UH-MatCom en eHealth-KD Challenge 2020: Modelos de aprendizaje profundo y conjuntos para el conocimiento Descubrimiento en documentos españoles

Juan Pablo Consuegra-Ayalaa , Manuel Palomarb,c

a Facultad de Matemáticas e Informática, Universidad de la Habana, 10200 La Habana, Cuba bInstituto Universitario de Investigaciones en Computación (IUII), Universidad de Alicante, 03690 Alicante, España cDepartamento de Lenguaje y Sistemas Informáticos, Universidad de Alicante, 03690 Alicante, España

Resumen El

reto eHealth-KD organizado en IberLEF 2020 propone un conjunto de recursos y escenarios de evaluación para fomentar el desarrollo de sistemas de extracción automática de conocimiento a partir de textos no estructurados. Este artículo describe el sistema presentado por el equipo UH-MatCom en el desafío. Se entrenan y ensamblan varios modelos de aprendizaje profundo para extraer automáticamente entidades y relaciones relevantes de documentos de texto sin formato. Se aplican técnicas de última generación como BERT, Bi-LSTM y CRF. Se explora el uso de fuentes de conocimiento externas como ConceptNet. El sistema logró resultados promedio en el desafío, ocupando el quinto lugar en todos los diferentes escenarios de evaluación. El método conjunto produjo una ligera mejora en el rendimiento. Es necesario realizar trabajo adicional para que la tarea de extracción de relaciones se beneficie con éxito de fuentes de conocimiento externas.

Palabras clave

eSalud, descubrimiento de conocimientos, procesamiento del lenguaje natural, aprendizaje automático, reconocimiento de entidades, Extracción de relaciones

1. Introducción

Este artículo describe el sistema presentado por el equipo UH-MatCom en el desafío eHealth-KD en IberLEF 2020 [1]. El desafío propone un conjunto de escenarios y corpus de evaluación para fomentar el desarrollo de sistemas para la extracción automática de conocimiento a partir de textos no estructurados.
Se solicita identificar tanto las entidades relevantes como las relaciones entre ellas que ocurren en documentos de texto plano . Aunque los documentos relacionados con la salud son la principal fuente de texto, el conocimiento extraído es de propósito general y los documentos no relacionados con la salud también estaban disponibles como colecciones de evaluación adicionales.

Nuestro sistema utiliza varios modelos con diferentes arquitecturas, que fueron evaluados en tres ejecuciones diferentes. Dos de las ejecuciones presentadas consisten en arquitecturas basadas en aprendizaje profundo, que utilizan técnicas de PNL de última generación para el reconocimiento de entidades nombradas (NER) y la extracción de relaciones. Las capas BERT [2], CRF [3] y Bi-LSTM [4] son los componentes principales de estos modelos. La primera arquitectura utiliza conocimientos de dominio general extraídos de ConceptNet [5]

Actas del Foro de Evaluación de Lenguas Ibéricas (IberLEF 2020) correo electrónico: jpconsuegra@matcom.uh.cu (JP Suegra); mpalomar@dlsi.ua.es (M. Paloma) Orcid: 0000-0003-2009-393X (JP Consuegra-Ayala)

ISSN 1613-0073 Actas del taller CEUR (CEUR-WS.org)

enriquecer la información sobre pares de entidades en la tarea de extracción de relaciones. El segundo utiliza características sintácticas adicionales, como información del árbol de dependencia en lugar de ConceptNet, para codificar la relación entre pares de entidades. La tercera ejecución combina estos dos modelos y dos adicionales para producir un resultado agregado. Los parámetros del conjunto se ajustan automáticamente de acuerdo con la colección de capacitación y desarrollo.

El resto del artículo está organizado de la siguiente manera. Las secciones 2 y 3 describen las diferentes arquitecturas utilizadas por el sistema y el método de conjunto aplicado. En la Sección 4 se presentan los resultados oficiales logrados en el desafío. En la Sección 5, se comparten algunas ideas encontradas sobre la calidad de cada estrategia. Finalmente, la Sección 6 presenta las conclusiones del artículo junto con algunas recomendaciones de trabajo futuro.

2. Descripción del sistema

Nuestro sistema utiliza dos arquitecturas principales, a partir de las cuales se encienden o apagan varios componentes, para producir diferentes variantes. Ambas arquitecturas se basan en el aprendizaje profundo y en adelante se hará referencia a ellas como etiquetador BILUOV y clasificador por pares. La distinción entre las dos arquitecturas principales surge del tipo de problema que resuelven. Cada uno se utiliza de forma independiente para resolver la tarea correspondiente del desafío.

El etiquetador BILUOV resuelve un problema de etiquetado de secuencia, en el que a cada token de una secuencia de entrada se le asigna una etiqueta del esquema de etiquetado de entidades BILUOV [6] . Este modelo se utiliza para resolver la Tarea A del desafío, orientada a extraer las entidades relevantes mencionadas en una oración de texto plano. Los cuatro tipos de entidades propuestas en el desafío (es decir, Concepto, Acción, Predicado y Referencia) se manejan de forma independiente. Se entrena una instancia de etiquetador BILUOV para cada tipo de entidad. Opcionalmente, estas instancias pueden compartir capas intermedias durante el entrenamiento, un parámetro que fue explorado y optimizado durante la fase de desarrollo del desafío.

El clasificador por pares resuelve un problema de clasificación de múltiples clases, en el que se procesa una secuencia de tokens y luego se le asigna una única etiqueta. Este modelo se utiliza para resolver la Tarea B del desafío, orientada a identificar las relaciones relevantes (si las hay) entre las entidades previamente extraídas. Una única instancia de este modelo está entrenada para manejar todo tipo de relaciones. Opcionalmente, este modelo puede reutilizar algunas de las capas previamente entrenadas del etiquetador BILUOV, dependiendo de si los modelos de extracción de entidades compartieron esas capas o no.

2.1. Manejo de entrada

Ambos modelos funcionan en el caso más general sobre secuencias de tokens. El etiquetador BILUOV recibe las oraciones tokenizadas tal como están y produce una salida para cada token en la secuencia de entrada. El etiquetador por pares recibe la ruta de los tokens en el árbol de dependencia de la oración entre cada par de entidades que aparecen en la oración (o entre su ancestro común más bajo (LCA) en el caso de entidades de varias palabras). Además, las dos secuencias de tokens que forman las entidades fuente y principal, respectivamente, también se proporcionan al etiquetador por pares con su correspondiente tipo de entidad previamente asignado. Las Figuras 1 y 2 muestran la forma en que cada arquitectura transforma las entradas .

1https://devopedia.org/named-entity-recognition (Consultado el 1 de mayo de 2020).

2.1.1. Común a ambas tareas

Cada oración de texto sin formato se tokeniza a nivel de palabra. Se extraen las siguientes características para cada token. La Figura 1 ilustra cómo se utilizan directamente en el etiquetador BILUOV.

- Representación de caracteres, que consiste en asignar un valor entero a cada carácter del token según su índice en un vocabulario predefinido. El vocabulario seleccionado fue uno de todas las letras, dígitos y símbolos de puntuación ASCII. Se agrega relleno al final para garantizar que todos los tokens tengan la misma cantidad de caracteres.

- Word Embedding, que consiste en una representación vectorial del token en un espacio semántico continuo. Se consideraron incrustaciones de palabras clásicas previamente entrenadas o contextuales, como BERT .

- Representación de etiquetas POS, que consiste en una codificación one-hot de la etiqueta Parte del discurso de la ficha.

2.1.2. Detalles de la tarea B

Además de las características anteriores, se incluyen algunas características adicionales en cada token para el clasificador por pares. La Figura 2 ilustra cómo se utilizan directamente en el clasificador por pares.

- Representación del árbol de dependencia, que consiste en una codificación one-hot de la dependencia. etiqueta de decencia del token en el árbol de la oración.

- Representación del tipo de entidad, que consiste en una codificación one-hot del tipo de entidad del token según la etiqueta que le fue asignada en la tarea de extracción de entidades previamente finalizada. Algunos tokens no pertenecen a ninguna entidad, por lo que se les asigna una etiqueta adicional No es una entidad (NaE). No se tuvo en cuenta ninguna consideración especial para los tokens que pertenecen a múltiples entidades, sino que se seleccionó una etiqueta arbitraria entre los candidatos.

La representación del árbol de dependencia solo se usa para los tokens que forman la ruta entre las entidades fuente y principal de la relación. De manera similar, la representación del tipo de entidad no se proporciona para la secuencia de tokens fuente y principal, sino que se proporciona como una característica adicional para toda la secuencia.

2.1.3. ConceptoNet

Finalmente, el conocimiento externo de ConceptNet sobre las entidades fuente y principal de la relación se proporciona opcionalmente al clasificador por pares. La información se captura en 36 relaciones ConceptNet [5] divididas en tres rangos de coincidencia diferentes.

El rango de coincidencia directa representa las relaciones encontradas entre las entidades exactas (posiblemente múltiples). word) en ConceptNet o sus versiones lematizadas.

El rango de coincidencia relacionada representa las relaciones encontradas entre las entidades exactas, sus lemas o palabras que pertenecen a la misma clase de equivalencia que ellas según las relaciones SimilarTo, Synonym, RelatedTo y EtymologicallyDerivedFrom de ConceptNet . El uso de este

Este tipo de coincidencia permite verificar las relaciones entre idiomas, lo cual es crucial para aprovechar al máximo el conocimiento contenido en ConceptNet, ya que algunas relaciones solo existen entre sus equivalentes en inglés.

El rango de coincidencia parcial representa coincidencias directas y relacionadas encontradas entre el individuo palabras que componen las entidades.

Esta información está codificada en un vector k-hot cuyos componentes activos (es decir, establecidos en 1) son los correspondientes a las relaciones para las que se encontró una coincidencia. Sólo el subconjunto de relaciones que ocurren entre términos en español y/o inglés fue preprocesado debido a limitaciones de recursos computacionales. Sin embargo, esta información es suficiente en el contexto del desafío ya que todos los documentos están escritos en español y el subconjunto en inglés permite capturar las relaciones más relevantes entre términos en español a través de relaciones de equivalencia.

2.2. Modelo de reconocimiento de entidades

Para resolver la tarea de reconocimiento de entidades, se utiliza la arquitectura de etiquetado BILUOV . La Figura 1 resume la arquitectura compartida por estos modelos.

En primera instancia, se utiliza una capa de codificación de caracteres para transformar la representación de caracteres de cada token en un único vector que captura las dependencias morfológicas del token. Esta técnica ha demostrado ser eficaz para tratar palabras fuera del vocabulario que comparten algunas similitudes con palabras vistas anteriormente. La capa de codificación de caracteres consta de una capa de incrustación (A) (una tabla de búsqueda) seguida de una capa Bi-LSTM (B). El Bi-LSTM consume la secuencia de caracteres incrustados y produce la concatenación de las salidas del LSTM interno.

La representación a nivel de caracteres calculada previamente, la incrustación de palabras y la representación de etiqueta POS se concatenan para cada token en la secuencia de entrada. Estos vectores son procesados por una lista de capas Bi-LSTM apiladas (C) para producir una secuencia de vectores que codifican los tokens en la oración de entrada. El número de capas apiladas es un parámetro a optimizar, pero durante el desafío todas las evaluaciones se realizaron con una sola capa.

Finalmente, se aplican una capa densa lineal y CRF (D) para transformar cada vector de token en la etiqueta más probable correspondiente en el esquema de etiquetado BILUOV.

2.2.1. Tarea A

Se entrenan cuatro instancias independientes con esta arquitectura, cada una correspondiente a un tipo de entidad. Hacerlo tiene dos beneficios principales. En primer lugar, esto simplifica la codificación y decodificación de entidades en la oración, ya que no es necesario diferenciar las etiquetas BILUOV entre tipos de entidades en las mismas oraciones. En segundo lugar, permite modelar la superposición entre entidades de diferentes tipos sin consideraciones adicionales. Por otro lado, dividir la tarea en cuatro subproblemas tiene la desventaja de reducir los datos de entrenamiento relevantes para cada modelo (es decir, aumenta el número de anotaciones en blanco por oración), y los errores al predecir con cada modelo se acumulan y propagan.

Para mitigar los inconvenientes anteriores, la capa de codificación de caracteres se comparte opcionalmente entre modelos y se entrena de forma conjunta. Esto permite beneficiarse de ejemplos de entrenamiento adicionales para ciertos componentes de los modelos mientras se continúa modelando diferentes espacios de hipótesis en cada modelo.

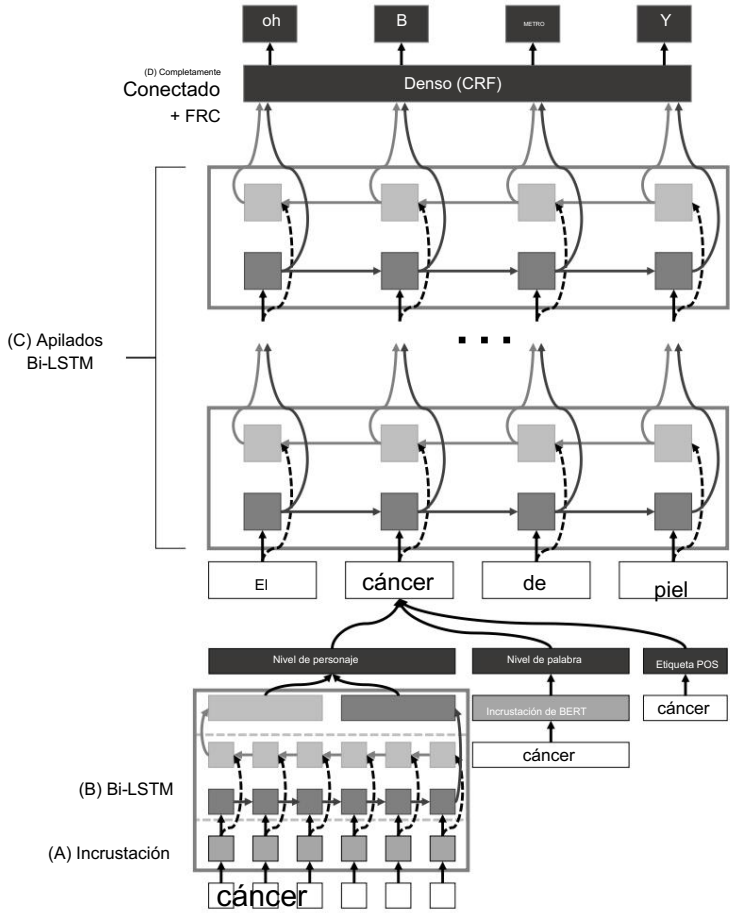


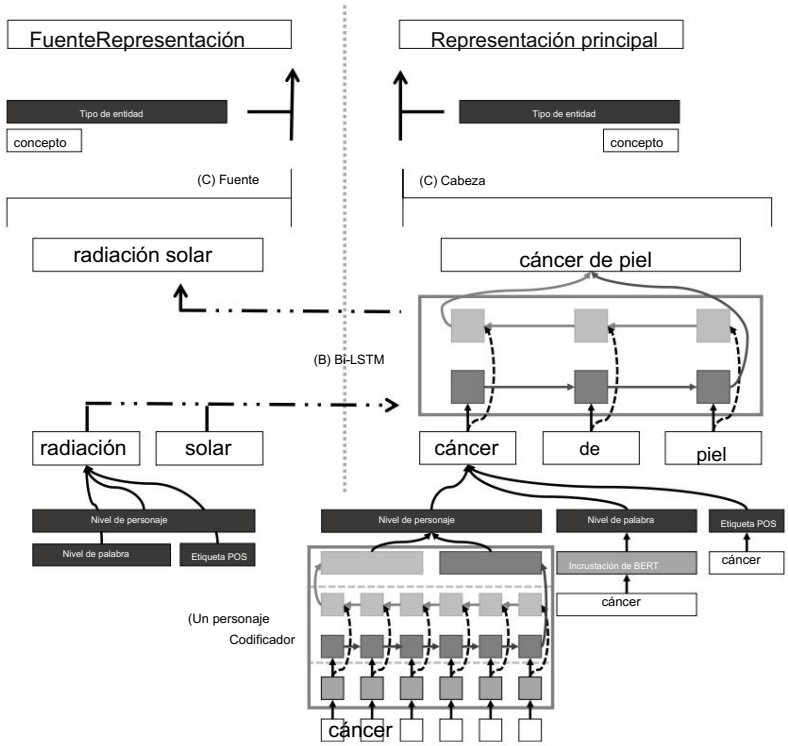
Figura 1: Resumen de la arquitectura del etiquetador BILUOV . Una oración de ejemplo (“El cáncer de piel”) es procesada por el etiquetador BILUOV que se creó para el tipo de entidad Concept. El modelo genera una secuencia de etiquetas (OBME), que luego se decodifica para obtener los Conceptos mencionados (solo “cáncer de piel” en este ejemplo).

La opción de compartir capas es un parámetro del sistema, pero en el contexto del desafío, todos los modelos probados fueron entrenados conjuntamente debido a una mejora en el rendimiento observada empíricamente. Opcionalmente, se agregan capas de abandono durante el entrenamiento entre cada par de capas contiguas.

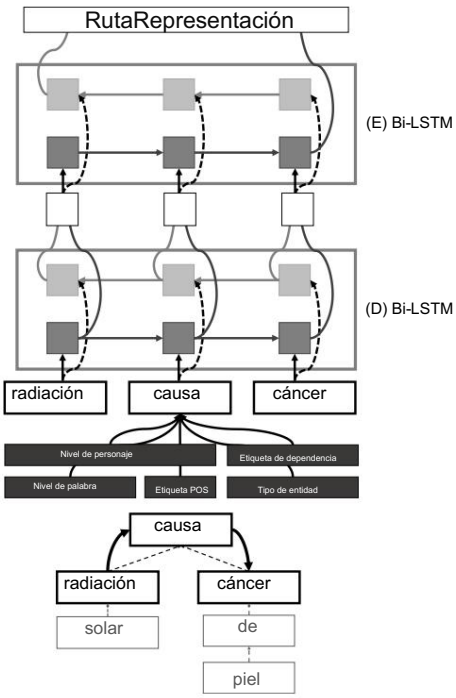
2.3. Modelo de extracción de relaciones

Para resolver la tarea de extracción de relaciones, se utiliza la arquitectura del clasificador por pares . La Figura 2 resume la arquitectura utilizada por este modelo. La arquitectura se divide en tres etapas: codificación de las entidades fuente y principal (Figura 2a), codificación de la ruta entre las entidades (Figura 2b) y combinación de representación y predicción (Figura 2c).

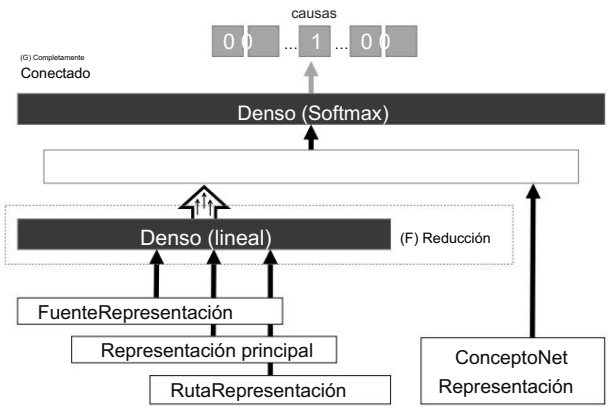
De manera similar al etiquetador BILUOV, el clasificador por pares utiliza una capa de codificador de caracteres (A) para transformar la representación de caracteres de un token en un único vector. En caso de que los cuatro modelos de reconocimiento de entidades se hayan entrenado compartiendo esta capa, los pesos calculados previamente se congelan.



(a) Fragmento del clasificador por pares responsable de calcular la representación de las entidades fuente y principal.
ciones.



(b) Fragmento del clasificador por pares responsable de
codificar el camino entre los
entidades fuente y cabecera.



(c) Fragmento del clasificador por pares responsable de
combinando todas las representaciones para finalmente predecir la
relación más probable mencionada sobre la fuente
y entidades cabecera.

Figura 2: Resumen de la arquitectura del clasificador por pares .

y reutilizado aquí. En otro caso, se crea y entrena una nueva capa de codificador de caracteres.

Esta arquitectura utiliza tres capas Bi-LSTM para codificar los tokens y producir representaciones intermedias que capturan dependencias entre pares de entidades. El primer Bi-LSTM (B) consume la secuencia de tokens que formaron la entidad de origen (y la entidad principal) para producir una representación compacta de la entidad. La representación del tipo de entidad de las entidades fuente y principal se concatena a los vectores obtenidos previamente (C). El segundo Bi-LSTM (D) procesa los tokens en la ruta entre los tokens fuente y principal (es decir, el LCA de los tokens de sus respectivas entidades) sobre el árbol de dependencia y los transforma en una representación oculta. Luego, el tercer Bi-LSTM (E) consume las representaciones ocultas para producir un vector que codifica toda la ruta.

La representación codificada de las entidades principal y fuente y la ruta entre ellas están concatenadas. Posteriormente se incluye una capa densa lineal opcional (F) para hacer que el vector concatenado coincida con tres veces el tamaño de la representación de ConceptNet (si se proporciona). Esta reducción de dimensiones se aplicó en algunos casos buscando que la representación de ConceptNet se considerara tanto como otras características. El vector resultante es procesado por una capa densa lineal final (G), que produce los logits para el tipo de relación más probable (si existe alguna) entre el par en cuestión.

2.3.1. Tarea B

Se entrena un único modelo con esta arquitectura para resolver la tarea. Se prueban todos los pares de entidades que aparecen en la misma oración. Opcionalmente, se puede omitir el cálculo de la ruta de dependencia para considerar solo la información de las entidades fuente y principal. Dado que calcular la ruta es una tarea costosa de realizar, la ejecución del desafío que considera esta información no pudo completarse a tiempo para el primer y cuarto escenario de evaluación, y en ese caso se utilizó la solución de referencia.

La falta de relaciones entre un par de entidades se modela con un tipo de relación adicional. Esta etiqueta es la más representativa de toda la colección formativa ya que sólo unas pocas entidades están relacionadas respecto a todos los pares posibles. Para mitigar el desequilibrio del conjunto de datos obtenido, opcionalmente empleamos un esquema de ponderación orientado a clases durante la fase de entrenamiento. De esta manera, el modelo consigue “prestar más atención” a muestras de una clase subrepresentada.

2.4. Manejo de salida

El etiquetador BILUOV genera una secuencia de etiquetas BILUOV. Estas etiquetas significan: Comenzar, para marcar el inicio de una entidad; Interior, para marcar su continuación; Por último, para marcar su fin; Unidad, para representar entidades de un solo token; y Otros, para representar tokens que no pertenecen a ninguna entidad. Además, la etiqueta oVerlapping se utiliza para tratar tokens que pertenecen a varias entidades. Por ejemplo, en la oración “El cáncer de estómago y de esófago causan problemas”, el modelo que detecta Conceptos debe generar: OVILOILOU, resultando en la detección de “cáncer de estómago”, “cáncer de esófago” y “problemas”.

El proceso de decodificación se maneja en dos fases, basado en la metodología descrita por el equipo UH-MAJA-KD en la edición anterior del desafío [7]. Primero, entidades discontinuas.

tabla 1

Resumen de los parámetros de bajo nivel utilizados para entrenar las variaciones de los modelos concretos que se utilizaron en el desafío. Las cuatro variaciones utilizan el mismo conjunto de parámetros.

Parámetros comunes		Etiquetas BILUOV		Clasificador por pares	
char_vocab_size	96	num_labels	6	num_labels	14
char_embedding_dim 100		abandono	0.0	dep_repr_dim 29	
padding_idx 0		stacked_layers	1	entidad_repr_dim 5	
char_repr_dim 200				subárbol_repr_dim 300	
palabra_repr_dim 768				token_repr_dim 300	
postag_repr_dim	19				
token_repr_dim	300				

Tabla 2

Resumen de los parámetros de alto nivel de los cuatro modelos que fueron entrenados y utilizados en el desafío.

Modelo	peso	conjunto	cnet	camino	reducir	
cnet	x	x	x			
deptree	x	x		x		
cnet-deptree	x	x	x	x		
solo-bert	x	x	x			x

Se extraen las que coincidan con cualquiera de las siguientes expresiones regulares.

( ) + ((|)) + || ((|)) + ( ) +

Posteriormente, todas las entidades restantes se consideran secuencias continuas de tokens. El
El proceso de codificación y decodificación se probó para medir el error causado por simplemente cambiar el representación. Esto resultó en un 1 de 0,997, un retiro de 0,998 y una precisión de 0,995, en el colección de entrenamiento.

Para resolver la Tarea A, se pide a los cuatro etiquetadores BILUOV que produzcan la secuencia BILUOV para cada oración. Cada secuencia de salida se decodifica de forma independiente para obtener un conjunto de entidades. Después, todas las entidades se fusionan en una sola colección y se informan como salida.

Para resolver la Tarea B, se construyen todos los pares de entidades que pertenecen a la misma oración. Después, Se pide al clasificador por pares que prediga la relación más probable entre cada par. Todo Las relaciones no vacías se recopilan y reportan como salida.

2.5. Entrenamiento del sistema

Se entrenaron y canalizaron cuatro variaciones concretas de las arquitecturas descritas anteriormente. Todos ellos comparten los parámetros que se muestran en la Tabla 1. La diferencia entre ellos viene dada por la selección de decisiones opcionales. La Tabla 2 resume la configuración de alto nivel de cada uno. de ellos.

El modelo cnet ignora la ruta entre las entidades fuente y principal en el clasificador por pares. y desactiva la capa de reducción. El modelo deptree no utiliza la información de ConceptNet.

y desactivar la capa de reducción. El modelo cnet-deptree combina ambas estrategias.

El modelo only-bert utiliza una versión simplificada del clasificador por pares en el que solo se utiliza la información de ConceptNet y el vector de incrustación BERT de los tokens fuente y principal. Se ignoran la ruta entre ellos, la incrustación de caracteres, la representación de etiquetas POS y la representación del tipo de entidad . En este caso se aplica la capa de reducción ya comentada del clasificador por pares .

Para entrenar los modelos solo se utilizó la colección de entrenamiento proporcionada en el desafío. La colección de desarrollo se utilizó durante el entrenamiento como colección de validación para evitar el sobreajuste en la colección de entrenamiento. La colección de frases adicionales no se utilizó en absoluto. Para el escenario 4 no se tuvieron en cuenta consideraciones particulares (más allá de lo hecho en los demás escenarios). El uso de ConceptNet como fuente externa de conocimiento estuvo motivado por la idea de lograr un mejor desempeño en términos no vistos durante la formación y, por tanto, en dominios no relacionados con la salud.

Las oraciones se tokenizaron y analizaron utilizando la biblioteca de Python spaCy2 (v2.2.4). Tanto el árbol de dependencia, la lematización y la información de etiquetas POS se extrajeron de allí, utilizando el modelo previamente entrenado es_core_news_md . Se utilizó un modelo multilingüe BERT3 previamente entrenado para calcular previamente los vectores de incrustación para cada token en la oración. No se aplicó ningún proceso de ajuste . Un único token espacioso se traduce en múltiples tokens BERT (BERT utiliza la tokenización de WordPieza [8]). Los vectores de incrustación obtenidos de BERT se asignaron al token de spacy eligiendo el primer token de BERT que se deriva del token de spacy. Promediar los vectores, en lugar de elegir el primero, produjo un desempeño ligeramente peor.

Todos los modelos fueron entrenados durante 100 épocas como máximo, pero en la práctica, convergieron a su precisión final en alrededor de 10 o 20 épocas. Los modelos se implementaron utilizando la biblioteca Python PyTorch4 (v1.4.0). Los experimentos se realizaron en una máquina con las siguientes estadísticas: CPU Intel Core i9-9900K (-MT-MCP-) de 8 núcleos, velocidad/máx.: 3651/5000 MHz, caché: 16384 KB y RAM: 64 GB.

3. conjunto

Un sistema conjunto está formado por varios modelos de bajo sesgo cuyas predicciones se combinan para producir una predicción final más sólida. Se deben aplicar múltiples consideraciones al ensamblar modelos de extracción de información. Por ejemplo, el sistema debe decidir qué anotaciones están siendo votadas por un conjunto común de modelos y cuáles no. En caso de anotaciones contradictorias , se debe seleccionar la(s) predominante(s). Posteriormente, el sistema también debe decidir qué anotaciones votadas son lo suficientemente buenas para ser incluidas. Todas estas decisiones tienen un papel importante en la producción del resultado final.

De acuerdo con esto, definimos un conjunto de reglas artesanales que se canalizaron para construir un sistema de ensamblaje de la siguiente manera:

1. Dada la colección de anotaciones por modelo, cree el conjunto de unión de todas las anotaciones mientras realiza un seguimiento de qué modelos votaron por cada anotación, es decir, el modelo incluyó la anotación.

2https://spacey.io/ 3https://github.com/google-research/bert (bert-base-multilingüe-casado) 4https://pytorch.org/

2. Ponderar los votos otorgados por cada modelo a cada anotación.
3. Calcular una puntuación agregada de la calidad de cada anotación según sus votos.
4. Decidir si conservar la anotación o no y, en caso de anotaciones contradictorias, decidir cuáles conservar.

Un sistema de votación clásico podría asignar un peso uniforme a cada modelo de votación. Sólo se reportan las anotaciones que superan un conteo predefinido y en caso de conflicto, sólo aquellas que logran la mayoría de votos. Se puede implementar un sistema experto calculando previamente la importancia por tipo de anotación de cada modelo y utilizándolo para sopesar los votos. Los votos emitidos por el mejor modelo correspondiente se ponderan en consecuencia y los demás se valoran como cero.
Sólo se reportan las anotaciones con voto distinto de cero y en caso de conflicto, sólo aquellas que lograron la mejor puntuación.

Además de las reglas elaboradas a mano, se puede entrenar un conjunto de algoritmos de aprendizaje automático para reunir todos los votos del sistema. El modelo de aprendizaje automático predice la probabilidad de que una anotación etiquetada sea correcta, es decir, la probabilidad de que pertenezca a una colección anotada en oro, según los votos que le dio cada sistema. Aquí se pueden configurar dos aspectos clave: primero, cómo se transforma cada anotación en un vector de características y qué modelo la manejará, y segundo, qué arquitectura de modelo concreta se utilizará.

En el contexto del desafío, se realizó una búsqueda evolutiva probabilística para explorar el espacio de configuración del conjunto. La búsqueda comienza con una estrategia de muestreo aleatorio, pero a medida que evalúa más tuberías, modifica un modelo de muestreo probabilístico para que las tuberías similares a las mejores encontradas se muestreen con mayor frecuencia. La función de aptitud está configurada para maximizar directamente la puntuación 1 del conjunto resultante de acuerdo con la colección de referencia. La colección de referencia se configuró en el escenario 1 de la colección de entrenamiento.

Las configuraciones exploradas no se cubrirán en este documento. Se ensamblaron los cuatro modelos que se muestran en la Tabla 2 . El proceso de ensamblaje mejor encontrado para los escenarios 2 y 3 fue utilizar un clasificador binario de máquina de vectores de soporte (SVM) diferente para cada tipo diferente de entidad y relación. Solo se ensamblaron dos modelos (cnet y only-bert) para los escenarios 1 y 4 debido a que no se tenían las presentaciones correspondientes de los modelos restantes (deptree y cnet-deptree) en estos escenarios (debido a limitaciones de tiempo). La canalización de ensamblaje mejor encontrada fue utilizar un único clasificador de regresión logística para todos los tipos de entidades y relaciones.

4. Resultados

Se presentaron al desafío dos de los cuatro modelos entrenados: los modelos cnet y deptree . Además, se aplicó el método de conjunto descrito en la Sección 3 para producir una tercera ejecución. La Tabla 3 muestra los resultados obtenidos por las tres carreras del desafío. Debido a problemas de tiempo, algunos escenarios no se enviaron para algunas ejecuciones y en su lugar se utilizó la implementación de referencia (resaltada con una en la tabla). También hubo un problema con el método de conjunto en el escenario 3 (Tarea B), lo que resultó en una presentación mal formada. La ejecución conjunta obtuvo los mejores resultados de las tres ejecuciones, excepto por el escenario enviado incorrectamente.

La Tabla 4 resume los resultados generales obtenidos por el sistema del equipo en el desafío. Ocupó el quinto lugar en todos los escenarios de evaluación, empatando en el cuarto lugar en el escenario de evaluación principal. Los resultados más prometedores del sistema se lograron en el escenario 2 (Tarea A).

Tabla 3

Resumen del rendimiento (1) logrado por el sistema en las tres ejecuciones. La mejor puntuación conseguida.
en cada escenario del desafío se incluye con fines comparativos. Además, se incluye la línea base de implementación proporcionada
por los„organizadores del desafío. Las ejecuciones que utilizan la solución de referencia están marcadas
con ".

Sistema	1-principal	2 tareasA	3 tareasB	4-transferencia
alto rendimiento	.	.	.	.
ejecución 1: cnet	0.552	0,792	0.306	0.367
ejecución 2: deptree	0,395	.	.	0.138
ejecución 3: conjunto	.	.	0.005	.
base	0,395	0.542	0,131	0.138

Tabla 4

Resumen del desempeño (1) alcanzado por cada participante en el desafío. Los equipos están ordenados.
según su desempeño en el escenario 1.

Equipo	1-principal	2 tareasA	3 tareasB	4-transferencia
vicomtec	.	0.821	0.583	0.563
Talp-UPC	0,627	0.816	0.575	.
UH-MAJA-KD	0,625	0.814	0,599	0.548
IXA-DOWN-RE	0.557	0,692	.	0,479
UH-MatCom	0.557	0,795	0.545	0.373
SINAÍ	0,421	.	0,462	0.281
Línea de	0,395	0,542	0,316	0.138
base HAPLAP	0,395	0,542	0,131	0.138
exsim	0.246	0.314	0,131	0,122

5. Discusión

Se entrenaron y probaron varios modelos en la colección de desarrollo, aunque no eran
se ejecuta en la colección de prueba y, por lo tanto, no se utiliza en la ejecución del conjunto. Usando BERT para incrustar
las palabras causaron la mayor mejora en el rendimiento en la Tarea A. De hecho, simplemente usando
la representación BERT y una capa densa lineal, se logran resultados competitivos en el
colección de desarrollo. Esto no es suficiente para la Tarea B, en la que sólo se utilizan los elementos concatenados.
La representación BERT de los tokens fuente y principal y una capa lineal densa dan como resultado muy bajos
actuación.

Otra gran mejora en el rendimiento del sistema se observó al utilizar la clase
ponderación para entrenar el clasificador por pares. Penalizar a las clases más representadas (la falta de
relación en este contexto) ayudó a evitar modelos mal entrenados que maximizan la precisión del entrenamiento
pero sin relacionarse con la métrica. Es necesario explorar procedimientos de formación alternativos, como
como resolver un problema de clasificación de etiquetas múltiples con selección de umbral para evitar la inclusión

de la falta de relación de clase. Optimizando directamente la $_1$ métrica durante el proceso de formación es otra alternativa a explorar.

La incorporación de información de ConceptNet no tuvo el impacto positivo que esperábamos. Esto podría haber sido causado por limitaciones en la representación (vector cero-uno) y la forma en que se introdujo en el modelo. Sin embargo, pensamos que el uso de este tipo de información puede resultar realmente valioso en tareas como ésta, en las que hay pocos ejemplos de referencia disponibles. Es necesario realizar trabajos futuros a este respecto.

6. Conclusiones

Este artículo describe el sistema presentado por el equipo UH-MatCom en el desafío eHealth-KD en IberLEF 2020. Se entrenaron y ensamblaron varios modelos de aprendizaje profundo para extraer automáticamente entidades y relaciones relevantes de documentos de texto plano. El sistema logró resultados promedio en el desafío, ocupando el quinto lugar en todos los diferentes escenarios de evaluación. Debido a problemas de recursos, algunas estrategias prometedoras no se probaron completamente. Se realizarán trabajos futuros para resolver esas limitaciones.

Se utilizó información tanto sintáctica como semántica. El uso de bases de conocimiento externas para iniciar la tarea de extracción de relaciones y el escenario de transferencia de aprendizaje no funcionó como se esperaba. Es necesario trabajar más en este sentido, para mejorar el rendimiento en futuras ediciones del desafío.

Expresiones de gratitud

Funding: This research has been partially funded by the University of Alicante and the Univer-sity of Havana, the Generalitat Valenciana (Conselleria d'Educació, Investigació, Cultura i Esport) and the Spanish Government through the projects LIVING-LANG (RTI2018-094653-B-C22) and SIIA (PROMETEO/2018/089, PROMETEU/2018/089).

Referencias

[1] Piad-Morffis, Y. Gutierrez, H. Cañizares-Diaz, S. Estevez-Velarde, Y. Almeida-Cruz, R. Muñoz y A. Montoyo, Overview of the eHealth Knowledge Discovery Challenge en IberLEF 2020, en : Actas del Foro Ibérico de Evaluación de Lenguas compartido con la 36ª Conferencia de la Sociedad Española para el Procesamiento del Lenguaje Natural, IberLEF@SEPLN 2020,

[2] J. Devlin, M.-W. Chang, K. Lee, K. Toutanova, Bert: Entrenamiento previo de transformadores bidireccionales profundos para la comprensión del lenguaje, preimpresión de arXiv arXiv:1810.04805 (2018).

[3] J. Lafferty, A. McCallum, FC Pereira, Campos aleatorios condicionales: modelos probabilísticos para segmentación y etiquetado de datos de secuencia (2001).

[4] A. Graves, J. Schmidhuber, Clasificación de fonemas framewise con lstm bidireccional y otras arquitecturas de redes neuronales, Neural Networks 18 (2005) 602–610.

[5] R. Speer, J. Chin, C. Havasi, Conceptnet 5.5: Un gráfico multilingüe abierto de conocimiento general , 2017. URL: http://aaai.org/ocs/index.php/AAAI/AAAI17/ documento/vista/14972.

[6] L. Ratinov, D. Roth, Desafíos de diseño y conceptos erróneos en el reconocimiento de entidades nombradas, en: Actas de la Decimotercera Conferencia sobre Aprendizaje Computacional de Lenguajes Naturales (CoNLL-2009), 2009, págs.
[7] JM Alvarado, EQ Caballero, A. Rodríguez, Uh-maja-kd en el desafío ehealth-kd (2019).
[8] Y. Wu, M. Schuster, Z. Chen, QV Le, M. Norouzi, W. Macherey, M. Krikun, Y. Cao, Q. Gao, K. Macherey, et al., traducción automática neuronal de Google sistema: Cerrando la brecha entre la traducción humana y automática, preimpresión de arXiv arXiv:1609.08144 (2016).