Detección de Intenciones y Reconocimiento de Entidades con una RNN de arquitectura Seq2Seq y mecanismo de Atención

Daniel Arteaga
Escuela de Posgrado - Informática
Pontificia Universidad Católica del Perú
Lima, Perú
darteagam@pucp.edu.pe

Juan Tovar
Escuela de Posgrado - Informática
Pontificia Universidad Católica del Perú
Lima, Perú
juan.tovarg@pucp.edu.pe

Ruben Cordova
Escuela de Posgrado - Informática
Pontificia Universidad Católica del Perú
Lima, Perú
ruben.cordova@pucp.edu.pe

***Abstract*—Actualmente aplicaciones como chatbots permiten a las organizaciones interacturar de forma automatizada con sus usuarios. Estas interacciones, al ser de interés pues se encuentran dentro del dominio de Entendimiento de Lenguaje Hablado, han sido modelados usando diferentes arquitecturas. En el presente trabajo se explora un modelo de Red Neuronal Recurrente con arquitectura secuencia a secuencia y mecanismo de atención, enfocado en la detección conjunta de intenciones y reconocimiento de entidades. En este trabajo se realiza la comparación con modelos del estado del arte en este campo y se compara el efecto del padding al inicio y al final de la secuencia. También se plantea la aplicación del modelo con un conjunto de datos propio para el dominio de un chatbot.**

***Index Terms*—seq2seq, attention, intent detection, slot filling, padding**

I. Introducción

La detección de intenciones y reconocimiento de entidades, son elementos importantes en soluciones como chatbots, que generan un valor agregado en las organizaciones para la satisfacción de los usuarios. Es por ello, que el objetivo de este estudio es la creación de un modelo de Deep Learning con arquitecura secuencia a secuencia (*sequence-to-sequence*) para la detección de intenciones y reconocimiento de entidades, automatizando tareas y ejecutando acciones presentes en un diálogo. Sequence-to-sequence (seq2seq) es una arquitectura empleada ampliamente en diversos problemas de Entendimiento de Lenguaje Natural como traducción de máquina. Específicamente, para la etapa de modelado de la secuencia se emplea el paradigma de codificador - decodificador, donde el codificador lee la entrada y produce una representación del contexto para luego generar la secuencia objetivo. Adicionalmente, se incluye el concepto de atención, el cual le permite al modelo centrarse en partes específicas de la entrada en los diferentes pasos de la red neuronal. Con esto, el modelo obtiene la capacidad de centrarse solamente en las partes relevantes de la entrada.

El objetivo de este trabajo es experimentar con técnicas de Deep Learning para la detección de intenciones y reconocimiento de entidades de forma simultánea, realizando comparaciones con diferentes modelos, conjuntos de datos y técnicas de padding.

El trabajo está organizado de la siguiente manera: en la sección II se presenta el estado de arte, donde se analizan las técnicas utilizadas en trabajos similares. En la sección III se detalla la metodología empleada; en la sección IV se presenta la experimentación y resultados, específicamente la descripción del conjunto de datos, el entorno de experimentación y los resultados obtenidos. En la sección V se discute los resultados obtenidos y en la sección VI se presentan la conclusiones y trabajos futuros.

II. Estado del Arte

Estudios proponen el uso de técnicas de deep learning, como Redes Neuronales Recurrentes (RNN) basados en la atención, para la detección de intenciones y reconocimiento de entidades, pasos importantes para la comprensión del habla y sistemas de diálogo. En [1] Mesnil et al. proponen el uso de RNNs para realizar reconocimiento de entidades en escenarios de compresión de lenguaje hablado (SLU). Este trabajo es uno de las primeras propuestas en este tipo de redes neuronales, cuyos resultados obtenidos muestran una mejora respecto a las técnicas tradicionalmente empleadas como Conditional Random Field (CRF), específicamente en la reducción del 2% en el error absoluto. Con el paso de los años, la atención de las investigaciones se orientó a desarrollar modelos que puedan realizar de forma conjunta la detección de intención y reconocimiento de entidades. En [2] Bing utiliza ATIS (Airline Travel Information Systems) data set, ampliamente utilizado en la investigación de la comprensión del lenguaje hablado. Donde utiliza un modelo de red neuronal recurrente bidireccional con buenos resultados, un F1 score del 95.78 y un error del 5.6% para el reconocimiento de entidades y detección de intenciones, respectivamente. En [3] Schumann, además de utilizar un modelo recurrente basado en la atención, se incorpora la corrección del error del reconocimiento automático del habla (ASR) y con el uso del ATIS dataset y agregando hipótesis ASR (creadas utilizando "Google Text to Speech" API), se logra unos resultados de F1

score del 87.13 y un error del 5.04% para el reconocimiento de entidades y detección de intenciones, respectivamente. En [4] Wang, se propone un modelo híbrido de red neuronal convolucional y red de memoria a largo y corto plazo (CNN-BLSTM) para la codificación y una red neuronal recurrente basada en la atención para la decodificación, con el uso del ATIS dataset se logra unos resultados de F1 score del 97.76% y una precisión del 97.17% para el reconocimiento de entidades y detección de intenciones, respectivamente. Por ejemplo, en En [5] Xu y Sarikaya proponen el uso de Redes Neuronales Convolucionales basadas en CRF Triangular (TriCRF) para la detección conjunta. Con esta técnica, los autores logran obtener 5.91% en el valor de error de intención (reducción de 1%) y 95.42% en la métrica de F1 (incremento de 1%), respecto a otros métodos. Adicionalmente, en el presente trabajo no solo se emplea el dataset ATIS sino también en otros dominios (comunicaciones, calendarios, alarmas y notas), donde el modelo obtuvo valores de error de intención entre 5.5% y 7.1% y de la métrica F1 entre 86% y 90%. Otro ejemplo es [6], donde Guo et al. proponen el uso de Redes Neuronales Recursivas (RecNN) para la detección conjunta. En el trabajo se evalúan los resultados del modelo en comparación con otros modelos (p. ej. TriCRF), así como el impacto en utilizar el algoritmo de Viterbi para la optimización del modelo. Respecto a otros modelos, se obtienen valores de exactitud de 95.4% para detección de intención y 93.22% para reconocimiento de entidades. Al incluir el algoritmo de Viterbi, se mejora en 0.4% la exactitud del reconocimiento de entidades. Cabe resaltar que además de emplear el dataset de ATIS, también se emplea la data de Cortana (asistente personal de Microsoft) en los dominios de calendario, comunicaciones y clima. Los resultados obtenidos cuando se usaba la data de los 3 dominios en conjunto muestran una exactitud de 92.28% y 87.86% para detección de intención y reconocimiento de entidades, respectivamente. En [7] se utiliza el mecanismo de atención tanto para las entidades y las intenciones. Además se presenta dos subredes para las intenciones y entidades, para las entidades se adiciona una capa de campo aleatorio condicional para el etiquetado de las secuencia con resultados de F1 score del 95.80% y una precisión del 97.76% para el reconocimiento de entidades y detección de intenciones, respectivamente. Adicionalmente, en [8] se presenta un modelo basado en la arquitectura BERT (Bidirectional Encoder Representations from Transformers) para la detección conjunta. Se observa que al evaluar el modelo con el dataset de ATIS, se obtiene una exactitud de 97.5% en detección de intención y un valor de 96.1% en la métrica F1 para reconocimeinto de entidades.

III. Metodología

En este trabajo se explora el uso de un modelo Secuencia a Secuencia ("seq2seq") [9], también conocido como Codificador - Decodificador (*encoder-decoder*), con mecanismo de Atención [10] para detectar intenciones y reconocer intenciones simultáneamente. El modelo propuesto es una adaptación del modelo Codificador - Decodificador con Alineamiento y Atención descrito en [2]. Como parte del estudio se propone realizar los siguientes experimentos:

- Comparar el rendimiento del modelo propuesto con otros modelos sobre el conjunto de datos ATIS [11]
- Comparar la influencia del padding al inicio y al final de la secuencia
- Probar el modelo propuesto con un conjunto de datos manualmente etiquetados en el dominio de consultas sobre cursos de capacitación.

En esta sección se describe el preprocesamiento de datos, las métricas de evaluación, el modelo propuesto, el procedimiento de ajuste de hiperparámetros y el marco de la metodología.

A. Preprocesamiento de datos

En el caso del conjunto de datos ATIS [11], los textos ya se encuentran con el formato de anotación para entidades IOB2. El primer paso del preprocesamiento para este conjunto de datos consiste en adecuar los textos al formato de padding. Cuando se aplica un padding al inicio (*pre-padding*), se mantiene (o agrega) el token de inicio de oración "BOS" y se prescinde del token de fin de oración "EOS". Si se aplica un padding al final (*post-padding*) es necesario tanto el token "BOS" como el token "EOS".

El siguiente paso en este preprocesamiento es realizar la tokenización. Para la tokenización de palabras y entidades es necesario agregar un token "UNK" para las palabras desconocidas (las que no aparecen en el vocabulario de entrenamiento). También es importante considerar que el token "PAD" para el padding esté en el diccionario en la primera posición. La tokenización se realizó a través del Tokenizer de Tensorflow.

Una vez que se tienen los tokenizadores para las palabras, entidades e intenciones se realiza la conversión de texto a secuencias de los índices que representan estas palabras.

Como paso final del preprocesamiento se tiene que desfazar las secuencias de entidades, pues esta es usada tanto para entrada como salida del modelo. El desface debe ser únicamente de un índice.

Para el caso del conjunto de datos propio es necesario hacer una selección de intenciones más frecuentes, de forma que el conjunto de datos esté más equilibrado. Las intenciones a utilizar son: "informacion_curso", "informacion_usuario", "saludo", "despedida" y "gracias". Posteriormente se debe de modificar los textos para dar el formato de anotación IOB2. Esto se realiza con el uso de expresiones regulares. Una vez que el texto posee este formato se puede proceder de manera similar que para el conjunto de entranamiento ATIS [11].

B. Métricas de Evaluación

Tanto como para el caso de la detección de intenciones como para el reconocimiento de entidades, los elementos a predecir (entidades e intenciones) presentan una frecuencia heterogénea por lo que la mejor métrica es el F1-score. Adicionalmente se ha encontrado en el estado del arte que la métrica de Exactitud también se emplea para la detección de las intenciones.

C. Modelo Propuesto

El modelo propuesto consta de un codificador basado en una sola capa LSTM bidireccional y decodificadores basados en capas LSTM unidireccionales. Así mismo, se ha considerado un alineamiento de las salidas del codificador (como secuencia, es decir salidas por cada palabra de la secuencia de entrada) a las entradas respectivas del decodificador. Adicionalmente, se ha empleado el mecanismo de Atención de Bahdanau [12]. En la Fig. 1 se muestra un diagrama de la arquitectura seguida para el modelo propuesto.

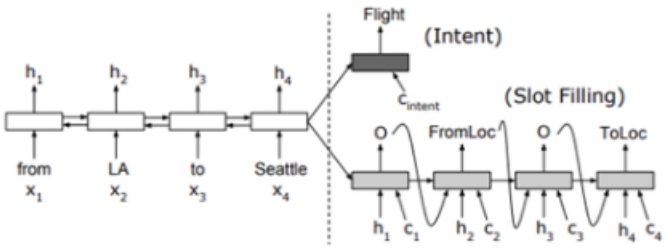


Fig. 1: Arquitectura seguida para el modelo propuesto

La implementación de este modelo se basa en [2]. Las entradas del modelo (secuencia de palabras y secuencia de entidades) ingresan a una capa de embeddings previo a su paso por las capas LSTM. Los embeddings han sido inicializados con los pesos calculados a través de Word2Vec.

Adicionalmente, esta implementación aplica al estado de memoria del codificador una transformación similar que al estado oculto, pero otros pesos aprendidos.

D. Ajuste de Hiperparámetros

El ajuste de los hiperparámetros del modelo se ha realizado mediante un tunning manual basado en la evolución de los resultados. Como se indica al inicio de esta sección, se han realizado diferentes experimentos para los cuales se han ajustando los hiperparámetros. El mejor modelo obtenido en los primeros experimentos ha servido como punto de partida para continuar la experimentación siguiente. Los hiperparámetros principales son: el tamaño de los embeddings, la dimensión latente de las capas LSTM, el tamaño del batch y el learning rate.

E. Marco de la Metodología

Este trabajo se desarrolló en el marco que se muestra en la Fig. 2, que implica seis fases claves [13].

- **Comprensión del negocio**: La automatización de tareas, la toma de decisiones y ejecución de acciones mediante diálogos, son factores críticos en las organizaciones; por ello las soluciones como los chatbots agregan valor a una organización, dado que permiten mejorar la atención al usuario y, por tanto, satisfacer su necesidad
- **Comprensión de los datos**: Recolección inicial de los datos, identificación de problemas de calidad y conocimiento preliminar sobre los datos, con la finalidad de formular hipótesis
- **Preparación de los datos**: Preprocesamiento de los datos para construir el conjunto final de datoso preliminar sobre los datos, con la finalidad de formular hipótesis
- **Modelado**: En esta fase, se seleccionan y aplican las técnicas de modelado, y se calibran los parámetros para obtener métricas de validación óptimas
- **Evaluación**: Con los modelos construidos se comparan los resultados con los objetivos de negocio
- **Despliegue**: Creación del modelo final y puesta en producción para que el usuario pueda usarlo.

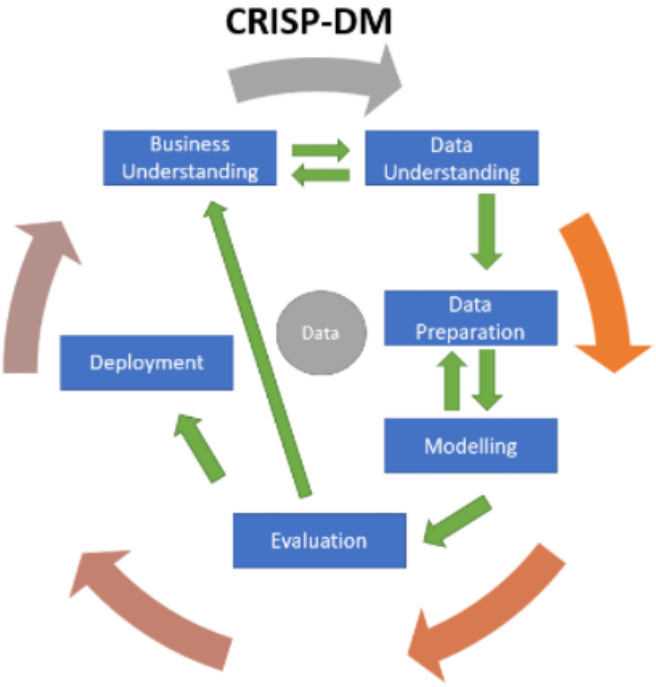


Fig. 2: Metodología del desarrollo del modelo a utilizar

IV. Experimentación y Resultados

A. Descripción del conjunto de datos

*1) **ATIS**:* El conjunto de datos ATIS (Airline Travel Information System) [11] es una referencia ampliamente utilizada para el entrenamiento y prueba de modelos con la arquitectura Secuencia a Secuencia. Este conjunto de datos se ha conformado a partir de grabaciones de voz de personas haciendo reservas de vuelo. El conjunto de datos utilizado se conforma de 4978 oraciones de entrenamiento y 893 oraciones de prueba. Cada oración (secuencia de entrada) posee una intención, que se puede entender como una clase, y una secuencia de etiquetas correspondientes a las entidades reconocidas (secuencia de salida). Este conjunto de datos tiene 943 palabras, 129 entidades y 26 intenciones. ATIS ha sido conformado a partir de textos en inglés. En la Fig. 3 se presenta un ejemplo de ATIS.

*2) **Conjunto de Datos Propio**:* Se viene conformando un conjunto de datos manualmente etiquetados a partir de textos de correos electrónicos enviados al área de informes del Instituto Nacional de Investigación y Capacitación de

Sentence	first	class	fares	from	boston	to	denver
Slots	B-class_type	I-class_type	O	O	B-fromloc	O	B-toloc
Intent	airfare						

Fig. 3: Ejemplo del Conjunto de Datos ATIS

Telecomunicaciones (INICTEL-UNI), donde se consulta por información de los cursos de capacitación impartidos. Se han etiquetado hasta la fecha un total de 1200 correos electrónicos, lo que representa 6306 oraciones. Para cada oración se ha obtenido una intención y una secuencia de etiquetas de las entidades del dominio. Este conjunto de datos posee 4806 palabras, 36 entidades y 21 intenciones. Sin embargo, debido a que los números de ejemplos por cada intención se encuentran desbalanceados, se aplicará una selección de 7 intenciones para probar el modelo propuesto. En la Fig. 4 se muestra un ejemplo de este conjunto de datos.

Oraciones	Secuencia de Entidades	Intenciones
Informacion del curso de SISTEMAS DE video vigilancia CCTV porfa	<o> <o> <o> <o> B-<nombre_curso> I-<nombre_curso> I-<nombre_curso> I-<nombre_curso> I-<nombre_curso> <o>	informacion_curso

Fig. 4: Ejemplo del Conjunto de Datos Propio

B. Descripción del entorno de experimentación

El modelo propuesto se desarrolló en Colaboratory (Colab), servicio en línea ofrecido por Google basado en los Notebooks de Jupyter. Esta solución está orientada a estudiantes e investigadores para que desarrollen sus experimentos de inteligencia artificial o ciencia de datos en Python y de forma colaborativa, sin tener que realizar alguna configuración de la infraestructura. También se emplearon la framework Pytorch para los modelos comparativos, así como la librería TensorFlow y el API Keras para el modelo final propuesto. Cabe la pena resaltar que para el entrenamiento del modelo propuesto, se utilizó en Colab la opción de aceleración por hardware mediante GPUs.

C. Reproducción de Resultados

Para poder tener modelos comparativos se reprodujo los resultados de otros modelos en el estado del arte. Primero se reprodujeron los resultados obtenidos del modelo base [2]. Se utilizó una implementación en Pytorch, que a diferencia del modelo propuesto, posee solamente como entrada a la secuencia de palabras. Los hiperparámetros utilizados son los mismos que se proponen en el modelo base. Debido a que el modelo propuesto se basa en el mismo trabajo, y a que esta primera implementación no muestra las métricas de F1 para entidades y exactitud para intenciones, no se usó esta implementación en la comparación final.

Posteriormente se planteó emplear modelos más recientes en el estado del arte. Así se reprodujeron los resultados de un modelo basado en redes atencionales como BERT para la tarea de detección de intenciones y reconocimiento de entidades [8]. Esta implementación también se encontraba en Pytorch. Para poder comparar los resultados se empleó un tamaño del batch de 32 ejemplos, el tamaño máximo de la secuencia se estableció en 50, se utilizó postpadding y se entrenó con el mismo conjunto de datos ATIS (con los mismos ejemplos en el entrenamiento, validación y prueba).

Otro modelo que se utilizó está basado en [7], el cuál es una mejora del modelo base ya que incorpora dos subredes para la detección de intenciones y el reconocimiento de entidades. Además, para este último añade una capa de Campo Aleatorio Condicional (*Conditional Random Field - CRF*). Al igual que con la implementación de BERT, se empleó 32 ejemplos por batch, un tamaño máximo de secuencia de 50, postpadding y el mismo conjunto de datos ATIS.

D. Entrenamiento del Modelo Propuesto.

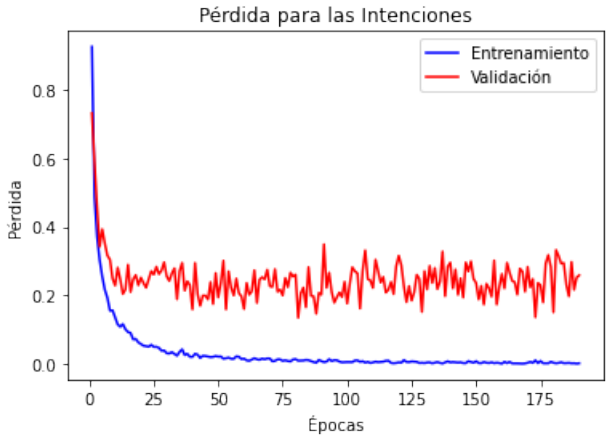
El modelo propuesto ha sido entrenado con diversas configuraciones de hiperparámetros. Se empezó usando un tamaño de embedding de palabras de 64, una dimensión latente de las capas LSTM de 64, 16 ejemplos por batch y un learning rate fijo de 0.001. Siguiendo las recomendaciones del modelo base se entrenó con el optimizador de Adam y un recorte de la gradiente (*gradient clipping*) para evitar que esta crezca exageradamente (*exploding gradients*).

Se decidió iniciar con dichos hiperparámetros para realizar pruebas rápidas de rendimiento empleando 30 épocas como máximo. Así se fueron ajustando los valores decidiendo aumentar a 128 el tamaño del embedding de palabras y la dimensión latente de las capas LSTM. Como es de esperarse, este aumento significó un mayor tiempo de entrenamiento. Cuando se realizó una prueba para un tamaño de embedding de 256 y una dimensión latente de 250, la plataforma Colab se desconectó antes de terminar el entrenamiento debido al incremento en el tiempo. Por esta razón, la configuración de hipermarámetros que se utilizó para los resultados finales con el conjunto de datos ATIS fue la siguiente:

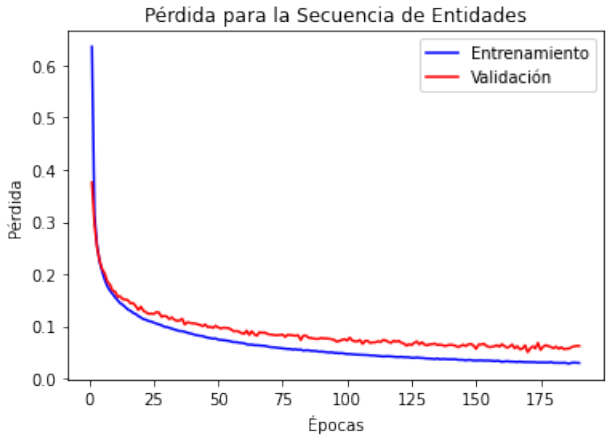
- Tamaño máximo de la secuencia: 50
- Tamaño del embedding de palabras: 128
- Tamaño del embedding de entidades: el valor entero más cercano y menor a la tercera parte del total de entidades
- Dimensión latente de las capas LSTM: 128
- Tamaño del batch: 32 (para entrenamiento) y 64 (para validación)
- Learning rate: se empleó un decaimiento exponencial cada 5 épocas con un ratio de 0.95 y un valor inicial de 0.001

Para el caso del conjunto de datos propio, las diferencias en la configuración de hiperparámetros con respecto a la anterior fue: un tamaño máximo de la secuencia de 150, y un tamaño del embedding de palabras de 250.

Para todos los entrenamientos se empleó como función de pérdida a la Entropía Cruzada Categórica Sparce(*Sparce Categorical Cross-Entropy*), como métricas para la detección de intenciones se empleó la Exactitud Categórica Sparce (*Sparce Categorical Accuracy*) y el F1-Score; y como métrica para el reconocimiento de entidades se usó el F1-Score.

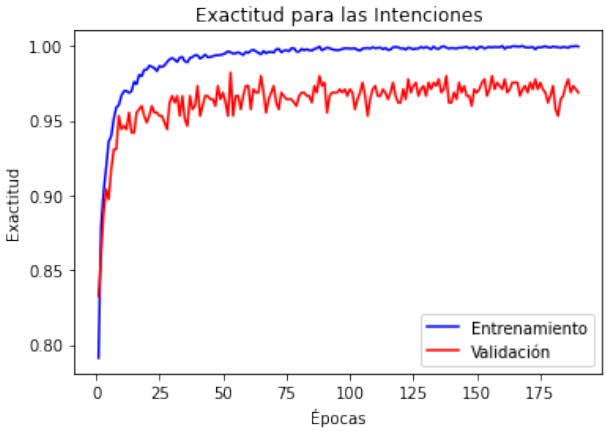


(a) Pérdida para la Detección de Intenciones

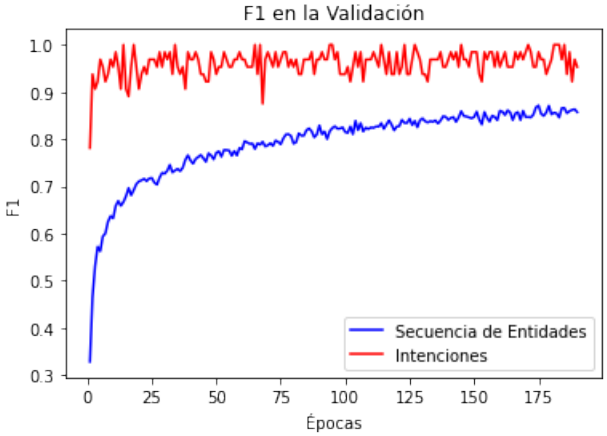


(b) Pérdida para el Reconocimiento de Entidades

Fig. 5: Curvas de Aprendizaje en Entrenamiento y Validación



(a) Accuracy para la Detección de Intenciones



(b) F1-Score para el Reconocimiento de Entidades y Detección de Intenciones en la Validación

Fig. 6: Curvas de Rendimiento sobre ATIS

Para el conjunto de datos ATIS, se estableció 250 épocas con interrupción temprana (*early stopping*) tras 15 épocas sin que aumente el valor del F1-Score para el reconocimiento de entidades. Con este conjunto de datos se realizó la comparación de emplear padding al inicio y padding al final de la secuencia.

En la Fig. 5 se muestran las curvas de aprendizaje para el entrenamiento y validación del modelo propuesto con el conjunto de datos ATIS con padding al final de la secuencia. El número de épocas que finalmente se ejecutaron fue de 190 debido al *early stopping*.

En la Fig. 6 se muestran las curvas de rendimiento del modelo para la detección de intenciones (en el entrenamiento y validación) y para el reconocimiento de entidades (en la validación) del modelo propuesto con el conjunto de datos ATIS con padding al final de la secuencia.

Para el conjunto de datos propio, se estableció 100 épocas con la misma configuración de la interrupción temprana (*early stopping*). Para este conjunto de datos únicamente se empleó el padding al final de la secuencia debido a que se obtuvo mejores resultados.

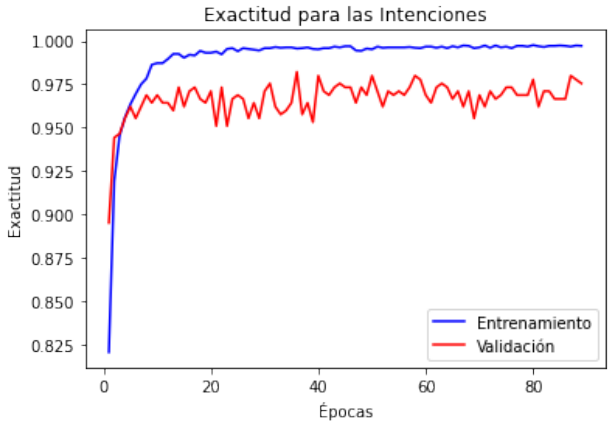
En la Fig. 7 se muestran las curvas de rendimiento del modelo para la detección de intenciones (en el entrenamiento y validación) y para el reconocimiento de entidades (en la validación) del modelo propuesto con el conjunto de datos propio con padding al final de la secuencia.

V. Discusión

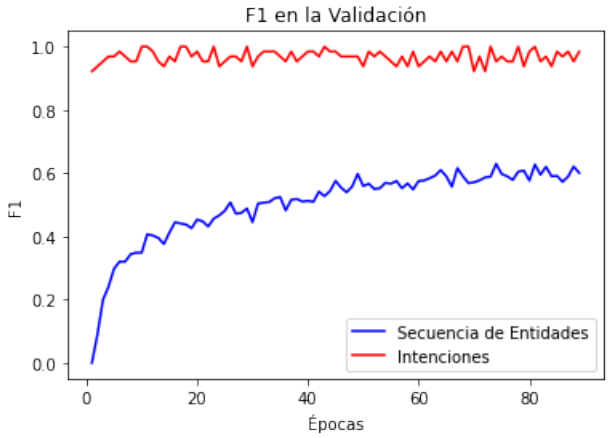
A. Comparación de Modelos

En este trabajo se ha realizado la comparación del modelo propuesto con los modelos reproducidos comentados en la sección anterior. La comparación se ha realizado empleando el conjunto de datos ATIS con padding al final de la secuencia. Para poder hacer una comparación justa se ha planteado utilizar 32 ejemplos por batch en el entrenamiento, 64 ejemplos por batch en la validación y un tamaño máximo de secuencia igual a 50.

En la Tabla I se muestran los resultados obtenidos en la ejecución de los modelos del estado del arte y el modelo propuesto. Se observa que los modelos del estado del arte superan al modelo propuesto para la tarea de reconocimiento de entidades (F1-score); sin embargo en la tarea de detección de intenciones (Exactitud) poseen un rendimiento similar. Las



(a) Accuracy para la Detección de Intenciones



(b) F1-Score para el Reconocimiento de Entidades y Detección de Intenciones en la Validación

Fig. 7: Curvas de Rendimiento sobre el Conjunto de Datos Propio

métricas de comparación han sido calculadas con el conjunto de validación.

Otra comparación que se ha realizado en este trabajo es sobre el efecto del padding al inicio y al final de la secuencia. Esta comparación se desarrolló para el modelo propuesto únicamente. En la Tabla II, se evidencia que el entrenamiento con padding al incio de la secuencia es menos eficiente que con el padding al final de la secuencia.

Haciendo uso del padding al inicio de la secuencia se obtiene un valor de F1-score para el reconocimiento de entidades ligeramente menor que cuando se emplea padding al final de la secuencia. Además de ello, el tiempo de entrenamiento con padding al inicio de la secuencia es casi una hora mayor que el otro enfoque; a pesar de que se realizan 19 iteraciones menos.

TABLE I: Comparación de modelos

Modelo	F1-Score (Entidades)	Exactitud (Intenciones)
BERT	96.10	97.5
SF-ID Network	95.80	97.76
Modelo Propuesto	87.13	97.54

TABLE II: Comparación de prepadding y postpadding

Modelo	F1-Score	Exactitud	Épocas	Tiempo
Postpadding	87.13	97.54	190	4h 25min
Prepadding	85.95	97.54	171	5h 17min

B. Interpretación de los resultados

De la comparación entre los modelos del estado del arte y el modelo propuesto se observa que la tarea de reconocer exactamente las entidades es una tarea más compleja que la tarea de identificar intenciones y los modelos más actuales presentan mejores resultados. Esto se debe a la complejidad extra de los modelos más actuales, como por ejemplo estar basado en redes atencionales y embeddings contextualizados. Esta complejidad permite a los modelos más recientes la capacidad de poder prestar atención, y así incorporar contexto, en cada posición de la secuencia de entrada; además de poseer representaciones de los tokens de acuerdo al contexto.

Por otro lado, de la comparación del padding al inicio y al final de la secuencia, se interpreta que el padding al final de la secuencia termina siendo más eficiente debido a que si bien es cierto el codificador bidireccional procesa la secuencia en ambas direcciones (resultado similar el efecto padding al final y al inicio), el decodificador (que es unidireccional) procesa la secuencia iniciando por los tokens de las entidades construyendo un contexto que debe recordar hasta que llega a los tokens de padding. Cuando se aplica padding al inicio de la secuencia, el decodificador procesa primero los tokens de padding y el contexto que recibe del decodificador se pierde hasta que procesa los tokens de las entidades.

Finalmente, de la comparación de aplicar el modelo propuesto con el conjunto de datos ATIS y el conjunto de datos propio, se puede inferir que el hecho de haber empleado una mayor cantidad (el triple) de tokens en las secuencias es una de las razones de la disminución del rendimiento del modelo debido a que es más difícil mantener un contexto con una secuencia de mayor longitud.

C. Limitaciones experimentadas con el Modelo Propuesto

La principal limitación que se ha evidenciado con el modelo propuesto se presenta cuando se procesan secuencias de texto largas. A pesar de que la compresión del contexto en un vector resulta beneficioso para la arquitectura secuencia a secuencia, secuencias largas resultan ser difíciles de representar en un vector de contexto de tamaño fijo. Esto se refleja en el rendimiento del modelo cuando se procesó textos del conjunto de datos propio con una secuencia de 150 palabras, alcanzando un F1-score al rededor de 0.6; mientras que para el conjunto de datos ATIS con un tamaño de secuencia máximo de 50, el F1-score se aproxima a 0.9.

D. Mejoras futuras del sistema

El rendimiento del modelo propuesto para su uso en un sistema como un chatbot puede mejorarse aumentando el tamaño del embedding de palabras y la dimensión latente de las capas LSTM. Este experimento ha sido realizado por uno

de los autores del trabajo en la aplicación de este modelo en un dominio similar para clasificar textos y detectar efectivame los nombres, apellidos, instituciones, cargos y fechas de personas expuestas políticamente a partir de resoluciones del Diario Oficial El Peruano.

Otra mejora importante a tener en cuenta es el uso de un conjunto de datos de calidad y con una adecuada cantidad de ejemplos. De hecho, uno de los problemas en Entendimiento del Lenguaje Natural es el de no contar con una cantidad considerable de datos de entrada. Para poder lidiar con ello, se puede emplear la técnica de Data Augmentation mediante el uso de Autoencoders (en particular los Autoencoders Variacionales o VAE) para obtener una mayor cantidad de datos con los que entrenar el modelo. Por el lado de la calidad de los datos, es importante realizar un buen etiquetado de los datos. Esta es una tarea que puede semiautomatizarse pero que siempre requiere de la intervención humana de expertos en el dominio a tratarse.

VI. Conclusiones y Trabajos futuros

Los modelos basados en redes atencionales y embeddings condicionales como BERT superan el rendimiento del modelo propuesto basado en una RNN con arquitectura secuencia a secuencia, inclusive a pesar de aplicar alineamiento y atención de codificador a decodificador. A pesar de ello, se ha comprobado su potencial y facilidad de implementación (en comparación con los modelos del estado del arte) para poder aplicarse en Entendimiento del Lenguaje Natural como parte de sistemas como chatbots.

Como trabajo a futuro, se propone explorar la aplicación, viabilidad e integración de modelos de mayor complejidad como BERT en aplicaciones como chatbots en el idioma español.

References

[1] G. Mesnil, Y. Dauphin, K. Yao, Y. Bengio, L. Deng, D. Hakkani-Tur, X. He, L. Heck, G. Tur, D. Yu, *et al.*, "Using recurrent neural networks for slot filling in spoken language understanding," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 23, no. 3, pp. 530–539, 2014.

[2] B. Liu and I. Lane, "Attention-based recurrent neural network models for joint intent detection and slot filling," *arXiv preprint arXiv:1609.01454*, 2016.

[3] R. Schumann and P. Angkititrakul, "Incorporating asr errors with attention-based, jointly trained rnn for intent detection and slot filling," in *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 6059–6063, IEEE, 2018.

[4] Y. Wang, L. Tang, and T. He, "Attention-based cnn-blstm networks for joint intent detection and slot filling," in *Chinese Computational Linguistics and Natural Language Processing Based on Naturally Annotated Big Data*, pp. 250–261, Springer, 2018.

[5] P. Xu and R. Sarikaya, "Convolutional neural network based triangular crf for joint intent detection and slot filling," in *2013 ieee workshop on automatic speech recognition and understanding*, pp. 78–83, IEEE, 2013.

[6] D. Guo, G. Tur, W.-t. Yih, and G. Zweig, "Joint semantic utterance classification and slot filling with recursive neural networks," in *2014 IEEE Spoken Language Technology Workshop (SLT)*, pp. 554–559, IEEE, 2014.

[7] P. Niu, Z. Chen, M. Song, *et al.*, "A novel bi-directional interrelated model for joint intent detection and slot filling," *arXiv preprint arXiv:1907.00390*, 2019.

[8] Q. Chen, Z. Zhuo, and W. Wang, "Bert for joint intent classification and slot filling," *arXiv preprint arXiv:1902.10909*, 2019.

[9] I. Sutskever, O. Vinyals, and Q. V. Le, "Sequence to sequence learning with neural networks," in *Advances in neural information processing systems*, pp. 3104–3112, 2014.

[10] V. Mnih, N. Heess, A. Graves, *et al.*, "Recurrent models of visual attention," in *Advances in neural information processing systems*, pp. 2204–2212, 2014.

[11] C. T. Hemphill, J. J. Godfrey, and G. R. Doddington, "The atis spoken language systems pilot corpus," in *Speech and Natural Language: Proceedings of a Workshop Held at Hidden Valley, Pennsylvania, June 24-27, 1990*, 1990.

[12] D. Bahdanau, K. Cho, and Y. Bengio, "Neural machine translation by jointly learning to align and translate," *arXiv preprint arXiv:1409.0473*, 2014.

[13] R. Wirth and J. Hipp, "Crisp-dm: Towards a standard process model for data mining," in *Proceedings of the 4th international conference on the practical applications of knowledge discovery and data mining*, vol. 1, Springer-Verlag London, UK, 2000.