Procesamiento del Lenguaje Natural, Revista nº 67, septiembre de 2021, pp. 45-57 recibido 29-04-2021 revisado 04-06-2021 aceptado 07-06-2021

Procesamiento de Expresiones Multipalabra en gallego mediante Aprendizaje Profundo

Multiword expressions processing in Galician using Deep Learning

Víctor Darriba[1], Yerai Doval[1], Elmurod Kuriyozov[2]
[1]Universidad de Vigo, Depto. de Informática
[2]Universidad de A Coruña, CITIC
{darriba,yerai.doval}@uvigo.es, e.kuriyozov@udc.es

Resumen: El tratamiento de Expresiones Multipalabra es todavía una tarea pendiente en el Procesamiento del Lenguaje Natural. En este trabajo pretendemos determinar experimentalmente la utilidad de los modelos de Aprendizaje Automático para el procesamiento de Expresiones Multipalabra en gallego. Para ello usamos CORGA, un corpus con 40 millones de palabras, con el cual entrenamos modelos *transformer* de Aprendizaje Profundo, y comparamos su rendimiento con el de modelos más tradicionales de campo aleatorio condicional.
Palabras clave: Expresiones multipalabra, aprendizaje automático, transformers, gallego.

Abstract: Treatment of Multiword Expressions is still a pending task in Natural Language Processing. In this work, we want to experimentally determine the usefulness of Machine Learning models for Multiword Expression processing in Galician. With that aim, we use *CORGA*, a 40 million word corpus, with which we train Deep Learning-based *transformers*, comparing their performances with those of more traditional conditional random fields.
Keywords: Multiword expressions, machine learning, transformers, Galician.

1 Introducción

Aunque no existe una única definición universalmente aceptada, se suelen considerar como Expresiones Multipalabra (EMs) las "combinaciones de palabras habituales y recurrentes del lenguaje común" (Firth, 1957). Hay muchos tipos diferentes: frases hechas ("cantar las cuarenta"), colocaciones ("derecho de veto"), nombres propios ("Miguel Pérez"), etc.

Las EMs son comunes en todos los idiomas y dominios (Jackendoff, 1997). Por ejemplo, Ramisch (2015) informa que el 51,4 % de los nombres y el 25,5 % de los verbos de la versión inglesa de WordNet son multipalabra. Desde el punto de vista léxico, muchas EMs tienden a comportarse como palabras individuales, y su semántica no tiene por qué resultar de la simple composición del significado de sus palabras constituyentes. Por lo tanto, es aconsejable incorporar el tratamiento de EMs en tareas basadas en Procesamiento del Lenguaje Natural (PLN) (Ramisch, 2015).

En la literatura se distinguen dos tareas en el procesamiento de EMs (Constant et al., 2017): detección e identificación. La detección se centra en encontrar EMs no vistas anteriormente en corpora textuales, con el fin de almacenarlas en algún tipo de repositorio (como un lexicón) para su uso futuro: por ejemplo, detectar como EM el nombre propio "Oseja de Sajambre" cuando aparezca en un corpus. Por su parte, la identificación consiste en anotar automáticamente las EMs presentes en un texto. Ambas tareas están relacionadas. Una lista de EMs obtenidas mediante detección pueden ser utilizadas como un recurso externo por herramientas de identificación. Por otra parte, una herramienta de identificación con capacidad de generalización a partir de ejemplos conocidos puede usarse para detectar nuevas EMs.

Este trabajo se centra en el uso de herramientas de Aprendizaje Automático (AA) supervisado para el procesamiento de EMs en el idioma gallego. Para ello, tratamos la detección e identificación de EMs como tareas de etiquetación de secuencias. Cada palabra individual en el corpus de entrenamiento recibe una etiqueta que indica si forma parte de una EM o no, de modo que el modelo entrenado

ISSN 1135-5948. DOI 10.26342/2021-67-4

con dicho corpus aplica el mismo esquema de anotación a cualquier nuevo texto que reciba como entrada, intentando asignar la secuencia de etiquetas más idónea a cada frase de dicho texto.

En este sentido, nuestro propósito es testear la viabilidad de dos modelos modernos de Aprendizaje Profundo (AP): BERT multilingue (mBERT) (Devlin et al., 2019) y XLM-RoBERTa (XLM-R) (Conneau et al., 2020). Se trata de dos arquitecturas *transformer*, basadas en la generación de modelos de lenguaje para múltiples idiomas, que después pueden ser reentrenados para una tarea concreta. Para estimar la posible mejora introducida por estos modelos respecto a aproximaciones previas, utilizaremos como referencia mwetoolkit (Ramisch, 2015), una herramienta de que permite entrenar modelos de campo aleatorio condicional (*Conditional Random Fields* o CRF) (Lafferty, McCallum, y Pereira, 2001) para la anotación automática de EMs en textos.

Para entrenar y probar dichos modelos usamos el *Corpus de Referencia do Galego Actual* (CORGA) (Centro Ramón Piñeiro para a Investigación en Humanidades, 2019a), que cuenta con una versión con anotación morfosintáctica y lematización realizadas automáticamente, en la que se analizan como una sola unidad, en forma de "palabras con espacios", dos tipos de EMs: Entidades Nombradas (*Named Entities* o NEs) y locuciones.

La estructura del resto de este artículo es la siguiente. La Sección 2 presenta trabajos anteriores sobre procesamiento de EMs usando AA, y en la Sección 3 se describen la metodología, modelos y datos empleados en nuestros experimentos. En la Sección 4 se presentan los resultados de los mismos, y en la Sección 5 se detallan nuestras conclusiones.

2 Trabajos relacionados

Centraremos nuestro análisis principalmente en los trabajos basados en técnicas de AA supervisado, más populares en la identificación de EMs, aunque también veremos como se han usado en la tarea de detección.

2.1 Identificación

En esta tarea es habitual utilizar algún tipo de modelo de etiquetación de secuencias. Un ejemplo es Blunsom y Baldwin (2006), que asignan tipos léxicos a las palabras de la secuencia de entrada usando un modelo CRF, mientras que Vincze, Nagy T., y Berend (2011) entrenan un CRF con un corpus previamente anotado con EMs, junto con lexicones externos, y Diab y Bhutada (2009) entrenan Máquinas de Vectores de Soporte (SVMs) sobre un corpus anotado con construcciones nombre-verbo, con el objetivo de distinguir aquellas que son idiomáticas.

Es frecuente integrar información de diccionarios externos de EMs para calcular características de las entradas usadas en el proceso de entrenamiento. Así, Constant y Sigogne (2011) entrenan un modelo CRF para la realización conjunta del análisis léxico y etiquetación morfosintáctica partiendo de un banco de árboles. Por su parte, Constant, Sigogne, y Watrin (2012) comparan el rendimiento de un etiquetador de secuencias (un CRF) frente a un analizador sintáctico con reordenación de árboles. Las características obtenidas a partir de lexicones usadas en este trabajo son refinadas por Schneider et al. (2014) para el entrenamiento de un perceptrón estructurado, usando un juego de etiquetas que permite anotar EMs no contiguas y EMs anidadas. Finalmente, Riedl y Biemann (2016) intentan determinar el impacto en el rendimiento de características obtenidas a partir de anotación manual y mediante anotación automática, partiendo de las usadas en los trabajos anteriores.

Otra aproximación es usar información sintáctica en el proceso de identificación (Constant, Sigogne, y Watrin, 2012). Dicha información puede integrarse como características de un modelo de etiquetación de secuencias (Maldonado et al., 2017), pero es más habitual identificar EMs durante el análisis sintáctico, entrenando un modelo a partir de un banco de árboles anotado con EMs. Se pueden usar diferentes formalismos gramaticales: Green et al. (2011) trabajan con gramáticas de sustitución de árboles, anotando cada EM como un subárbol de estructura plana, aproximación también usada por Green, de Marneffe, y Manning (2013) junto con gramáticas independientes del contexto probabilísticas. Otra posibilidad es el análisis sintáctico de dependencias, empleando arcos específicos para denotar los componentes de una EM (Vincze, Zsibrita, y Nagy T., 2013; Simkó, Kovács, y Vincze, 2017). También se pueden escoger diferentes representaciones según el tipo de EM: Candito y Constant (2014) y Constant y Nivre

(2016) usan relaciones de dependencia para las expresiones sintácticamente regulares y una estructura plana para las irregulares.

La popularidad creciente de las Redes de Neuronas ha llevado a su empleo en la identificación de EMs (Legrand y Collobert, 2016), haciendo uso de representaciones vectoriales continuas (*word embeddings*). Múltiples arquitecturas de red han sido propuestas para esta tarea: Klyueva, Doucet, y Straka (2017) y Zampieri et al. (2018) usan *Gated Recurrent Units* (GRUs), mientras que Taslimipoor y Rohanian (2018) combinan Redes Convolucionales y *Long Short-Term Memory* (LSTMs). Recientemente, se ha empezado a hacer uso de arquitecturas *transformer*: Taslimipoor, Bahaadini, y Kochmar (2020) usan BERT como modelo de lenguaje para el aprendizaje conjunto de EMs y análisis sintáctico de dependencias, mientras que Kurfalı (2020) emplea BERT y mBERT para la identificación de EMs verbales.

2.2 Detección

Lo más habitual en esta tarea es utilizar métodos no supervisados, distinguiendo EMs en base a sus puntuaciones en métricas de asociación, sustitubilidad de palabras componentes o similitud semántica. Sin embargo, también existen trabajos basados en técnicas supervisadas de AA, usando a menudo las métricas mencionadas como características en el aprendizaje junto con información lingüística. Por ejemplo, Pecina (2009) utiliza modelos basados en Regresión Logística, Análisis Discriminante Lineal y SVMs para el descubrimiento de colocaciones. Los SVMs son también usados por Farahmand y Martins (2014) en conjunción con medidas estadísticas y características contextuales de las EMs, como prefijos y sufijos. Por su parte, Lapata y Lascarides (2003) comparan el rendimiento de árboles de decisión y clasificadores bayesianos para extraer nombres compuestos, mientras que Dubremetz y Nivre (2014) prueban varios algoritmos de aprendizaje para extraer grupos nombre-nombre y nombre-adjetivo, obteniendo los mejores resultados con redes bayesianas. Los modelos así entrenados pueden utilizarse para determinar cuales de las características usadas en el entrenamiento son más útiles (Ramisch et al., 2008), o para intentar reducir su dependencia sobre recursos de entrenamiento, como Rondon, Caseli, y Ramisch (2015), que implementan un sistema iterativo basado en SVMs que, partiendo de un lexicón y modelo iniciales, mina texto de la web, anota sus EMs y usa el nuevo texto anotado para entrenar una nueva versión del modelo.

2.3 Nuestra aportación

Este trabajo usa un enfoque similar al de Kurfalı (2020), estudiando la viabilidad de las arquitecturas *transformer* para el tratamiento de EMs, si bien hay grandes diferencias entre ambas aproximaciones. En primer lugar, nos centraremos en un único idioma, el gallego, minoritario y pobre en recursos para PLN, y tratamos de determinar la mejor configuración de entrenamiento en base a la información léxica contenida en CORGA. Además, probamos dos arquitecturas distintas, mBERT y XLM-R, y usamos un modelo CRF para determinar si ofrecen un mejor rendimiento que aproximaciones más tradicionales. Finalmente, en lugar de centrarnos solamente en la tarea de identificación, también tratamos de medir el rendimiento de estas arquitecturas en el reconocimiento de nuevas EMs para la tarea de detección.

3 Enfoque

En esta sección vamos a examinar los recursos usados en nuestros experimentos, así como el diseño de los mismos.

3.1 CORGA

El *Corpus de Referencia do Galego Actual* (Centro Ramón Piñeiro para a Investigación en Humanidades, 2019a) es una colección documental, abierta y representativa que recoge textos y transcripciones de habla en gallego, desde 1975 hasta la actualidad, con el objetivo de proporcionar datos para el estudio lingüístico de dicho idioma. Contiene 40.178.271 millones de palabras y 48.184.012 millones de elementos gramaticales (palabras o entidades léxicas amalgamadas en palabras, signos de puntuación, etc) procedentes de diversas fuentes: periódicos, revistas, libros, guiones televisivos, blogs y transcripciones de programas radiofónicos (Domínguez Noya, López Martínez, y Barcala Rodríguez, 2019).

Para posibilitar el estudio del idioma desde múltiples perspectivas, CORGA está etiquetado morfosintácticamente y lematizado de manera automática. Para ello, se ha empleado el *Etiquetador Lematizador do Galego Actual* (XIADA) (Centro Ramón Piñeiro pa-

ra a Investigación en Humanidades, 2019b), una herramienta basada en Modelos de Markov entrenada sobre un subconjunto de CORGA con 744.530 elementos gramaticales, con anotación automática corregida por expertos. Para recoger la rica morfología de la lengua gallega, se usa un juego de 453 etiquetas.[1] Aunque no se ha medido la exactitud de la versión actual del etiquetador sobre CORGA, una versión previa alcanzó un 96 % sobre textos periodísticos (Domínguez Noya, Barcala Rodríguez, y Molinero Álvarez, 2009).

El proceso de etiquetación/lematización de XIADA incluye la identificación de dos tipos de EMs: a) NEs, como "Universidade de Santiago", "Ministerio de Agricultura", "Fondo Monetario Internacional", etc; y b) Locuciones adverbiales, preposicionales y conjuntivas, como "por tanto", "por se acaso" ("por si acaso"), "sen dúbida"("sin duda"),"si ben" ("si bien"), etc. Para el proceso de identificación de estas últimas, XIADA hace uso de un lexicón extraído de varias fuentes, con un total de 628 locuciones.

Dicha tarea de identificación es una característica opcional del etiquetador, por lo que hemos dispuesto de dos versiones de CORGA con etiquetación y lematización automáticas: una con cada EM etiquetada y lematizada como una unidad, y otra en donde cada entidad constituyente de una EM es etiquetada y lematizada por separado. Ello permite recuperar la información morfosintáctica de los constituyentes de las EMs, combinando la información de ambas versiones del corpus.

Otra característica importante es la separación de amalgamas. El idioma gallego presenta un gran número de palabras gramaticales formadas por amalgamas de entidades léxicas, como verbos con clíticos o contracciones de preposiciones, artículos o determinantes: "encóllese" = "encolle"+"se" ("se encoge"), "coa"="con"+"a" ("con la"), "naqueloutro"= "en"+"aquel"+"outro"("en aquel otro"), etc. XIADA separa y etiqueta/lematiza esos constituyentes léxicos, lo que hace posible el tratamiento de EMs directamente sobre las formas amalgamadas o sobre las entidades léxicas que las integran.

3.2 Modelos de aprendizaje

En nuestros experimentos empleamos modelos de AP bien conocidos en el estado del arte en PLN y basados en la arquitectura *transformer*: mBERT (Devlin et al., 2019) y XLM-R (Conneau et al., 2020). El componente principal en estos modelos es el mecanismo de atención, mediante el cual se pueden realizar cálculos para un elemento de la secuencia de entrada ponderando las contribuciones del resto de dicha secuencia. Ello viene a sustituir al mecanismo de recurrencia de las redes basadas en celdas LSTMs (Hochreiter y Schmidhuber, 1997) o GRUs (Cho et al., 2014) que sólo consideraban para sus cálculos el estado anterior al procesar en secuencia el elemento actual. Sin embargo, esta indudable ventaja se obtiene a costa de un aumento significativo de la complejidad computacional.

Otro elemento característico de estos modelos es su entrenamiento en dos etapas: una primera de preentrenamiento en una tarea no supervisada, normalmente una variación del modelado del lenguaje, y una segunda de entrenamiento (semi-)supervisado en la tarea específica que se desea resolver, como el procesamiento de EMs en nuestro caso.

Tanto mBERT como XLM-R están preentrenados sobre grandes cantidades de texto en varios idiomas: volcados de la Wikipedia en cada idioma en el primer caso y una versión filtrada de Common Crawl particionada por idioma en el segundo. El gallego es uno de los idiomas incluidos, si bien la proporción del mismo en los datos de entrenamiento de estas arquitecturas es muy pequeña en comparación con idiomas más mayoritarios como el inglés.[2] Sin embargo, dado el alto coste de reentrenar y optimizar estos modelos desde el principio, consideramos *a priori* estos recursos multilingües como adecuados para nuestros experimentos. La principal diferencia entre ambas arquitecturas radica en la escala del corpus usado durante sus respectivos preentrenamientos, mucho mayor en el caso de XLM-R, que además posee un mayor número de parámetros. Además, en el caso de mBERT utilizamos dos variantes: una preentrenada con la totalidad del texto pasado a minúsculas, y otra con el texto original.

A nivel de implementación, usamos la librería Python Transformers de HuggingFace (Wolf et al., 2020) junto con los modelos preentrenados y distribuidos desde sus canales. Dichos modelos poseen un tamaño considerable: 179 millones de parámetros distri-

[1] Ver http://corpus.cirp.gal/xiada/etiquetario/taboa/ para una descripción de las mismas.

[2] Tal y como se puede ver en la Figura 1 de Conneau et al. (2020).

buidos en 12 capas para mBERT (168 millones de parámetros en la variante preentrenada en texto en minúsculas) y alrededor de 270 millones de parámetros distribuidos de forma similar en el caso de XLM-R. Para el entrenamiento, mantenemos los valores por defecto en la mayor parte de los hiperparámetros y especificamos un tamaño de *batch* y una longitud máxima de secuencia de 64 elementos, sin realizar ningún proceso de optimización de estos valores. El proceso se desarrolla durante una sola iteración sobre el conjunto completo de datos de entrenamiento.

Para comparar los resultados de las técnicas de AP con aproximaciones más tradicionales, usamos mwetoolkit (Ramisch, 2015). Esta herramienta permite entrenar un modelo CRF (Lafferty, McCallum, y Pereira, 2001), implementado con CRFSuite (Okazaki, 2007). El usuario puede determinar las características extraídas de las entradas del modelo, que en nuestro caso son las sugeridas por los autores de mwetoolkit (unigramas, bigramas y trigramas de etiquetas morfosintácticas y lemas, alrededor de la palabra actual), y que están inspiradas por los trabajos de Constant y Sigogne (2011), Schneider et al. (2014) y Riedl y Biemann (2016). Con respecto a los parámetros de la arquitectura, el único configurable desde mwetoolkit es el coeficiente λ de regularización L_2, teniendo el resto los valores por defecto fijados por CRFSuite. Tras algunas pruebas preliminares, hemos elegido λ =0,1 para todos los tests.

Finalmente, también hemos implementado un modelo simple de referencia (*Baseline*), consistente en extraer del corpus de entrenamiento las secuencias de etiquetas morfosintácticas correspondientes a EMs, eliminar aquellas con un número de ocurrencias inferior a 500 y anotar como EMs en el conjunto de prueba las entidades léxicas con esas secuencias de etiquetas. En caso de que haya dos o más posibles EMs que empiezan o terminan en la misma entidad léxica, seleccionamos la más larga; y si hay dos posibles EMs que se solapan, nos quedamos con la correspondiente a la secuencia de etiquetas más frecuente en el conjunto de entrenamiento.

3.3 Diseño experimental

Para poder entrenar los modelos de aprendizaje es necesario disponer de un esquema de etiquetación para EMs. Hemos elegido IOB2 (Ramshaw y Marcus, 1995),[3] que consta de tres etiquetas: B para marcar el elemento gramatical en donde comienza una EM, I para el resto de constituyentes de una expresión, y O para los elementos fuera de una EM. En el caso particular de mwetoolkit se usa el formato DiMSUM (Schneider et al., 2016), que permite incluir información adicional, como el lema y etiqueta morfosintáctica. En el caso de usar palabras amalgamadas, sus etiquetas morfosintácticas se generan automáticamente como la concatenación de las etiquetas de sus entidades léxicas constituyentes

Para encontrar el formato de CORGA que ofrezca el mejor rendimiento para el tratamiento de EMs, consideramos 8 configuraciones de prueba para cada arquitectura, combinando las siguientes opciones: a) usando palabras ortográficas amalgamadas, o sus constituyentes separados; b) usando todas las frases del corpus en el aprendizaje, o sólo las que contienen EMs; y c) usando las palabras o lemas tal y como aparecen en el corpus, o convirtiéndolas a minúsculas. Inicialmente, esta última modalidad se incluyó para probar la versión de mBERT preentrenada con texto en minúsculas, pero la hemos extendido a las otras arquitecturas.

Al probar el rendimiento en diferentes configuraciones de prueba, usamos la misma partición del conjunto de datos inicial en tres subconjuntos de entrenamiento, validación y prueba, con una proporción aproximada del 80 %/10 %/10 % de las frases. El conjunto de entrenamiento se utiliza en el aprendizaje del modelo. Una vez entrenado, su rendimiento en la predicción de EMs se mide sobre el conjunto de prueba con las métricas *Accuracy*, *Precision*, *Recall* [4] y valor F1. Los conjuntos de validación se han empleado para seleccionar el coeficiente λ en mwetoolkit y el umbral de ocurrencias de *Baseline*, adoptando los valores asociados a los mejores resultados de F1 sobre dichos conjuntos. También queremos emplearlos en el futuro para optimizar los hiperparámetros de mBERT y XLM-R.

Identificación. Primero hemos filtrado de CORGA fragmentos en idiomas extranjeros y frases cuya diferente segmentación en las versiones con o sin EMs hace imposible incorporar los lemas y etiquetas a los constituyen-

[3] De ***I**nside-**O**utside-**B**eginning*.

[4] En español, Exactitud, Precisión y Exhaustividad, respectivamente, pero usaremos los nombres en inglés por su mayor difusión.

	Entrenamiento	Validación	Prueba
Todas las EMs	36.183.656 (11.159.908)	3.647.604	3.474.748
Sólo NEs	31.619.486 (6.595.738)	3.379.553	3.333.069
Sólo Locuciones	28.431.686 (3.407.938)	2.970.140	2.976.181

Tabla 1: Número de elementos gramaticales en los conjuntos de entrenamiento usados para las pruebas de detección de EMs. Indicamos entre paréntesis el tamaño de los conjuntos de entrenamiento incluyendo sólo las frases sin EMs.

tes de las EMs automáticamente. A continuación, particionamos las secciones de CORGA (periódicos, revistas, libros, guiones televisivos, blogs y transcripciones de programas radiofónicos) de modo que el primer 80 % de las frases de cada sección se asigna al conjunto de entrenamiento, el siguiente 10 % al de validación, y el 10 % final al de prueba. De este modo, el número de elementos gramaticales es de 38.871.031 para el conjunto de entrenamiento (14.173.449 si sólo incluimos las frases con EMs), 4.955.264 para el de validación y 4.573.942 para el de prueba, incluyendo 738.118 EMs.[5]. Con este particionamiento, nos aseguramos de que sólo el 75,22 % de las EMs presentes en los conjuntos de prueba sean conocidas durante el entrenamiento.

Detección. El particionamiento anterior tiene el problema de que el reducido número de locuciones distintas identificadas en CORGA hace que la práctica totalidad de las mismas sean conocidas durante el aprendizaje, lo que no permite estimar la capacidad de generalización de las arquitecturas consideradas para identificar nuevas locuciones. Dicha capacidad de generalización puede ser útil, además, en la tarea de detección, aplicando modelos ya entrenados a nuevo texto y extrayendo las EMs desconocidas encontradas.

Por lo tanto, pretendemos evaluar el rendimiento de las arquitecturas consideradas en la detección de EMs, poniéndonos en el caso más desfavorable: una partición del corpus en la que no haya solapamiento entre las EMs presentes en los conjuntos de entrenamiento, validación y test. Para ello, seleccionamos un subconjunto de las EMs presentes en CORGA, lo ordenamos aleatoriamente y añadimos las frases conteniendo el primer 80 % de las expresiones en el conjunto de entrenamiento, las correspondientes a otro 10 % en el conjunto de validación y las del 10 % final en el conjunto de prueba. Completamos los conjuntos añadiendo frases sin EMs con la misma proporción de 80 %/10 %/10 %, aunque sin llegar a usar CORGA en su totalidad.

Además, realizamos el mismo proceso de evaluación de arquitecturas sobre las NEs y locuciones, respectivamente. En el primer caso, generamos los conjuntos de entrenamiento, validación y prueba como se detalla en el párrafo anterior, pero usando sólo las frases con NEs, y cambiando a O las etiquetas IOB2 de las locuciones que aparezcan en dichas frases. También añadimos frases sin EMs para las configuraciones de prueba que las requieran. El mismo proceso se realiza para los tests sobre locuciones, pero prescindiendo en ese caso de las frases con NEs.

La Tabla 1 incluye los tamaños de los conjuntos de entrenamiento, validación y prueba usados para la detección de EMs, NEs multipalabra y locuciones, y el número de estos. Los números entre paréntesis corresponden al tamaño de los conjuntos de entrenamiento cuando incluimos sólo las frases sin EMs. El número total de EMs en estos tests es de 488.035, siendo el 70,53 % de las mismas NEs, y el 29.47 % locuciones.

4 Resultados

En esta sección vamos a presentar los resultados de nuestros experimentos, para todas las arquitecturas de aprendizaje y configuraciones de prueba, tanto en el caso de la identificación como de la detección de EMs.

4.1 Identificación

Los resultados de los experimentos sobre identificación de EMs se presentan en la Tabla 2. Como se especificó en la Subsección 3.3, tenemos 8 configuraciones de prueba, con resultados para las cuatro arquitecturas de entrenamiento, para un total de 32 tests. En cada uno de ellos, medimos *Accuracy* (Acc), *Precision* (P), *Recall* (R) y valor F1.

A la izquierda de la tabla se muestran los resultados para las configuraciones en la que

[5]El total de elementos gramaticales supera a la cifra dada por Domínguez Noya, Barcala Rodríguez, y Molinero Álvarez (2009) porque nosotros contamos separadamente los constituyentes de las EMs.

			Con amalgamas				Sin amalgamas			
			Acc	P	R	F1	Acc	P	R	F1
Todo	Mays	mBERT	99,83	96,32	96,81	96,57	99,84	96,38	96,78	96,58
		XLM-R	99,82	96,32	96,54	96,43	99,83	96,13	96,52	96,32
		mwetoolkit	99,84	97,38	96,48	96,92	**99,85**	**97,49**	96,39	**96,94**
		Baseline	91,52	19,32	53,23	28,35	91,06	17,61	51,12	26,19
	Mins	mBERT	99,28	88,24	87,13	87,68	99,32	87,81	87,19	87,50
		XLM-R	99,25	87,18	87,15	87,16	99,31	87,48	86,85	87,16
		mwetoolkit	99,84	97,31	96,42	96,86	99,85	97,41	96,42	96,91
		Baseline	91,52	19,32	53,23	28,35	91,06	17,61	51,12	26,19
Solo EMs	Mays	mBERT	97,05	46,32	98,26	62,96	99,16	76,39	**98,30**	85,97
		XLM-R	98,50	62,80	98,08	76,57	99,34	80,48	98,04	88,39
		mwetoolkit	99,78	93,15	97,13	95,09	99,79	93,19	97,13	95,12
		Baseline	91,52	19,32	53,23	28,35	91,06	17,61	51,12	26,19
	Mins	mBERT	97,03	49,78	92,15	64,64	97,41	50,84	91,81	65,44
		XLM-R	96,80	47,62	91,77	62,71	97,48	52,77	91,60	66,96
		mwetoolkit	99,77	92,99	97,07	94,99	99,79	93,08	97,16	95,07
		Baseline	91,52	19,32	53,23	28,35	91,06	17,61	51,12	26,19

Tabla 2: Resultados de los tests de identificación de EMs.

se ha usado texto con palabras amalgamadas, mientras que a la derecha aparecen los resultados para el entrenamiento con entidades léxicas sin amalgamas. "Todo" identifica a las configuraciones en las que se ha usado todo el corpus en el proceso de aprendizaje y prueba, mientras que "Solo EMs" corresponde a los casos en los que sólo hemos usado las frases que incluyen EMs en el conjunto de entrenamiento. Finalmente, "Mins" corresponde a los tests en los que hemos usado texto en minúsculas y "Mays" a aquellos en los que hemos mantenido las mayúsculas y minúsculas tal y como aparecen en CORGA.

Respecto a los valores de las métricas, representan porcentajes y, por motivos de espacio, se representan con dos cifras decimales. Se destaca en negrita el mejor resultado de cada métrica para todos los tests, y se muestran subrayados los mejores resultados en cada configuración de prueba. Para determinar los mejores valores se han usado más de dos cifras decimales.

También hemos usado la prueba de los rangos con signo de Wilcoxon para comprobar la significancia estadística de las conclusiones obtenidas analizando la tabla. Para ahorrar espacio, sólo citaremos el valor de p en la minoría de casos en los que $p \geq 0{,}05$.

Lo primero que salta a la vista en la tabla 2 es el liderazgo en rendimiento de mwetoolkit. Obtiene los mejores valores para *Accuracy* (99,85 %), *Precision* (97,49 %), y F1 (96,94 %), mientras que mBERT sólo es capaz de aventajarlo en *Recall* (98,30 %). Además, mwetoolkit obtiene los mejores valores para todas las métricas en todas las configuraciones de prueba, excepto *Recall*, para la que mBERT es el mejor en las cuatro configuraciones que usan texto con mayúsculas y minúsculas ($p = 0{,}055$).

Con respecto a las configuraciones de prueba, los mejores valores de las métricas se agrupan en la configuración "Sin amalgamas"+"Todo"+"Mays", excepto en el caso de *Recall*, cuyo mejor caso es "Sin amalgamas"+"Solo EMs"+"Mays". Comparando cada opción de configuración, entrenar sin amalgamas mejora los valores de *Accuracy*, *Precision* y F1, respectivamente, en un 75 % ($p = 0{,}137$), 62,5 % ($p = 0{,}353$) y 62,5 % ($p = 0{,}372$) de los casos, aunque entrenar con amalgamas consigue un mejor *Recall* en el 75 % de los tests.

Para comparar los resultados de "Todo" vs "Solo EMs" y "Mays" vs "Min", hemos excluído *Baseline*, dado que obtiene el mismo rendimiento en cada par de opciones.[6] El entrenamiento con "Todo" obtiene mejores o iguales valores de *Accuracy*, *Precision* y F1 en todos los tests, mientras que "Solo EMs" siempre genera un mejor *Recall*. Por su parte, usar mayúsculas y minúsculas también mejora los valores de *Accuracy* (100 % de los tests), *Precision* (91,67 % de los tests), *Recall* (83,33 %) y F1 (91,67 %), lo que no resulta sorprendente, dado que buena parte de las EMs que estamos intentando identificar son nombres propios. Hasta la versión de mBERT preentrenada con texto en minúsculas (que usamos en el caso "Mins") parece obtener peores resultados que la versión preen-

[6] Haremos lo mismo en los tests de detección.

			Con amalgamas				Sin amalgamas			
			Acc	P	R	F1	Acc	P	R	F1
Todo	Mays	mBERT	99,58	90,06	83,50	86,65	99,64	**93,68**	81,58	87,22
		XLM-R	99,57	90,18	83,10	86,49	99,70	93,23	84,13	**88,44**
		mwetoolkit	99,58	92,94	81,37	86,77	99,60	92,64	79,27	85,43
		Baseline	91,70	9,41	53,91	16,03	91,10	8,36	54,30	14,49
	Mins	mBERT	99,76	43,73	68,42	53,36	**99,79**	41,74	63,29	50,31
		XLM-R	99,76	43,36	68,25	53,03	99,78	38,64	62,54	47,77
		mwetoolkit	99,58	92,79	81,20	86,61	99,59	92,35	78,91	85,11
		Baseline	91,70	9,41	53,91	16,03	91,10	8,36	54,30	14,49
Solo EMs	Mays	mBERT	97,53	34,67	**94,94**	50,79	99,44	76,59	83,56	79,92
		XLM-R	97,88	39,57	94,25	55,74	99,54	79,53	87,92	83,51
		mwetoolkit	99,52	87,67	83,76	85,67	99,52	85,90	80,91	83,33
		Baseline	91,70	9,41	53,91	16,03	91,10	8,36	54,30	14,49
	Mins	mBERT	96,72	31,87	86,68	46,61	97,14	30,20	83,06	44,30
		XLM-R	96,44	29,67	85,06	43,99	96,84	27,76	79,00	41,08
		mwetoolkit	99,52	87,59	83,81	85,66	99,52	85,62	80,99	83,24
		Baseline	91,70	9,41	53,91	16,03	91,10	8,36	54,30	14,49

Tabla 3: Resultados de los tests de detección de EMs.

trenada con mayúsculas y minúsculas (aunque con $p = 0{,}063$, $p = 0{,}125$, $p = 0{,}063$, $p = 0{,}125$ para *Accuracy*, *Precision*, *Recall* y F1, respectivamente), lo que se repetirá en los tests de detección de EMs.

En resumen, los valores de todas las métricas son bastante altos, lo que prueba la viabilidad de las arquitecturas empleadas, si bien no son tan significativos estadísticamente en aquellos casos en los que no hay una arquitectura/configuración netamente superior. Esto es esperable y se repetirá en el resto de tests.

4.2 Detección

En esta tarea, para la que intentamos predecir EMs desconocidas en el entrenamiento, seguimos las mismas configuraciones de prueba y tests de significancia que en la subsección anterior, pero realizamos tres juegos de tests distintos, según las EMs que se procesan: todas, sólo NEs y sólo locuciones.

Todas las EMs. En la Tabla 3 presentamos los resultados para el primer caso, en el que mBERT obtiene los mejores valores de *Accuracy* (99,79 %), *Precision* (93,68 %) y *Recall* (94,94 %), XLM-R lidera en F1 (88,44 %), y mwetoolkit sigue obteniendo los mejores resultados en la mayoría de las configuraciones de prueba: mejor *Accuracy* en 4 de las 8, mejor *Precision* en 7 y mejor F1 en 6. Por su parte, mBERT obtiene los mejores valores de *Recall* en 4 de dichas configuraciones ($p = 0{,}334$).

Respecto a las opciones de configuración, "Sin amalgamas"+"Todo"+"Mays" obtiene la mejor *Precision* y F1, mientras que "Sin amalgamas"+"Todo"+"Mins" obtiene la mejor *Accuracy*, y el mejor *Recall* corresponde a "Con amalgamas"+"Solo EMs"+"Mins". Entrenar con amalgamas mejora los valores de *Precision*, *Recall* y F1 en el 75 % ($p = 0{,}281$), 68,75 % y 75 % de los casos, respectivamente. Sin embargo, empeora los valores de *Accuracy* en un 78,75 % de los tests. Por su parte, "Todo" da mejores valores de *Accuracy*, *Precision* y F1 en todos los tests, aunque "Solo EMs" siempre da lugar a mejores valores de *Recall*, una situación que ya encontrábamos en los tests de identificación. Finalmente, usar mayúsculas y minúsculas tal y como aparecen en CORGA también mejora los resultados de *Accuracy*, *Precision*, *Recall* y F1 en el 58,33 % ($p = 0{,}190$), 100 %, 83,33 % y 100 % de los tests.

En general, observamos un empeoramiento de los resultados con respecto a la tarea de identificación, lo que es razonable dado que estamos asegurando que todas las EMs presentes en el conjunto de prueba sean desconocidas durante el entrenamiento.

Sólo NEs multipalabra. Presentamos los tests correspondientes a este caso en la Tabla 4, con mBERT liderando en *Accuracy* (99,98 %), *Precision* (98,14 %) y valor F1 (98,16 %), mientras que XLM-R obtiene los mejores resultados en *Recall* (98,47 %). Sin embargo, mwetoolkit obtiene mejores valores que las arquitecturas *transformer* en *Accuracy*, *Precision* y F1 para 6 de las 8 configuraciones posibles, y mejor *Recall* en 4. A modo de comparación, mBERT obtiene el mejor *Recall* en 3 configuraciones, y lidera en el res-

			Con amalgamas				Sin amalgamas			
			Acc	P	R	F1	Acc	P	R	F1
Todo	Mays	mBERT	99,89	93,08	96,38	94,70	**99,98**	**98,14**	98,18	**98,16**
		XLM-R	99,90	94,04	96,14	95,08	99,97	98,08	98,13	98,11
		mwetoolkit	99,90	95,50	95,23	95,36	99,92	95,76	95,63	95,69
		Baseline	99,48	72,96	86,70	79,24	99,52	72,97	86,65	79,22
	Mins	mBERT	99,85	50,57	75,83	60,68	99,87	49,77	75,38	59,96
		XLM-R	99,84	47,09	76,05	58,17	99,86	48,55	75,32	59,04
		mwetoolkit	99,90	95,40	95,09	95,24	99,91	95,63	95,36	95,49
		Baseline	99,48	72,96	86,70	79,24	99,52	72,97	86,65	79,22
Solo EMs	Mays	mBERT	99,08	50,87	97,89	66,95	99,96	95,95	98,40	97,16
		XLM-R	99,63	73,78	97,82	84,12	99,94	92,57	**98,47**	95,43
		mwetoolkit	99,88	92,46	96,92	94,64	99,89	91,89	97,13	94,44
		Baseline	99,48	72,96	86,70	79,24	99,52	72,97	86,65	79,22
	Mins	mBERT	96,22	23,82	90,52	37,72	96,82	25,50	90,91	39,83
		XLM-R	96,45	25,41	90,47	39,68	96,96	26,73	90,10	41,22
		mwetoolkit	99,87	91,80	96,62	94,15	99,88	91,85	97,02	94,36
		Baseline	99,48	72,96	86,70	79,24	99,52	72,97	86,65	79,22

Tabla 4: Resultados de los tests de detección de NEs multipalabra.

to de métricas en sólo 2 casos.

De manera similar a lo que ocurría en los tests de identificación, los mejores valores de las métricas se encuentran en la configuración "Sin amalgamas"+"Todo"+"Mays", excepto en el caso de *Recall*, cuyo mejor caso es "Sin amalgamas"+"Solo EMs"+"Mays". Comparando cada opción de configuración, entrenar sin amalgamas mejora los valores de *Accuracy*, *Recall*, *Precision* y F1, respectivamente, en un 100 %, 87,5 %, 66,25 % y 72,5 % de los tests, mientras que entrenar con "Todo" permite obtener mejores valores de *Accuracy*, *Precision* y F1 en todos los tests, y entrenar con "Solo EMs" siempre genera mejores valores de *Recall*. Entrenar con "Mays" da lugar a mejores valores para todas las métricas respecto a usar sólo minúsculas. En general, los valores de las métricas tienden a ser mejores que en los tests previos.

Sólo locuciones. Este caso es, sin duda, el que arroja peores resultados, como se puede observar en la Tabla 5. La mejor *Accuracy* corresponde a mBERT (99,92 %), la mejor *Precision* (82,5 %) a mwetoolkit, y los mejores valores de *Recall* (73,88 %) y F1 (55,41 %) a XML-R. Todos los valores anteriores, excepto *Accuracy*, son sustancialmente más bajos que sus homólogos en los juegos de tests anteriores, y el mejor resultado de *Precision* se obtiene a costa de un *Recall* muy bajo (y viceversa). Además, no hay una arquitectura más consistente que las demás: mwetoolkit obtiene los mejores valores de *Accuracy* ($p = 0{,}390$) y *Precision* ($p = 0{,}232$) en 4 de las 8 configuraciones de test, XLM-R obtiene mejor *Recall* ($p = 0{,}106$) en 6, y mBERT ($p = 0{,}390$) y XML-R ($p = 0{,}232$) obtienen los mejores valores de F1 3 veces cada uno.

Con respecto a las configuraciones de prueba, entrenar con amalgamas mejora *Accuracy*, *Recall*, *Precision* y F1, respectivamente, en un 56,25 % ($p = 0{,}798$), 93,75 %, 100 % y 93,75 % de los tests. Esto es completamente opuesto a lo visto las pruebas anteriores, en las que usar entidades léxicas no amalgamadas mejoraba sustancialmente los resultados. Con respecto a las frases usadas en el entrenamiento, usar todas sólo ofrece mejoras en *Accuracy* (en todos los tests), mientras que usar sólo las frases con EMs mejora *Precision*, *Recall* y F1 en un 66,33 % ($p = 0{,}545$), 100 % y 87,33 % de los casos, respectivamente. De nuevo, ello contrasta fuertemente con lo que habíamos visto en tests anteriores. Finalmente, "Mays" mejora *Precision* y *Recall* (en ambos casos, en un 58,33 % de los tests, con $p = 0{,}238$ y $p = 0{,}193$, respectivamente), pero "Mins" ofrece mejores *Accuracy* y F1 (ambas en un 58,33 % de los tests, con $p = 0{,}361$ para la segunda).

Los resultados con locuciones son pobres y diferentes al resto de los tests, con menos significancia estadistica. Ello puede deberse a la mayor heterogeneidad de estas construcciones en relación con las NEs, al menor número de las mismas, y a que la lista de locuciones en gallego anotadas en CORGA no es necesariamente completa. Tampoco podemos descartar que los posibles errores de etiquetación morfosintáctica también tengan un efecto en el rendimiento.

			Con amalgamas				Sin amalgamas			
			Acc	P	R	F1	Acc	P	R	F1
Todo	Mays	mBERT	99,48	78,55	41,78	54,55	99,43	18,53	2,70	4,72
		XLM-R	99,48	78,54	38,76	51,90	99,44	17,31	2,74	4,73
		mwetoolkit	99,43	**82,05**	32,41	46,47	99,43	3,59	0,34	0,62
		Baseline	91,55	2,84	28,03	5,15	90,46	1,22	19,09	2,29
	Mins	mBERT	99,90	33,99	32,70	33,33	**99,92**	6,45	4,31	5,17
		XLM-R	99,91	37,87	38,71	38,28	99,92	5,90	4,49	5,10
		mwetoolkit	99,43	82,00	32,41	46,46	99,43	3,59	0,34	0,62
		Baseline	91,55	2,84	28,03	5,15	90,46	1,22	19,09	2,29
Solo EMs	Mays	mBERT	98,25	22,38	71,57	34,09	99,24	27,65	45,10	34,28
		XLM-R	98,52	25,27	**73,88**	37,65	99,26	24,32	27,16	25,66
		mwetoolkit	99,36	63,43	39,53	48,70	99,35	13,61	4,43	6,69
		Baseline	91,55	2,84	28,03	5,15	90,46	1,22	19,09	2,29
	Mins	mBERT	99,12	39,61	64,91	49,20	99,21	24,71	38,14	29,99
		XLM-R	99,29	47,74	66,02	**55,41**	99,28	28,25	39,09	32,80
		mwetoolkit	99,36	63,58	39,49	48,72	99,35	13,52	4,39	6,63
		Baseline	91,55	2,84	28,03	5,15	90,46	1,22	19,09	2,29

Tabla 5: Resultados de los tests de detección de locuciones.

5 Conclusiones y trabajo futuro

Podemos extraer algunas conclusiones preliminares de este trabajo. La primera de ellas es que las arquitecturas de AA consideradas parecen funcionar bastante bien en la tarea de identificación, si bien los resultados son más relevantes para NEs multipalabra que para locuciones, dado que estas últimas son conocidas durante el entrenamiento casi en su totalidad. En la tarea de detección, los resultados son prometedores para NEs, y algo decepcionantes para las locuciones.

En lo referente a las arquitecturas consideradas, mwetoolkit ha funcionado mejor de lo esperado en relación a las modelos *transformer*, si bien se beneficia de una información (etiquetas morfosintácticas y lemas) no siempre disponible. Además, el gallego sólo representa una pequeña parte de los recursos usados para entrenar dichos modelos *transformer*. Por otra parte, en la tarea de detección, mBERT tiende a obtener los valores más altos en las métricas consideradas más veces que XLM-R, a pesar de usar un modelo más pequeño y menos complejo.

Con respecto a las posibles configuraciones de CORGA para el entrenamiento, excluyendo la detección de locuciones, los resultados de los tests indican que: a) se tiende a obtener los mejores resultados entrenando con entidades léxicas sin amalgamar, aunque en la mayor parte de las configuraciones usar amalgamas parece beneficioso; b) incluir texto de entrenamiento sin EMs parece mejorar los resultados en *Precision* y F1, pero empeora los de *Recall*; c) por lo tanto, si nos interesa recuperar el mayor número posible de EM a expensas de la *Precision*, entrenar sólo con frases que contengan EMs puede ser buena idea; y d) pasar a minúsculas el texto no parece mejorar los resultados, incluso en una arquitectura (mBERT) específicamente preentrenada con texto en minúsculas.

En lo referente a la identificación de locuciones, nuestra intención es estudiar por qué los resultados tienen a ser tan inferiores en estos tests. También queremos comprobar hasta qué punto la optimización de hiperparámetros puede mejorar el rendimiento en los modelos *transformer*. Para ello, nos proponemos hacer dicha optimización en los tests en los que mBERT y XLM-R hayan sacado peores resultados.

Finalmente, hasta ahora sólo hemos trabajado con la versión de CORGA etiquetada automáticamente. Nos proponemos hacer experimentos con la porción del mismo que ha sido corregida de manera manual y utilizada para entrenar XIADA, para comprobar si usar un corpus corregido manualmente tiene algún impacto en los resultados.

Agradecimientos

Este trabajo ha sido parcialmente financiado por la Xunta de Galicia, a través del Convenio de colaboración plurianual entre el Centro Ramón Piñeiro para la Investigación en Humanidades y la Universidad de Vigo, y la Ayuda para la Consolidación y Estructuración de Unidades de Investigación Competitivas ED431C 2018/50, y por el Ministerio de Economía, Industria y Competitividad a través del proyecto TIN2017-85160-C2-2-R.

Bibliografía

Blunsom, P. y T. Baldwin. 2006. Multilingual deep lexical acquisition for HPSGs via supertagging. En *Proceedings of the 2006 Conference on Empirical Methods in Natural Language Processing*, páginas 164–171, Sydney, Australia, Julio. Association for Computational Linguistics.

Candito, M. y M. Constant. 2014. Strategies for Contiguous Multiword Expression Analysis and Dependency Parsing. En *ACL'14 - The 52nd Annual Meeting of the Association for Computational Linguistics*, Baltimore, United States. ACL.

Centro Ramón Piñeiro para a Investigación en Humanidades. 2019a. Corpus de Referencia do Galego Actual (CORGA) [v3.2]. http://corpus.cirp.gal/corga/.

Centro Ramón Piñeiro para a Investigación en Humanidades. 2019b. Etiquetador/Lematizador do Galego Actual (XIADA) [v2.7]. http://corpus.cirp.gal/xiada/.

Cho, K., B. van Merriënboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, y Y. Bengio. 2014. Learning phrase representations using RNN encoder–decoder for statistical machine translation. En *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing*, páginas 1724–1734, Doha, Qatar, Octubre. Association for Computational Linguistics.

Conneau, A., K. Khandelwal, N. Goyal, V. Chaudhary, G. Wenzek, F. Guzmán, E. Grave, M. Ott, L. Zettlemoyer, y V. Stoyanov. 2020. Unsupervised cross-lingual representation learning at scale. En *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, páginas 8440–8451, Online, Julio. Association for Computational Linguistics.

Constant, M., G. Eryiğit, J. Monti, L. van der Plas, C. Ramisch, M. Rosner, y A. Todirascu. 2017. Multiword Expression Processing: A Survey. *Computational Linguistics*, 43(4):837–892.

Constant, M. y J. Nivre. 2016. A transition-based system for joint lexical and syntactic analysis. En *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, páginas 161–171, Berlin, Germany, Agosto. Association for Computational Linguistics.

Constant, M. y A. Sigogne. 2011. MWU-aware part-of-speech tagging with a CRF model and lexical resources. En *Proceedings of the Workshop on Multiword Expressions: from Parsing and Generation to the Real World*, páginas 49–56, Portland, Oregon, USA, Junio. Association for Computational Linguistics.

Constant, M., A. Sigogne, y P. Watrin. 2012. Discriminative strategies to integrate multiword expression recognition and parsing. En *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, páginas 204–212, Jeju Island, Korea, Julio. Association for Computational Linguistics.

Devlin, J., M.-W. Chang, K. Lee, y K. Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. En *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, páginas 4171–4186, Minneapolis, Minnesota, Junio. Association for Computational Linguistics.

Diab, M. y P. Bhutada. 2009. Verb noun construction MWE token classification. En *Proceedings of the Workshop on Multiword Expressions: Identification, Interpretation, Disambiguation and Applications*, páginas 17–22, Singapore, Agosto. Association for Computational Linguistics.

Domínguez Noya, E. M., F. M. Barcala Rodríguez, y M. Á. Molinero Álvarez. 2009. Avaliación dun etiquetador automático estatístico para o galego actual: Xiada. *Cadernos de Lingua*, 30-31:151–193.

Domínguez Noya, E. M., M. S. López Martínez, y F. M. Barcala Rodríguez. 2019. O Corpus de Referencia do Galego actual (CORGA): composición, codificación, etiquetaxe e explotación. *Verba: Anuario Galego de Filoloxía*, Anexo 74:179–219.

Dubremetz, M. y J. Nivre. 2014. Extraction of nominal multiword expressions in French. En *Proceedings of the 10th Workshop on Multiword Expressions (MWE)*, páginas 72–76, Gothenburg, Sweden, Abril. Association for Computational Linguistics.

Farahmand, M. y R. Martins. 2014. A supervised model for extraction of multiword expressions, based on statistical context features. En *Proceedings of the 10th Workshop on Multiword Expressions (MWE)*, páginas 10–16, Gothenburg, Sweden, Abril. Association for Computational Linguistics.

Firth, J. R. 1957. *Papers in Linguistics, 1934-1951*. Oxford University Press, London.

Green, S., M.-C. de Marneffe, J. Bauer, y C. D. Manning. 2011. Multiword expression identification with tree substitution grammars: A parsing tour de force with French. En *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, páginas 725–735, Edinburgh, Scotland, UK., Julio. Association for Computational Linguistics.

Green, S., M.-C. de Marneffe, y C. D. Manning. 2013. Parsing models for identifying multiword expressions. *Computational Linguistics*, 39(1):195–227.

Hochreiter, S. y J. Schmidhuber. 1997. Long short-term memory. *Neural Computation*, 9(8):1735–1780, 11.

Jackendoff, R. 1997. Twistin' the night away. *Language*, 73(3):534–559, Septiembre.

Klyueva, N., A. Doucet, y M. Straka. 2017. Neural networks for multi-word expression detection. En *Proceedings of the 13th Workshop on Multiword Expressions*, páginas 60–65, Valencia, Spain, Abril. Association for Computational Linguistics.

Kurfalı, M. 2020. TRAVIS at PARSEME shared task 2020: How good is (m)BERT at seeing the unseen? En *Proceedings of the Joint Workshop on Multiword Expressions and Electronic Lexicons*, páginas 136–141, online, Diciembre. Association for Computational Linguistics.

Lafferty, J. D., A. McCallum, y F. C. N. Pereira. 2001. Conditional Random Fields: Probabilistic Models for Segmenting and Labeling Sequence Data. En *Proceedings of the Eighteenth International Conference on Machine Learning*, ICML '01, páginas 282–289, San Francisco, CA, USA. Morgan Kaufmann Publishers Inc.

Lapata, M. y A. Lascarides. 2003. Detecting novel compounds: The role of distributional evidence. En *10th Conference of the European Chapter of the Association for Computational Linguistics*, Budapest, Hungary, Abril. Association for Computational Linguistics.

Legrand, J. y R. Collobert. 2016. Phrase representations for multiword expressions. En *Proceedings of the 12th Workshop on Multiword Expressions*, páginas 67–71, Berlin, Germany, Agosto. Association for Computational Linguistics.

Maldonado, A., L. Han, E. Moreau, A. Alsulaimani, K. D. Chowdhury, C. Vogel, y Q. Liu. 2017. Detection of verbal multiword expressions via conditional random fields with syntactic dependency features and semantic re-ranking. En *Proceedings of the 13th Workshop on Multiword Expressions*, páginas 114–120, Valencia, Spain, Abril. Association for Computational Linguistics.

Okazaki, N. 2007. CRFsuite: a fast implementation of Conditional Random Fields (CRFs). http://www.chokkan.org/software/crfsuite/.

Pecina, P. 2009. *Lexical Association Measures: Collocation Extraction*. ÚFAL, Praha, Czechia.

Ramisch, C. 2015. *Multiword Expressions Acquisition: A Generic and Open Framework*, volumen XIV de *Theory and Applications of Natural Language Processing*. Springer.

Ramisch, C., A. Villavicencio, L. Moura, y M. Idiart. 2008. Picking them up and figuring them out: Verb-particle constructions, noise and idiomaticity. En A. Clark y K. Toutanova, editores, *Proceedings of the Twelfth Conference on Natural Language Learning*, páginas 49–56, Manchester, UK. ACL.

Ramshaw, L. A. y M. Marcus. 1995. Text chunking using transformation-based learning. En D. Yarowsky y K. Church, edito-

res, *Third Workshop on Very Large Corpora, VLC@ACL 1995, Cambridge, Massachusetts, USA, June 30, 1995*.

Riedl, M. y C. Biemann. 2016. Impact of MWE resources on multiword recognition. En *Proceedings of the 12th Workshop on Multiword Expressions*, páginas 107–111, Berlin, Germany, Agosto. Association for Computational Linguistics.

Rondon, A., H. Caseli, y C. Ramisch. 2015. Never-ending multiword expressions learning. En *Proceedings of the 11th Workshop on MWEs*, páginas 45–53, Denver, CO, USA. ACL.

Schneider, N., E. Danchik, C. Dyer, y N. A. Smith. 2014. Discriminative lexical semantic segmentation with gaps: Running the MWE gamut. *Transactions of the Association for Computational Linguistics*, 2:193–206.

Schneider, N., D. Hovy, A. Johannsen, y M. Carpuat. 2016. SemEval-2016 task 10: Detecting minimal semantic units and their meanings (DiMSUM). En *Proceedings of the 10th International Workshop on Semantic Evaluation (SemEval-2016)*, páginas 546–559, San Diego, California, Junio. Association for Computational Linguistics.

Simkó, K. I., V. Kovács, y V. Vincze. 2017. USzeged: Identifying verbal multiword expressions with POS tagging and parsing techniques. En *Proceedings of the 13th Workshop on Multiword Expressions*, páginas 48–53, Valencia, Spain, Abril. Association for Computational Linguistics.

Taslimipoor, S., S. Bahaadini, y E. Kochmar. 2020. MTLB-STRUCT @Parseme 2020: Capturing unseen multiword expressions using multi-task learning and pre-trained masked language models. En *Proceedings of the Joint Workshop on Multiword Expressions and Electronic Lexicons*, páginas 142–148, online, Diciembre. Association for Computational Linguistics.

Taslimipoor, S. y O. Rohanian. 2018. SHOMA at parseme shared task on automatic identification of vmwes: Neural multiword expression tagging with high generalisation. *CoRR*, abs/1809.03056.

Vincze, V., I. Nagy T., y G. Berend. 2011. Multiword expressions and named entities in the wiki50 corpus. En *Proceedings of the International Conference Recent Advances in Natural Language Processing 2011*, páginas 289–295, Hissar, Bulgaria, Septiembre. Association for Computational Linguistics.

Vincze, V., J. Zsibrita, y I. Nagy T. 2013. Dependency parsing for identifying Hungarian light verb constructions. En *Proceedings of the Sixth International Joint Conference on Natural Language Processing*, páginas 207–215, Nagoya, Japan, Octubre. Asian Federation of Natural Language Processing.

Wolf, T., L. Debut, V. Sanh, J. Chaumond, C. Delangue, A. Moi, P. Cistac, T. Rault, R. Louf, M. Funtowicz, J. Davison, S. Shleifer, P. von Platen, C. Ma, Y. Jernite, J. Plu, C. Xu, T. L. Scao, S. Gugger, M. Drame, Q. Lhoest, y A. M. Rush. 2020. Transformers: State-of-the-art natural language processing. En *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, páginas 38–45, Online, Octubre. Association for Computational Linguistics.

Zampieri, N., M. Scholivet, C. Ramisch, y B. Favre. 2018. Veyn at PARSEME shared task 2018: Recurrent neural networks for VMWE identification. En *Proceedings of the Joint Workshop on Linguistic Annotation, Multiword Expressions and Constructions*, páginas 290–296, Santa Fe, New Mexico, USA, Agosto. Association for Computational Linguistics.