabra, sino también en las palabras que se encuentran después de ella en la oración. Las LSTMs son versátiles, pueden funcionar en direcciones diferentes y sus salidas pueden combinarse de maneras diferentes. Formalmente, dado un vector de *word embeddings* $(\mathbf{we}_1, \ldots, \mathbf{we}_n)$, y una palabra a desambiguar, $\mathbf{we}_t$, una LSTM puede codificar las palabras de la secuencia de entrada tanto de izquierda a derecha ($\overrightarrow{\text{LSTM}}$) como de derecha a izquierda ($\overleftarrow{\text{LSTM}}$), como indican las flechas. Las salidas de ambas LSTMs pueden ser combinadas y procesadas por una o más capas. Por tanto, codificar el significado de una secuencia de palabras puede ser expresado mediante una función $G(\cdot)$ que combina la salida de un número m de redes LSTMs (ver Ecuación 2).

$$\text{meaning} = G(\text{LSTM}_{1:m}) \tag{2}$$

II-C. Trabajos afines

Los principales enfoques que se encuentran en la amplia literatura [3] y [4] sobre WSD pueden clasificarse en cuatro grupos: (1) Métodos basados en el conocimiento, (2) no supervisados, (3) semisupervisados, y (4) supervisados. Nos centramos en la categoría supervisada.

Aunque los modelos supervisados requieren que el sentido de los datos sea manualmente etiquetado por humanos expertos, superan a los sistemas basados en conocimiento en los *bechmarks* estándar [16]. Se basan en el hecho de que palabras semánticamente similares tienden a tener distribuciones contextuales similares [17], [18] y [19]. Gran cantidad de modelos utilizan características manuales y métodos tradicionales de aprendizaje automático, como el sistema IMS [20] y [21].

Recientemente, los métodos de aprendizaje profundo están mostrando resultados muy prometedores [22], [23], [24], [25] y [26]. Melamud et al. [27] presentaron un modelo no supervisado basado en LSTM bidireccional. En el trabajo [28] utilizaron una arquitectura similar de red neuronal pero bajo un enfoque semi-supervisado para codificar el contexto de la palabra a desambiguar. En el trabajo [5] utilizaron una arquitectura similar combinada con capas ocultas bajo un enfoque supervisado, logrando una alta precisión en la tarea de desambiguación léxica de las primeras ediciones de Senseval [29] y [30].

III. Modelo neuronal para WSD

En la Sección III-A, se detalla el modelo LSTM basado en una arquitectura similar a la utilizada en [5]. En la Sección III-B, se presenta nuestra propuesta.

Ambos modelos[1] se basan en la codificación de una ventana contextual, que definimos como un número fijo de k palabras

[1] Sentencia mostrada en las Figuras 1 y 2: "La ciudad de Granada es muy bonita en invierno."

consecutivas de la secuencia de entrada. Además, comparten la capa de entrada y representación. La capa de entrada es definida como una secuencia de n palabras, $\{w_1, \ldots, w_n\}$. La capa de representación es una matriz de *word embeddings*, definida como $WE_{1:n}$. Se usa el conjunto de pre-entrenado de vectores de *word embeddings* 6B *GloVe* [10]. Se aplica *Dropword* como método de regularización.

Los modelos difieren en: (1) la dirección en la que el modelo procesa las palabras de cada ventana contextual, y (2) la posición de la palabra en la sentencia, a partir de la cual el modelo inicia el procesamiento de cada ventana de contexto.

III-A. Targeted Two k-Contextual Windows

El modelo *Targeted Two k-Contextual Windows*, al que nos referiremos como $\overrightarrow{CW^{t-k}}\overleftarrow{CW^{t+k}}$ tiene una arquitectura similar al modelo presentado en [5]. Determina las k-palabras de la ventana de contexto basándose en la posición de la palabra a desambiguar w_t. La primera ventana de contexto incluye las k-palabras situadas a la izquierda de la palabra a desambiguar mientras que la segunda ventana considera las k-palabras situadas a la derecha de w_t (ver Figura 1).

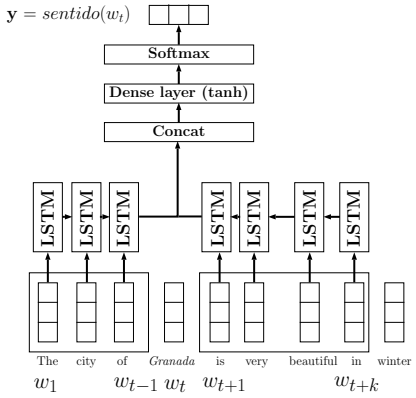

Figura 1. Modelo Targeted Two k-Contextual Windows considerando $k = 4$.

Las dos ventanas de contexto son procesadas por dos LSTM con d_{LSTM} unidades ocultas. El último vector de estado de las dos LSTMs son concatenadas y procesadas por una capa completamente conectada con h_{dense} unidades ocultas activadas mediante una función tangente hiperbólica (tanh). Finalmente, una capa *softmax* calcula la distribución de probabilidad correspondiente al conjunto de sentidos posibles de w_t. La ecuación 3 muestra las operaciones subyacentes.

$$\begin{aligned} sentido(w_t) &= \arg\max_{s \in S}(\mathbf{y}), \mathbf{y} \in \mathbb{R}^s \\ \mathbf{y} &= \text{softmax}(\mathbf{c}), \mathbf{c} \in \mathbb{R}^{h_{dense}} \\ \mathbf{c} &= \text{dense}(\textbf{concat}), \textbf{concat} \in \mathbb{R}^{2*h_{LSTM}} \\ \textbf{concat} &= [\mathbf{m}^1; \mathbf{m}^2], \mathbf{m}^1, \mathbf{m}^2 \in \mathbb{R}^{h_{LSTM}} \\ \mathbf{m}^2 &= \overleftarrow{\text{LSTM}}(WE_{t+1:t+k}), WE_{t+1:t+k} \in \mathbb{R}^{d \times k} \\ \mathbf{m}^1 &= \overrightarrow{\text{LSTM}}(WE_{t-k:t-1}), WE_{t-k:t-1} \in \mathbb{R}^{d \times k} \end{aligned} \tag{3}$$

III-B. Two k-Contextual Windows

Se propone un modelo que, a diferencia del modelo previo (ver Sección III-A), permite codificar las primeras y últimas k-palabras de la secuencia de entrada sin tener en cuenta la posición de w_t y permite solapar las dos ventanas de contexto. El modelo *Two k-Contextual Windows* ($\overrightarrow{CW^k}\overleftarrow{CW^{n-(k-1)}}$) procesa dos ventanas de contexto de k-palabras en direcciones opuestas. La primera analiza la oración hasta la palabra localizada en la posición *k* y la segunda analiza desde la palabra localizada en la posición *n-(k-1)*, siguiendo las direcciones indicadas por las flechas, como muestra la Figura 2.

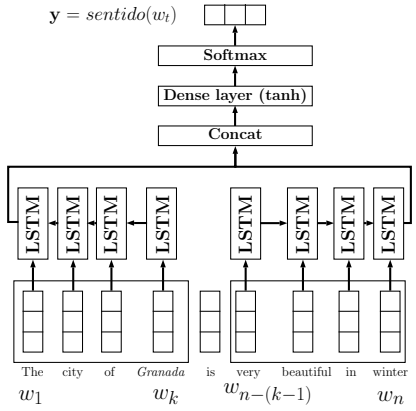

Figura 2. Modelo Two k-Contextual Windows considerando $k = 4$.

Ambas ventanas de contexto son individualmente procesadas usando una LSTM con h_{LSTM} unidades ocultas. Las salidas son concatenadas y procesadas por una capa completamente conectada con h_{dense} unidades ocultas activadas mediante una función tangente hiperbólica. Finalmente, *softmax* proporciona el sentido de la palabra a desambiguar w_t. La ecuación 4 muestra las operaciones realizadas por cada capa.

$$\begin{aligned} sentdo(w_t) &= \arg\max_{s \in S}(\mathbf{y}), \mathbf{y} \in \mathbb{R}^s \\ \mathbf{y} &= \text{softmax}(\mathbf{c}), \mathbf{c} \in \mathbb{R}^{h_{dense}} \\ \mathbf{c} &= \tanh(\textbf{concat}), \textbf{concat} \in \mathbb{R}^{2*h_{LSTM}} \\ \textbf{concat} &= [\mathbf{m}^1; \mathbf{m}^2], \mathbf{m}^1, \mathbf{m}^2 \in \mathbb{R}^{h_{LSTM}} \\ \mathbf{m}^2 &= \overleftarrow{\text{LSTM}}(WE_{1:k}), WE_{1:k} \in \mathbb{R}^{d \times k} \\ \mathbf{m}^1 &= \overrightarrow{\text{LSTM}}(WE_{n-(k-1):n}), WE_{n-(k-1):n} \in \mathbb{R}^{d \times k} \end{aligned} \tag{4}$$

IV. GENERACIÓN DEL CORPUS

En la siguiente Sección se describen los diferentes sentidos de la entidad *Granada* (Sección IV-A), se expone el preprocesamiento de los datos (Sección IV-B) y la anotación del corpus (Sección IV-C).

IV-A. Palabra a desambiguar

Un análisis profundo realizado en diferentes fuentes de Internet ha demostrado que la palabra *Granada* tiene al menos 54 sentidos. Los sentidos más frecuentes pueden agruparse en:

Sentido 1: Provincia/Ciudad/Calle localizada en el sureste de España, evento (no relacionado con música o deporte) o monumento/establecimiento de dicha localización.

Sentido 2: País/Provincia/Ciudad/Distrito/Calle/Río no localizado en España.

Sentido 3: Fruto, granada.

Sentido 4: Referencias a música o deporte.

Sentido 5: Otros significados menos relevantes como usuarios de redes sociales, canales de televisión, vehículos, periódicos, entre otros.

IV-B. Pre-procesamiento del corpus

El conjunto de datos contiene un total de 876.969 comentarios y opiniones que incluyen la palabra *Granada* publicados por usuarios de *Twitter* e *Instagram*. Cada instancia del dataset presenta los campos siguientes:

- *uid*: identificador de la instancia.
- *source*: fuente de la instancia.
- *body*: texto, comentario de la instancia.

La mayoría de las instancias que provienen de medios sociales contienen *emojis*, ya que son útiles para expresar los sentimientos de las personas [31]. Estos sentimientos pueden sugerir si una persona está hablando sobre un tema u otro. Por ejemplo, si una usuaria usa algunos *emojis* mostrando frutas, probablemente no esté hablando sobre deporte. Como los *emojis* no pueden ser procesados directamente por los modelos, se utiliza una descripción con palabras para representar cada *emoji*. Se elabora un diccionario de *emojis* público que contiene un total de 837 *emojis* expresados en inglés[2].

Se eliminan todas URLs a imágenes, dado que el procesamiento de imágenes no forma parte del flujo de trabajo de nuestra propuesta. Se eliminan los comentarios carentes de contenido y los caracteres repetitivos ya que no son útiles para nuestra tarea. Finalmente, se reemplazan caracteres como *<3*, *=)* por "*love*"/"amor", "*I am happy*"/"Estoy feliz", entre otros.

IV-C. Etiquetado del corpus

La primera autora, que tiene un alto dominio del inglés, etiquetó manualmente las instancias del conjunto de datos. Cada instancia se anota con un solo sentido. La Tabla I muestra el número de comentarios por cada sentido. El conjunto de instancias etiquetadas se almacena en un archivos XML.

Sentido 1	Sentido 2	Sentido 3	Sentido 4	Sentido 5	Total
11.985	3.381	452	7.438	1.445	24.701

Tabla I
NÚMERO DE INSTANCIAS ETIQUETADAS PARA CADA SENTIDO.

V. Marco experimental

Se realiza una experimentación profunda con el fin de evaluar nuestra propuesta. En primer lugar, se establecen dos subconjuntos de entrenamiento y test con el objetivo de evaluar nuestro modelo neuronal mediante diferentes distribuciones de los datos. También se evalúa la relevancia del uso de *emojis* como características para la tarea de desambiguación.

Además, se considera XGBoost como caso base, ya que presenta buenos resultados en tareas de clasificación. Dado que se considera WSD como una tarea de clasificación, se utilizan las medidas de evaluación propias de tareas de clasificación para evaluar la conducta de nuestra propuesta. Los detalles de la experimentación son los siguientes.

Partición de los datos Se crean dos particiones diferentes de conjuntos de entrenamiento y de test para evaluar nuestra propuesta: (1) P1, el 80 % de las instancias se encuentran en el conjunto de entrenamiento y el conjunto de test contiene el 20 % restante, y (2) P2, el 70 % de las instancias están en el conjunto de entrenamiento mientras que el 30 % restante están en el test.. El número de elementos de cada conjunto se muestra en el Tabla II.

Emojis Puesto que se considera que los *emojis* pueden ser una característica relevante para la tarea de desambiguación, comparamos resultados de: (1) eliminar los *emojis* de las instancias, y (2) sustituir cada *emoji* por una expresión textual.

Entrenamiento de los modelos Los valores de los hiperparámetros utilizados en los modelos neuronales se muestran en la Tabla III. El entrenamiento se llevó a cabo utilizando la función de pérdida de entropía cruzada y el algoritmo de optimización *Momentum* [32].

Baseline Con el fin de comparar, se aplica el modelo de clasificación *XGBoost* para bolsa de palabras [33]. *XGBoost* es un sistema de clasificación basado en árboles de decisión cuyo entrenamiento se basa en el método de aumento de gradientes. Los textos se *tokenizan* y las palabras se representan mediante su valor *tf-idf*.

Métricas de evaluación Dado que se re-formula la tarea de WSD como un problema de clasificación multiclase, se usan las versiones macro-promediadas de Precisión, Recall y F1.

VI. Resultados y Análisis

En esta sección se describen y analizan los resultados alcanzados para el caso base, XGBoost, y los dos modelos presentados en la Sección III, $\underrightarrow{CW}^{t-k}\underleftarrow{CW}^{t+k}$ y $\underleftarrow{CW}^{k}\underrightarrow{CW}^{n-(k-1)}$. En primer lugar se analiza y compara el rendimiento global de todos los modelos, y posteriormente se estudia el rendimiento por clase del mejor modelo. Finalmente, se analiza el impacto de eliminar o incluir *emojis* y el uso de *GloVe*.

La Tabla IV muestra los resultados alcanzados para los tres modelos indicados para cada partición de los datos incluyendo y sin incluir *emojis*. Los dos modelos neuronales superan al modelo XGBoost y logran resultados muy beneficiosos. Los mejores resultados para todos los modelos se obtienen en la partición P2 incluyendo *emojis*. Las mejores medidas para todas las particiones consideradas son proporcionadas por el modelo $\underleftarrow{CW}^{k}\underrightarrow{CW}^{n-(k-1)}$, seguido por $\underrightarrow{CW}^{t-k}\underleftarrow{CW}^{t+k}$. $\underrightarrow{CW}^{t-k}\underleftarrow{CW}^{t+k}$ alcanza peores resultados ya que considera menos información que $\underleftarrow{CW}^{k}\underrightarrow{CW}^{n-(k-1)}$, debido a que

P1 entrenamiento	P1 test	P2 entrenamiento	P2 test
19.760	4.941	17.290	7.411

Tabla II
TAMAÑO DEL CONJUNTO DE ENTRENAMIENTO Y TEST PARA P1 Y P2.

Hiperparámetro	Valor
Tamaño ventana de contexto (k)	70
Dimensión *word embedding* (d)	100
Unidades capa LSTM (h_{LSTM})	74
Unidades capa Dense (h_{dense})	200

Tabla III
HIPERPARÁMETROS DE AMBOS MODELOS NEURONALES.

[2]https://github.com/cristinazuhe/Disambiguation_SocialMedia_LSTM.

$\underrightarrow{CW^{t-k}}\underleftarrow{CW^{t+k}}$ analiza dos ventadas contextuales disjuntas. Por ejemplo, cuando la palabra a desambiguar se encuentra al principio o al final de la sentencia, solamente algunas de las palabras que están situadas a la derecha o a la izquierda de dicha palabra, respectivamente, se tienen en cuenta en el análisis de $\underrightarrow{CW^{t-k}}\underleftarrow{CW^{t+k}}$.

La evaluación en términos de Recall, Precisión y F1 del modelo $\underleftarrow{CW^{k}}\underrightarrow{CW^{n-(k-1)}}$ para cada sentido se muestra en la Tabla V. Los mejores resultados se obtienen para la partición P2. Incluir expresiones para los *emojis* mejora el rendimiento del modelo en todas las clases excepto para el sentido 5. El valor más alto para la medida F1 es alcanzado en los sentidos 1 y 5 mientras que los resultados más bajos se obtienen en los sentidos 3 y 4. Se puede concluir que sustituir los *emojis* por sus correspondientes expresiones es beneficioso para el rendimiento del sistema. En base a la medida F1, los *emojis* son buenas características para los sentidos 3 y 4, por ejemplo la medida F1 para el sentido 3 aumenta en un 3,14 % al usar *emojis* como características.

Emojis Los *emojis* que afectan positivamente a cada sentido después de reemplazarlos con sus correspondientes descripciones mediante palabras se muestran en la Tabla VI. Si se eliminan los *emojis* mostrados en la primera fila, el modelo clasifica las correspondientes instancias con el sentido 1. Sin embargo, estas instancias se clasifican con el sentido correcto, 4, si las instancias sí incluyen las descripciones de los *emojis*. Intuitivamente, es evidente que todos los *emojis* mostrados tienen una fuerte relación con las clases a las que se asocian después de entrenar los modelos, por lo que son características informativas. Por ejemplo, el melón y la fresa están relacionados con la fruta, los bíceps y las notas musicales están relacionadas con el deporte/música. Sin embargo, algunos *emojis* no están asociados trivialmente a un sentido particular. Por ejemplo, los *emojis* de la última fila, el coche de policía y la luz giratoria del coche de policía, se asocian durante el entrenamiento al sentido 2, que hace referencia a las localizaciones que no están situados en España. Por tanto, se concluye con nuestro estudio que en estas áreas hay demasiados disturbios policiales.

Word embedding Se analiza la importancia de usar *word embeddings* pre-entrenados. Se comparan los resultados alcanzados usando *GloVe* con los resultados obtenidos con *word embeddings* inicializados mediante una distribución gaussiana. Se realiza el análisis sobre la partición P2 incluyendo la descripción de los *emojis*. Los resultados se muestran en la Figura 3. La clasificación a partir de los *embeddings* pre-entrenados usando *GloVe* alcanzan resultados ligeramente más altos que la clasificación a partir de los *embeddings* gaussianos como se esperaba, ya que los *embeddings* pre-entrenados incorporan conocimiento previo semántico a los modelos neuronales.

Partición	*Emojis*	Sentido	%Prec.	%Recall	%F1
P1	No	S1	96,44	98,91	97,66
		S2	96,61	96,74	96,67
		S3	100,0	92,31	96,00
		S4	96,67	93,74	95,18
		S5	99,35	95,60	97,44
	Sí	S1	96,40	98,70	97,53
		S2	96,49	97,31	96,90
		S3	97,37	94,87	96,10
		S4	93,47	96,66	95,03
		S5	99,67	95,60	97,59
P2	No	S1	96,57	98,62	97,58
		S2	96,63	96,72	96,68
		S3	97,46	89,84	93,50
		S4	96,52	93,81	95,15
		S5	98,90	98,25	98,57
	Sí	S1	97,22	98,79	98,00
		S2	96,28	97,21	96,74
		S3	97,60	95,31	96,44
		S4	97,25	94,57	95,89
		S5	98,46	98,03	98,24

Tabla V
Resultados del mejor modelo analizados para cada sentido para P1 y P2 con y sin *emojis*.

Sin *emojis*	*Emojis*	Con *emojis*
1	🏉🍺🎸🎶🎵🎼⚽🏀💪🎹	4
4	🏢🐄🐄🐈🎄🌴👩🌞	1
1	🍎🍏🍊🍐🍆🍈🍓😋	3
1	🚓🚨	2

Tabla VI
El modelo clasifica correctamente el sentido de la palabra usando los *emojis* de la columna *Emojis*.

Modelo	Partición	*Emojis*	Prec.	Recall	F1
XGBoost	P1	No	18,41	17,26	17,82
		Sí	18,63	20,22	19,39
	P2	No	28,20	24,20	26,05
		Sí	55,19	49,14	51,99
$\underrightarrow{CW^{t-k}}\underleftarrow{CW^{t+k}}$	P1	No	95,27	93,63	94,44
		Sí	95,78	93,89	94,72
	P2	No	96,23	93,76	94,98
		Sí	95,67	95,35	95,51
$\underleftarrow{CW^{k}}\underrightarrow{CW^{n-(k-1)}}$	P1	No	**97,81**	**95,46**	**96,63**
		Sí	**97,32**	**95,99**	**96,65**
	P2	No	**97,22**	**95,45**	**96,32**
		Sí	**97,36**	**96,78**	**97,07**

Tabla IV
Resultados de los modelos para P1 y P2 con y sin *emojis*.

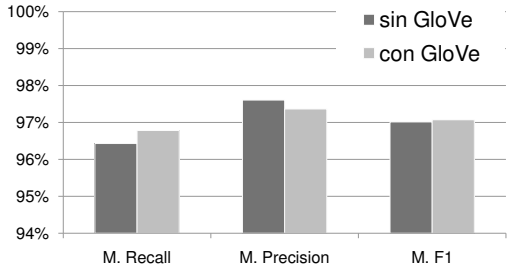

Figura 3. Resultados del mejor modelo usando *word embeddings* aleatorios (sin *GloVe*) o pre-entrenados (usando *GloVe*).

VII. Conclusiones y trabajo futuro

En este trabajo se propone una nueva arquitectura neuronal basada en el uso de las redes LSTM para la tarea de desambiguación de entidades, específicamente el modelo *Two k-Contextual Windows*. Además, se proporciona un nuevo corpus etiquetado para la desambiguación de la entidad *Granada* en textos extraídos de medios sociales, lo cuál es una aportación relevante para la comunidad investigadora.

En nuestra experimentación se pone en manifiesto que nuestro modelo propuesto logra resultados muy buenos en textos escritos en inglés extraídos de medios sociales. En contraste con [5], la codificación de la palabra a desambiguar contribuye a mejorar el rendimiento del sistema. Finalmente, tanto el uso de *embeddings* como la inclusión de expresiones para representar a los *emojis* permite alcanzar mejores resultados.

Como trabajo futuro, se desarrollarán nuevas técnicas para abordar el problema del desbalanceo de clases. Se trabajará en la creación de nuevos corpus de datos en otros idiomas con el objetivo de realizar experimentos en un entorno multilingüe.

Agradecimientos

Agradecemos a la empresa Mabrian por compartir los datos que han sido utilizados en la evaluación de nuestra propuesta. Este trabajo contó con el apoyo parcial del Ministerio de Economía y Competitividad de España en el marco del proyecto TIN2017-89517-P, y con una subvención del FEDER. Siham Tabik contó con el apoyo del Programa Ramón y Cajal (RYC-2015-18136) del Gobierno de España, y Eugenio Martínez Cámara fue apoyado por el Programa Juan de la Cierva Formación(FJCI-2016-28353) del Gobierno de España. En esta investigación usamos Titan X Pascal donada por NVIDIA Corporation.

Referencias

[1] M. T. Thai, W. Wu, and H. Xiong, *Big Data in Complex and Social Networks*. Chapman and Hall/CRC, 11 2016.

[2] A. Farzindar and D. Inkpen, *Natural Language Processing for Social Media*, 2nd ed., ser. Synthesis Lectures on Human Language Technologies. Morgan & Claypool Publishers, 12 2017, vol. 10.

[3] E. Agirre and P. Edmonds, *Word Sense Disambiguation: Algorithms and Applications (Text, Speech and Language Technology)*. Secaucus, NJ, USA: Springer-Verlag New York, Inc., 2006.

[4] R. Navigli, "Word sense disambiguation: A survey," *ACM Comput. Surv.*, vol. 41, no. 2, pp. 10:1–10:69, Feb. 2009.

[5] M. Kågebäck and H. Salomonsson, "Word sense disambiguation using a bidirectional LSTM," in *Proceedings of the 5th Workshop on Cognitive Aspects of the Lexicon (CogALex - V)*. Osaka, Japan: The COLING 2016 Organizing Committee, December 2016, pp. 51–56.

[6] E. Martínez-Cámara, M. T. Martín-Valdivia, L. A. Ureña López, and A. Montejo-Ráez, "Sentiment analysis in Twitter," *Natural Language Engineering*, vol. 20, no. 1, p. 1–28, 2014.

[7] I. Iacobacci, M. T. Pilehvar, and R. Navigli, "Embeddings for word sense disambiguation: An evaluation study," in *Proceedings of the 54th Annual Meeting of the ACL (Vol 1: Long Papers)*, 2016, pp. 897–907.

[8] M. Insall, T. Rowland, and E. W. Weistein, "Embedding."from mathworld–a wolfram web resource (access March 12, 2018)," 2015.

[9] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient estimation of word representations in vector space," *arXiv preprint:1301.3781*, 2013.

[10] J. Pennington, R. Socher, and C. Manning, "Glove: Global vectors for word representation," in *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing*. Doha, Qatar: ACL, October 2014, pp. 1532–1543.

[11] Y. Goldberg, *Neural Network Methods for Natural Language Processing*. Morgan & Claypool Publishers, 2017.

[12] R. Pascanu, T. Mikolov, and Y. Bengio, "On the difficulty of training recurrent neural networks," in *Proceedings of the 30th International Conference on International Conference on Machine Learning - Volume 28*, ser. ICML'13. JMLR.org, 2013, pp. III–1310–III–1318.

[13] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. MIT Press, 2016, ch. 10.

[14] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Computation*, vol. 9, no. 8, pp. 1735–1780, Nov. 1997.

[15] A. Graves, "Supervised sequence labelling with recurrent neural networks," Ph.D. dissertation, Technishe Universit´at M´unchen, 2008.

[16] A. Raganato, J. Camacho-Collados, and R. Navigli, "Word sense disambiguation: A unified evaluation framework and empirical comparison," in *Proceedings of the 15th Conference of the European Chapter of the ACL: Volume 1, Long Papers*. ACL, 2017, pp. 99–110.

[17] Z. S. Harris, "Distributional structure," *WORD*, vol. 10, no. 2-3, pp. 146–162, 1954.

[18] J. R. Firth, "A synopsis of linguistic theory 1930-55." vol. 1952-59, pp. 1–32, 1957.

[19] G. A. Miller and W. G. Charles, "Contextual correlates of semantic similarity," *LCP*, vol. 6, no. 1, pp. 1–28, 1991.

[20] Z. Zhong and H. T. Ng, "It makes sense: A wide-coverage word sense disambiguation system for free text," in *Proceedings of the ACL 2010 System Demonstrations*. Uppsala, Sweden: ACL, July 2010, pp. 78–83.

[21] H. Shen, R. Bunescu, and R. Mihalcea, "Coarse to fine grained sense disambiguation in wikipedia," in *Second Joint Conference on Lexical and Computational Semantics (*SEM), Volume 1: Proceedings of the Main Conference and the Shared Task: Semantic Textual Similarity*. Atlanta, Georgia, USA: ACL, June 2013, pp. 22–31.

[22] R. Collobert, J. Weston, L. Bottou, M. Karlen, K. Kavukcuoglu, and P. Kuksa, "Natural language processing (almost) from scratch," *Journal of Machine Learning Research*, vol. 12, pp. 2493–2537, Nov. 2011.

[23] C. D. Santos and B. Zadrozny, "Learning character-level representations for part-of-speech tagging," in *Proceedings of the 31st International Conference on Machine Learning*, E. P. Xing and T. Jebara, Eds., vol. 32, no. 2, 2014, pp. 1818–1826.

[24] E. Kiperwasser and Y. Goldberg, "Simple and accurate dependency parsing using bidirectional lstm feature representations," *Transactions of the ACL*, vol. 4, pp. 313–327, 2016.

[25] D. F. Wong, Y. Lu, and L. S. Chao, "Bilingual recursive neural network based data selection for statistical machine translation," *Knowledge-Based Systems*, vol. 108, pp. 15 – 24, 2016, new Avenues in Knowledge Bases for Natural Language Processing.

[26] K. Lin, D. Li, X. He, M.-t. Sun, and Z. Zhang, "Adversarial ranking for language generation," in *Advances in Neural Information Processing Systems 30*, I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, Eds. Curran Associates, Inc., 2017, pp. 3158–3168.

[27] O. Melamud, J. Goldberger, and I. Dagan, "context2vec: Learning generic context embedding with bidirectional lstm," in *Proceedings of The 20th SIGNLL Conference on Computational Natural Language Learning*. ACL, 2016, pp. 51–61.

[28] K. Taghipour and H. T. Ng, "Semi-supervised word sense disambiguation using word embeddings in general and specific domains," in *Proceedings of the 2015 Conference of the North American Chapter of the ACL: Human Language Technologies*. Denver, Colorado: ACL, May–June 2015, pp. 314–323.

[29] A. Kilgarriff, "English lexical sample task description," in *The Proceedings of the Second International Workshop on Evaluating Word Sense Disambiguation Systems*. ACL, 2001, pp. 17–20.

[30] R. Mihalcea, T. Chklovski, and A. Kilgarriff, "The senseval-3 english lexical sample task," in *Proceedings of senseval-3, the third international workshop on the evaluation of systems for the semantic analysis of text*, 2004.

[31] J. Read, "Using emoticons to reduce dependency in machine learning techniques for sentiment classification," in *Proceedings of the ACL Student Research Workshop*. ACL, 2005, pp. 43–48.

[32] N. Qian, "On the momentum term in gradient descent learning algorithms," *Neural Networks*, vol. 12, no. 1, pp. 145 – 151, 1999.

[33] T. Chen and C. Guestrin, "Xgboost: A scalable tree boosting system," in *Proceedings of the 22Nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, ser. KDD '16. New York, NY, USA: ACM, 2016, pp. 785–794.